VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMATIKOS INSTITUTAS
KOMPIUTERINIO IR DUOMENŲ MODELIAVIMO KATEDRA

Magistro baigiamasis darbas

Trimačių modelių rekonstrukcija iš kelių SEM nuotraukų

Atliko:
Gerimantas Trikšys (parašas)

Vadovė:
dr. Margarita Beniušė (parašas)

Vilnius
2021

Turinys

Anotacija

Šiame darbe pateiksime sprendimą, kaip sudaryti trimatį modelį naudojant nuotraukas, gautas skenavimo elektronų mikroskopu (sutrumpintai SEM), kurios plačiai naudojamos medicinoje, chemijoje, atvaizduoti paviršių. Norint sukurti trimatį modelį reik įvykdyti du pagrindinius etapus: struktūros iš judesio (angl. structure from motion) būdas [67] ir stereoskopinis (angl. multi view stereo) būdas [67]. Struktūros iš judesio etape sprendžiama kameros parametrų gavimo problema (vidiniai parametrai), nuotraukų išdėstymo trimatėje erdvėje parametrų gavimo problema (išoriniai parametrai) ir nevisiškai pilno 3D modelio (angl. sparse) gavimo problema. Stereoskopinio modelio etape yra sprendžiama tikslaus (angl. dense) 3D modelio sukūrimo problema. Šiame etape yra naudojami prieš tai gauti išoriniai ir vidiniai kameros parametrai, vykdomas gylio pagrindo (angl. depth map) radimas tarp nuotraukų ir gautų rezultatų suliejimas į vieną tikslų 3D modelį. Išorinių kameros parametrų gavimas yra svarbus 3D modelio kūrime. Šių parametrų kokybė priklauso nuo kelių dalykų: nuotraukos kokybės ir išorinių kameros parametrų gavimo algoritmų.

Gauti geros kokybės SEM nuotraukas yra sudėtingas procesas, nes nenaudojama šviesa, kaip paprastose fotoaparatuose, o elektronų pluoštas. SEM nuotraukos paprastai turi daug šešėlių, šviesių vietų svarbiose analizuojamo objekto plotuose, netolygių triukšmų. Kai kuriuose analizuojamuose objektuose didžioji dalis gautų nuotraukų ploto neturi išsiskiriančių, ryškių tekstūrų (ypatybių). Tad šiame darbe taip pat analizuosime nuotraukų apdorojimo būdus, kurie pašalina triukšmą, miglą, suliejimą, pagerina nuotraukos tekstūrą, kontrastą.

Išoriniai kameros parametrų kokybė priklauso nuo ypatybių taškų (angl. feature points) gavimo ir šių taškų panaudojimo atitikmenims (angl. feature matches) tarp dviejų nuotraukų rasti algoritmų. Vykdymo metu gali atsirasti pašaliniai taškai (angl. outliers), kurie gaunami dėl nuotraukų nepakankamos kokybės (triukšmo) arba kai nuotraukose yra daug tokių pačių plotų. Tai klaidina algoritmus, kurie randa blogus atitikmenis nuotraukose. Tad darbe lyginami egzistuojantys algoritmai, analizuojama, kaip būtų galima gauti geresnius rezultatus. Be to lyginama su praeitame etape gautomis modifikuotomis nuotraukomis, vykdomi eksperimentai, kada gaunami geresni rezultatai.

Eksperimentinės dalies metu mes pademonstruojame, kokias galimybes, trūkumus turi analizuojami metodai.

Raktažodžiai: Skenavimo elektronų mikroskopijos nuotraukos, trimačio modelio rekonstrukcija, SfM, trimatis taškų debesis, išoriniai, vidiniai kameros parametrai, dirbtiniai neuroniniai tinklai.

Summary

In this research we propose a solution to create 3D model using unordered scanning electron microscope (SEM) images. These images are very popular in medicine, chemistry to display specimen's surface. In order to reconstruct accurate 3D point cloud, you need to execute two steps: structure from motion [67] and multi view stereo [67]. In structure from motion process solves image position in the 3D space problem to get extrinsic (image translation, rotation), intrinsic (camera) parameters and reconstruct sparse 3D point cloud. In multi view stereo process using extrinsic and intrinsic parameters we generate depth maps using input images. Next these depth maps are fused to create a dense 3D point cloud. In later steps it uses algorithms to remove some noisy cloud points using filters and then we generate surface map to fuse with point cloud points. The main problem in this research is to get extrinsic camera parameters. The accuracy of these parameters depends on image quality and extrinsic parameters algorithms.

One of the biggest problems to reconstruct accurate 3D point cloud are input images. It is very difficult task to acquire a perfect SEM image because it needs to employ electrons instead of light that uses phones or cameras to determine specimen's surface. SEM images usually have noise, darkness, brightness, illumination, haze, blur and so on. So, we analyzed methods to enhance SEM image quality.

Extrinsic camera parameters accuracy depends on quality and quantity of feature points (inlier), results of feature matches between two images and its calibration. Other feature points - outliers which come from noise and false matching and have big impact on accuracy of 3D SEM reconstruction. We analyzed existing feature detection, feature matching algorithms and matches removal algorithms and other things. Also, we use previously modified images and analyze how feature matching accuracy improved.

The experiments are presented at the end of the thesis and advantages and disadvantages are evaluated based on the 3D model quality.

Keywords: Scanning electron microscopy images, 3D reconstruction, SfM, feature detection, point cloud, extrinsic, intrinsic camera parameters, deep neural networks.

Įvadas

Šiandien yra begalės sričių, kuriuose yra rekonstruojami trimačiai modeliai panaudojus iš įvairių pusių nufotografuoto objekto nuotraukas, kaip pavyzdžiui viso pasaulio žemėlapio atkūrimas virtualioje aplinkoje (skrendant lėktuvui yra daromos nuotraukos iš, kurių sudaromas trimatis žemėlapis), įvairių kompiuterinių žaidimų kūrimas (naudojama realistiškiems miestams kurti, personažų kūrimui), filmų kūrimui, medicinoje, virtualioje realybėje, 3D objektų spausdinimui, senovinių pastatų atkūrimui virtualioje aplinkoje ir t.t.

Šiame darbe, trimačių objektų kūrimui yra naudojamos skenavimo elektronų mikroskopo (angl. Scanning Electron Microscope - SEM) nuotraukos. SEM yra tokio tipo skenavimo elektronų mikroskopas, su kuriuo galima nuskenuoti tam tikro mėginio paviršių [20]. Nuotraukų gavimas yra kitoks negu paprasto fotoaparato, kuris naudoja šviesą ir šešėlius. Šito tipo mikroskopas sugeneruoja fokusuotus elektronų pluoštus, kurie spinduliuojami į pasirinktą mėginį. Kai kurie pluoštai susigeria į paviršių arba atsispindi. Atsispindėję pluoštai saugo informaciją apie paviršių, ir panaudojus programinę įrangą sukuriamos didelės rezoliucijos SEM nuotraukos [20]. Šios nuotraukos yra plačiai naudojamos chemijoje (analizuoti labai mažo dydžio formos objektus), medicinoje (analizuoti kraujo, audinių mėginius, virusus), biologijoje (analizuoti virusus, bakterijas, gyvūnų audinius). Šios SEM nuotraukos yra naudojamos kuriant trimatį fotografuojamo objekto modelį, kuris padeda lengviau išanalizuoti mėginį, įvertinti objekto dydį, tipą, paviršių nelygumus, formą ir t.t.

Trimačio modelio kūrimas iš nuotraukų yra gana sudėtingas uždavinys, reikalaujantis išspręsti nemažai problemų, itaikant visą aibę įvairių algoritmų. Be to yra nemažai būdų, kaip būtų galima rekonstruoti trimatį modelį. Imamos kelios to pačio objekto nuotraukos, nufotografuotos skirtingu kampu. Nuotraukų kokybė turėtų būti panaši, turėtų būti panašus kontrastas, šviesumas, triukšmingumas. Domina nustatyti nuotraukose esančio objekto poziciją ir kryptį. Todėl reikia sugebėti pozicionuoti tam tikras išsiskiriančias vaizdo vietas - ypatingiuosius taškus (angl. feature points) [67], kurie stipriai išsiskiria iš aplinkos įvairiose to pačio objekto nuotraukose. Tada norint konstruoti 3D modelį, dviejose to pačio objekto nuotraukose reikia nustatyti vietas, kur matomas tas pats ypatingas taškas.

Šiame darbe naudosime struktūros iš judesio (angl. structure from motion - SfM) [67] metodą, kuriuo gaunami vidiniai ir išoriniai kameros parametrai. Vidiniai kameros parametrai, tai pačio fotoaparato rodmenys, kurie išsaugoti exif formatu nuotraukoje. Išoriniai kameros parametrai, kaip posūkis ir postūmis, nusako poziciją trimatėje erdvėje. Po to įvykdome stereoskopinį (angl. multi view stereo - MVS) [67] metodą, kuriuo naudojant prieš tai gautus rezultatus (kameros parametrus) rekonstruojamas tikslus (tankus) trimačio modelio taškų debesis ir išfiltruojami nereikalingi taškai iš trimačio modelio (plačiau aprašysime 1 skyrelyje).

Rasti kameros parametrus išsidėsčiusius trimatėje erdvėje yra gana sudėtingas uždavinys. Rezultatų tikslumas priklauso nuo ypatybių taškų (angl. feature points) [67] nustatymo nuotraukose (objekto kampai, išsiskiriantys taškai), informacijos apie taško vietą ir mažo plotelio aplink ypatybių tašką informacijos, kuris vadinamas deskriptoriumi (angl. descriptor) [27]. Yra lyginami dvejose nuotraukose esančių ypatybių taškų deskriptoriai ir nustatomi taškų atitikmenys, t.y. kuris taškas vienoje nuotraukoje atitinka kurį tašką kitoje nuotraukoje. Taikant šį būdą, galima gauti ir daug blogų atitikmenų. Tad yra taikomi įvairūs filtravimo algoritmai, tačiau ne visados pašalina visus blogus atitikmenis, kartais net pašalina ir gerus atitikmenis. Gavus mažą kiekį atitikmenų, gali neužtekti informacijos gauti tikslius posūkio ir postūmio kameros parametrus.

Viena iš rekonstrukcijos problemų, tai naudojamų nuotraukų kokybė. Naudojant SEM nuotraukas, dėl netinkamai atsispindėjusių spindulių nuo analizuojamo objekto sugeneruotose nuotraukose gali susidaryti daug šešėlių, atsitiktinių šviesių vietų svarbiose analizuojamo objekto

plotuose, netolygių triukšmų ir artefaktų. Be to kai kurie analizuojami objektai turi mažai arba iš vis neturi išsiskiriančių, ryškių tekstūrų (ypatybių), tad ieškant ypatybių taškus (angl. feature points) atitikmenims galime negauti norminio kiekio taškų. Kiekvienoje nuotraukoje gali būti skirtingas triukšmo kiekis ir atitikmenų gavimo algoritmai sunkiai atras gerus atitikmenis. Tačiau naudojant įvairius kompiuterinės regos algoritmus, galima pagerinti ir suvienodinti nuotraukų kokybę.

Yra įvairių būdų, kaip galima rasti nuotraukos atitikmenis: naudojant įvairių kampų, kraštinių taškų aptikimą arba naudoti visus nuotraukoje esančius pikselius. Kitas būdas, kuris šiuo metu tyrimuose plačiai naudojamas, tai mašininio apmokymo būdas. Mašininis apmokymas, tai tam tikra kategorija algoritmų, kurie leidžia būti kas kartą tikslesniems (po truputėlį pagerinti našumą) tam tikrame prognozavime, nereikalaujant suprogramuoti geresnio algoritmo. Šie algoritmai turi naudoti tam tikrus statistinės analizės algoritmus, kurie prognozuotų išvesties rezultatus, naudojant įvesties duomenis, ir be to galėtų patikslinti išvesties rezultatus naudojant naujus duomenis. Šie algoritmai labai plačiai naudojami tyrimuose, kaip medicinoje, finansuose, reklamoj, versle. Viena iš šio būdo dalių, kurį mes šiame darbe analizuosime - gilusis apmokymas (angl. deep learning) [19]. Šie algoritmai audoja „sluoksnių“ struktūrą (neuroninius tinklus), su kuriais patys gali apsimokyti ir nustatyti, kokios ypatybės apmokymo duomenyse yra svarbiausios. Šis būdas jau po truputėlį naudojamas ir trimačio modelio sukūrimui, kaip ypatybių taškų radimui, atitikmenų gavimui ar net trimačio modelio sukūrimui iš vienos nuotraukos ar nuotraukos pozicijos radimas trimatėje erdvėje.

Šio darbo tikslas – pasiūlyti 3D modelio iš SEM nuotraukų kūrimo žingsnius ir naudotinus algoritmus, išanalizuoti gautus rezultatus ir pasiūlyti, kaip pagerinti trimačio modelio kūrimą, taikan įvairius sprendimus kūrimo žingsniuose.

Darbo uždaviniai:

- Išanalizuoti nuotraukų pagerinimo būdus, trimačio modelio rekonstrukcijos sukūrimą, neuroninių tinklų struktūrą ir kitus reikalingus metous. Pateikti susijusių darbų analizę.
- Pateikti sprendimą, kaip apdoroti SEM nuotraukas, taip kad nebūtų triukšmų ir tamsių ir šviesių vietų naudojant kompiuterinės regos algoritmus.
- Pateikti sprendimą, kaip mes gausime tikslius kameros parametrus vykdant struktūros iš judesio (angl. structure from motion) [67] etapą.
- Pateiksime eksperimentus naudojant pagerintas ir įvesties SEM nuotraukas įvykdant 3D rekonstrukcijos etapus.

Trumpas skyrelių aprašymas

Šio baigiamojo dar darbas susideda iš tokių skyrelių:

- Pirmame skyrelyje aptariami su šia tema susiję darbai. Pateikiama, kaip gaunamos SEM nuotraukos (poskyris paimtas iš magistrinio tiriamojo darbo). Aprašomi pagrindiniai trimačio modelio kūrimo etapai.
- Antrajame skyrelyje aptarsime neuroninius tinklus ir jų struktūrą.
- Trečiame skyrelyje pateiksime sprendimo architektūrą, kaip susidarys mūsų tiriamasis darbas. Po to pateiksime, su kuriomis nuotraukomis mes eksperimentuosime.
- Ketvirtame skyrelyje pateiksime būdus, kaip būtų galima pagerinti nuotraukos kokybę naudojant giliąsias apmokančias sistemas ir kitus algoritmus. Pirmiausiai pateiksime, kokias nuotraukų apdorojimo problemas mes spręsime šioje dalyje. Pateiksime

sprendimus, kaip pašalinti šviesius, tamsius plotus, triukšmą, miglą, suliejimą (angl. out of focus) ir pagerintų nuotraukos kontrastą. Pateiksime rezultatus.

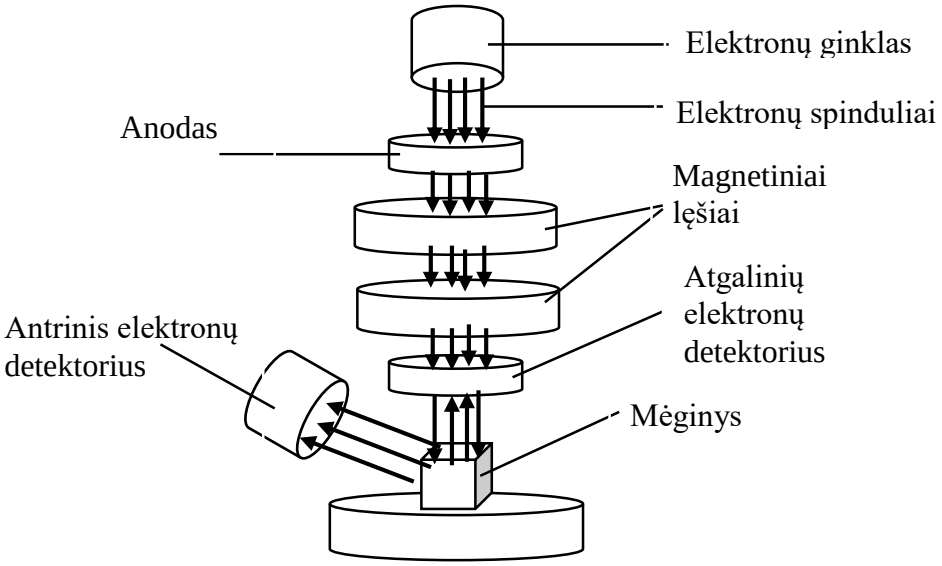
- Penktame skyrelyje pateiksime sprendimus, kaip būtų galima patobulinti struktūros iš judesio (angl. structure from motion) būdą, kad galėtume tiksliai gauti kameros parametrus. Pirmiausiai palyginsime egzistuojančius algoritmus - ypatybių gavimas, atitikmenų gavimas ir jų filtravimas. Pateiksime testavimo rezultatus.
- Paskutiniame skyrelyje aptarsime mūsų darbą, pateiksime išvadas ir gautus rezultatus. Be to aptarsime, kokius patobulinimus būtų galima padaryti ateinančiuose darbuose.

1. Susijusių darbų analizė

Pirmiausiai šiame skyrelyje pateiksime, kaip gaunamos skenavimo elektronų mikroskopu SEM nuotraukos, kokios problemos iškyla sudarant nuotraukas ir kaip galima šias problemas spręsti. Be to šiame skyriuje pateiksime trimačio modelio rekonstrukcijos etapą. Trumpai kiekvienoje dalyje aprašysime susijusių darbų analizę.

1.1. SEM nuotraukų sudarymas

Skenavimo elektronų mikroskopas, tai toks elektronų mikroskopas, kuris leidžia detaliai nuskenuoti labai mažus mėginius naudojant elektronų spindulius. Šie mikroskopai yra daugiausia naudojami chemijoje ir biologijoje. Aprašysime kaip sudaromos SEM nuotraukos. Iliustracijoje (1 pav.) atvaizduojamas skenavimo elektrono mikroskopo struktūra.



1 pav. Skenavimo elektronų mikroskopo struktūra

Mikroskopas susideda iš tokių dalių:

Vakuuminės kameros. Mikroskopas susideda iš dviejų kamerų: objekto vakuuminės kameros ir spindulio vakuuminės kameros. Šie mikroskopai naudoja vakuumą, kad galėtų spinduliuoti elektronų spindulius į analizuojamą objektą. Šių vakuuminių kamerų paskirtis - apsaugoti, kad oro dalelės neįeitų į vidų ir leidžia pakreipti analizuojamą objektą tam tikru kampu.

Elektrono ginklas. Sukuria elektronų pluoštus vakuume. Šis prietaisas susideda iš 3 pagrindinių dalių: gijos, skydo, anodo. Gija, tai metalinė viela, kurią pakaitinus, vakuume sukuriami elektronai, kurie naudojami pluoštui sudaryti [53]. Skydas padeda nukreipti elektronus tam tikra linkme (pagal atvaizduotą modelį, skydas elektronus nukreipia žemyn). Anodas, tai metalinė plokštelė, kuri turi teigiamą įkrovą. Anodas gautus iš gijos elektronus sujungia į vieną spindulį ir išspinduliuoja per mažą skylutę dideliu greičiu [20].

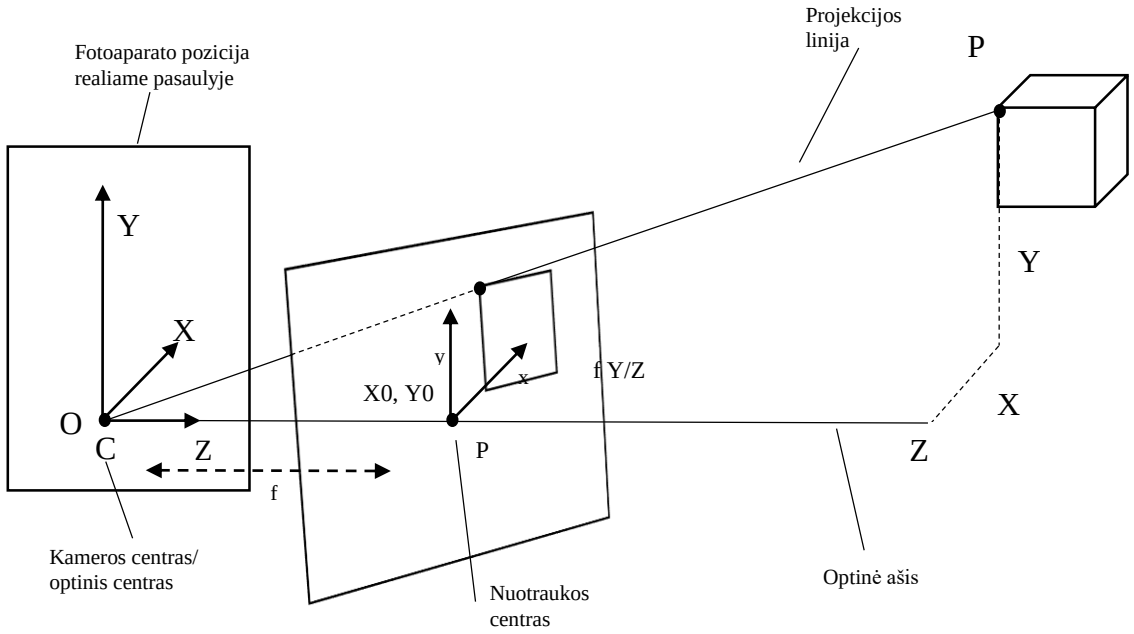
Magnetiniai lęšiai. Šie lęšiai padeda sudaryti aiškias nuotraukas. Lęšiai yra sudaryti ne iš stiklo, bet iš magnetų [4], kurie padeda reguliuoti, į kokią analizuojamo objekto vietą nukreipti pluoštą.

Ritės. Šios ritės yra šalia paskutinių lęšių. Šios ritės sukuria magnetinį lauką naudojant svyruojančią įtampą, kuri leidžia kontroliuoti elektrono signalą [21], pavyzdžiui norima padidinti analizuojamo objekto vaizdą.

Detektoriai. Spinduliuojant elektronus į analizuojamą objektą atsispindi du skirtingi pluoštai: atgaliniai elektronai (angl. BSE) ir antriniai elektronai (angl. SE). Atgaliniai elektronai atsispindi priešinga krypti nuo analizuojamo objekto tokiu pat greičiu ir energija. Dėl didelės energijos pluoštai gali giliau keliauti į mėginio vidų ir net kartais gali įsiskverbti į patį paviršių, tai pluoštai gali turėti mažiau informacijos apie analizuojamo objekto paviršių [42]. Dėl šių priežasčių BSE nuotraukos turi mažesnę rezoliuciją negu SE [4]. Antriniai elektronų signalai susidaro ant analizuojamo objekto paviršiaus ir turi mažą energiją. Šių signalų detektoriai pastatyti tam tikru kampu (1 pav.), kad būtų galima efektyviai sugerti šio tipo signalus [24]. Jeigu norima gauti SEM nuotraukos, kurios turi didesnę rezoliuciją, reikia sumažinti elektrono signalo energiją. Tačiau sumažinus per daug, nuotraukose atsiranda įvairūs triukšmai.

1.2. Standartinis kameros modelis

Standartinis kameros modelis [67] nurodo geometrinį sąryšį tarp trimačių taškų ir jų projekcijų dvimatėje nuotraukoje. Visos kameros turi skirtingus kameros modelius: skylių modelis, žuvies akies kameros modelis ir t.t. Šiame darbe pateiksime paveikslėlyje (2 pav.) paprasčiausią kameros modelį – skylių modelis (angl. pinhole camera), kuriame aprašoma taškų projekcijos 3D erdvėje į vaizdo plokštumą sąryšis. Tegul projekcijos centras yra Euklido koordinačių sistema ir plokštuma Z = f, vadinamas nuotraukos plokštuma (židinio plokštuma). 3D taškas su koordinatėmis $(X, Y, Z)^T$ yra atvaizduotas 2D nuotraukos plokštumoje (koordinatės) $(\frac{fX}{Z}, \frac{fY}{Z})^T$.



2 pav. Pavaizduotas standartinis kameros modelis pagal nuotrauką ir trimatį modelį

Kamera turi vidinius (angl. intrinsic) ir išorinius (angl. extrinsic) parametrus. Vidiniai kameros parametrai apibūdina kameros optinius parametrus, kaip židinio nuotolis, o išoriniai apibūdina nuotraukos poziciją ir orientaciją. Pirmiausia pateiksime vidinių kameros parametrų (kalibracijos) matricos išraiką, kuri pateikta (1.1) punkte:

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{f_x} & \boldsymbol{s} & \boldsymbol{x_0} \\ \boldsymbol{0} & \boldsymbol{f_y} & \boldsymbol{y_0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{1} \end{bmatrix} \quad \textbf{(1.1)}$$

- $\boldsymbol{f_x}, \boldsymbol{f_y}$ – tai židinio nuotolio parametrai. Šie parametrai nusako atstumą tarp kameros objektyvo, fotografuojamo objekto ir sudarytos analizuojamo objekto nuotraukos. Šioje matricoje abu židinio nuotolio parametrai yra tokie patys. Tačiau nuo kelių priežasčių gali būti skirtingi parametrai: klaidos kameros kalibravimo metu, kameros objektyvas sukelia iškraipymą, nuotraukos rezoliucijos pakeitimas po apdorojimo.
- $\boldsymbol{x_0}, \boldsymbol{y_0}$ – tai analizuojamos nuotraukos vidurys.
- $\boldsymbol{s}$ – tai ašies kreivumo parametras. Šis parametras sukelia nuotraukos iškraipymą.

$$(\boldsymbol{R}, \boldsymbol{t}) = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & \boldsymbol{1} \end{bmatrix} \quad \textbf{(1.2)}$$

Kameros poziciją galime pateikti šia 4x4 transformacijos matrica (1.2). Išoriniai kameros parametrai – R (rotacija) yra 3x3 matrica ir t (transliacija) yra 3x1 dydžio matrica, kurios nurodo kameros vietą ir orientaciją atsižvelgiant į žinomo pasaulio orientaciją. Orientacija turi tris pasukimo ašis - xyz, kurios dar vadinami Eulerio ašimis. Norint apskaičiuoti bendrą rotacijos (posūkio) parametrą, pirmiausiai reikia apskaičiuoti rotaciją kiekvienai ašiai - R_x R_y R_z. Pateiksime matricų išvedimus, kurie aprašyti šiame darbe [55]:

$$\boldsymbol{R_x}(\boldsymbol{\psi}) = \begin{bmatrix} \boldsymbol{1} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\cos\psi} & \boldsymbol{-\sin\psi} \\ \boldsymbol{0} & \boldsymbol{\sin\psi} & \boldsymbol{\cos\psi} \end{bmatrix} \quad \textbf{(1.3)}$$

$$\boldsymbol{R_y}(\boldsymbol{\theta}) = \begin{bmatrix} \boldsymbol{\cos\theta} & \boldsymbol{0} & \boldsymbol{\sin\theta} \\ \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{0} \\ \boldsymbol{-\sin\theta} & \boldsymbol{0} & \boldsymbol{\cos\theta} \end{bmatrix} \quad \textbf{(1.4)}$$

$$\boldsymbol{R_z}(\boldsymbol{\phi}) = \begin{bmatrix} \boldsymbol{\cos\phi} & \boldsymbol{-\sin\phi} & \boldsymbol{0} \\ \boldsymbol{\sin\phi} & \boldsymbol{\cos\phi} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{1} \end{bmatrix} \quad \textbf{(1.5)}$$

Norint gauti bendrą rotacijos matricą, reikia įvykdyti šią formulę:

$$\boldsymbol{R} = \boldsymbol{R_z}(\boldsymbol{\psi}) * \boldsymbol{R_y}(\boldsymbol{\theta}) * \boldsymbol{R_x}(\boldsymbol{\phi}) \quad \textbf{(1.6)}$$

kur ψ, θ, ϕ yra z, y, x ašies pasukimas, laipsniais.

1.3. Trimačio modelio sudarymas

Trimačio modelio sukūrimas yra gana sudėtingas procesas, kuris susideda iš nemažai vykdomų užduočių. Trimatį modelį galima sudaryti keliais būdais [67]:

- Vienos pusės vaizdo (angl. single-view) rekonstrukcija. Šis būdas naudoja šviesą (SEM nuotraukoms - elektronų pluoštus), kuriame vykdomas skirtingų pozicijų šviesos spinduliavimas ant vienos objekto pusės, kuris nėra sukamas. Taip gauname skirtingų pusių šešėlius, iš kurių sukuriame trimatį modelį.
- Iš įvairių pusių (angl. multi-view) rekonstrukcija. Šis metodas naudoja įvairių pusių nufotografuoto objekto nuotraukas. Yra vykdomas ypatybių taškų radimas nuotraukose ir po to vykdomas šių taškų atitikmenų radimas tarp nuotraukų porų.
- Hibridinis rekonstrukcijos būdas. Šis būdas sujungia vienos pusės vaizdo rekonstrukcijos ir įvairių pusių rekonstrukcijos pagrindines savybes.

Vienos pusės vaizdo ir hibridinis rekonstrukcijos metodai turi ir neigiamų savybių, kurios gali pabloginti trimačio modelio kūrimą. Pirma abu rekonstrukcijos metodai yra per daug sudėtingi kurti trimatį modelį. Naudojant vienos pusės vaizdo rekonstrukciją, matysime tik iš vienos pusės tikslų trimatį analizuojamą objektą, o naudojant hibridinį rekonstrukcijos būdą, būtų tiek laiko, tiek resursų gaišimas. Tad geriausias iš šių metodų - įvairių pusių rekonstrukcija. Šis metodas susideda iš dviejų skirtingų rekonstrukcijos būdų: iš reto (angl. sparse) taškų debesies sudarymo metodo, kuris vadinamas „struktūra iš judesio“ metodas (angl. structure from motion, sutrumpintai SfM) [67] ir tankaus (angl. dense) taškų debesies sudarymo metodo, kuris dar vadinamas stereoskopiniu (angl. multi-view stereo, sutrumpintai MVS) [67] rekonstrukcijos sudarymu.

1.4. Struktūra iš judesio (angl. structure from motion) trimačio modelio kūrimo metodas

„Struktūra iš judesio“ (angl. structure from motion, sutrumpintai SfM) [67] - tai vienas iš metodų, kuris gali rekonstruoti trimatį modelį ir gauti reikalingus kameros parametrus (išorinius ir vidinius) naudojant nuotraukų sąrašą. Be to šis būdas tinka ir kai nuotraukos nėra surūšiuotos tam tikra seka. Šis sprendimas susideda iš kelių žingsnių: ypatybių taškų (angl. feature point) radimas, ypatybių taškų atitikmenų radimas nuotraukų porose, blogų atitikmenų (angl. outliers) pašalinimas. Tada vykdoma linijinė trianguliacija pagal matricas, kurios gautos epipolarinės geometrijos vykdymo metu ir trimačio modelio rekonstravimo metu, yra vykdomas paketų koregavimas (angl. bundle adjustment), kuriuo metu yra tikslinami kameros parametrai ir taškų debesis (angl. point cloud). Po kiekvienos naujos nuotraukos įvedimo į 3D modelį yra naudojama n taško perspektyva (angl. perspective-n-point)[38], kuriu metu vykdoma naujos nuotraukos kameros pozicijos nustatymas. Įvykdžius visus skaičiavimus yra gaunami vidiniai kameros parametrai, kaip židinio nuotolis, kameros centro taškas ir išoriniai kameros parametrai: kameros pozicijos, postūmio matricos. Šį būdą naudoja nemažai laisvai prieinamų, atviro kodo programų, kaip VisualSFM [61], COLMAP [15] ir kitos.

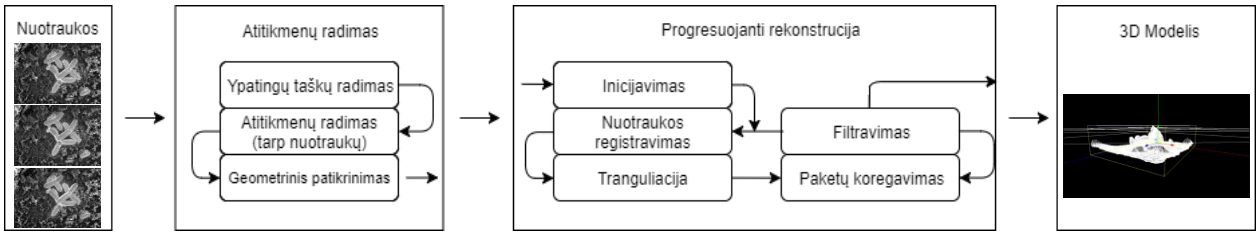
Struktūra iš judesio metodą galima vykdyti dviem būdais [67]:

- Progresuojantis (angl. incremental) rekonstrukcijos metodas, kuris po kiekvienos inicijuotos nuotraukos vykdo trianguliaciją. Nors šis būdas randa mažai neteisingų atitikimų (angl. outliers) tarp nuotraukų, tačiau yra lėtokas - reikalauja daug skaičiavimų, naudojant paketų koregavimo (angl. bundle adjustment) ir neatitikimų filtravimo

metodus. Paketų koregavimas vykdomas po kiekvienos inicijuotos nuotraukos ir 3D taškų debesies sudarymo pabaigoje vykdomas globalus koregavimas.

- Globalus (angl. global) rekonstrukcijos metodas, kuris vykdo kameros pozicijų skaičiavimą naudojant visas nuotraukas tuo pačiu metu. Šis metodas yra daug greitesnis už progresuojantį būdą. Galima lengviau rekonstruoti didelio masto trimačius modelius, tačiau labai jautrus neatitikimams (angl. outliers).

Šiame darbe naudosime progresuojantį struktūros iš judesio metodą, nes naudojame SEM nuotraukų rinkinius, kuriuose pateikta gana mažai nuotraukų ir kiekvienos nuotraukos pasukimo kampas yra didelis, tad turime turėti gana tikslius atitikmenis, kad gautume tikslius kameros parametrus. Iliustracijoje (3 pav.) pateiksime šio metodo struktūros modelį. Sekančiuose skyreliuose plačiau aprašysime kiekvieną vykdomą sprendimą.



3 pav. Trimačio modelio struktūra naudojant progresuojančią rekonstrukcijos metodą

1.4.1. Ypatybių taškų ir deskriptorių gavimas

Struktūra iš judesio metodas [67] rasti kameros išorinius parametrus (poziciją, orientaciją) priklauso nuo atitikmenų radimo tarp dviejų nuotraukų. Atitikmenis galima rasti ir nenaudojant ypatybių taškų gavimo algoritmo, tačiau tada naudosime visus nuotraukoje esančius pikselius. Todėl gali stipriai padidėti skaičiavimo laikas (priklausomai nuo nuotraukos rezoliucijos) ir gali atsitikti, kad dauguma gautų atitikmenų tarp nuotraukų poros nebus tikslūs. Taikant ypatybių taškų gavimą, mes supaprastiname nuotraukoje esančių objektų pavidalą, kad skaičiavimo programoms būtų lengviau skaičiuoti. Tai viena iš priežasčių, kodėl mes naudojame ypatybių taškų gavimo algoritmus.

Ypatybės taškas susideda iš dviejų dalių: raktinis taškas (angl. keypoint) ir deskriptoriaus (angl. descriptor) [27]. Raktinis taškas, tai tam tikras taškas (pikselis), kuris yra išsiskiriantis iš kitų nuotraukoje esančių taškų, kaip pavyzdžiui kampai, kraštinės arba kitos įdomios vietos. Deskriptoriai, tai aplink raktinį tašką esantys nedideli plotai, kurie apibrėžia raktinio (ypatybių) taško aplinkos savybes. Deskriptoriai yra reikalingi, kad galėtume pozicionuoti tą patį tašką dvejose skirtingose objekto nuotraukose. Deskriptoriai turi būti labai atsparūs dydžio, formos, pasukimo, triukšmo pasikeitimams. Viena iš problemų naudojant deskriptorius, tai jeigu nuotraukoje atvaizduojama daug panašių objektų (pvz. raštas ant kilimo), gali atsitikti, kad atitikimų paieškos metu bus aptinkami ne tie plotai. Ši problema sprendžiama vykdant atitikmenų filtravimą.

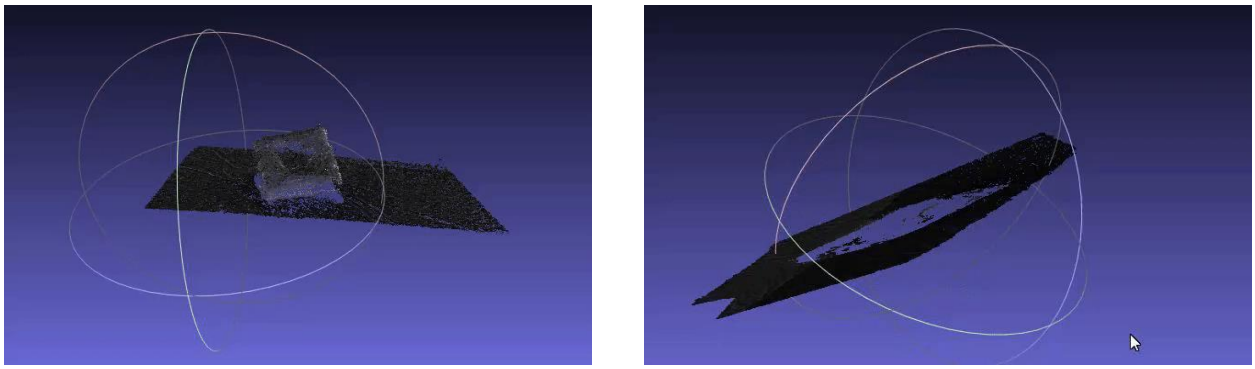
Rasti gerus ypatybių taškus, detektorius turi turėti nemažai savybių[58]:

- **Tikslumas**: ypatybių taškų gavimo algoritmai turi tiksliai atrasti tą patį tašką ir būti toje pačioje pikselių vietoje tarp dviejų ar daugiau skirtingų nuotraukų. Ši savybė yra svarbi, kai yra vykdomas plataus kampo pasukimo tarp objektų taškų atitikmenų radimas, pozicijos radimas ar struktūros iš judesio rekonstrukcija. Jeigu tikslumo nebus arba kai kurie atitikmenys nebus tikslūs (taškas vienoje nuotraukoje prilyginamas netinkamai vietai

kitoje nuotraukoje), tai gausime blogus nuotraukų kameros parametrus, kurie pakenks trimačio modelio kūrimui.

Pateiktoje iliustracijoje (pav. 4) matome, kaip po blogų atitikmenų radimo blogai urekonstruojamas objektas.

- **Pakartojamumas**: Jeigu nuotraukos yra pasuktos, pertemtos arba atsiranda artifaktai po nuotraukų kompresijos ar daug triukšmo, deformacijų (nuotraukos dydžio didinimas), tai ypatybių taškų gavimo algoritmai turi būti kuo mažiau jautrūs šiems parametrams, kad atrastume tuos pačius taškus.
- **Našumas**: pateikti algoritmai turi veikti greitai. Bet kartais greitis gali ir trukdyti didesniam taškų kiekiui nuotraukoje atrasti. Dar svarbu, kokia užduotis vykdoma. Pavyzdžiui jeigu yra rekonstruojama aplinka, kai savarankiškai be vairuotojo pagalbos važiuoja automobilis ar veido mimikos atpažinimas, tai reikės labai greito detektoriaus aptikti aplinkos ypatybių taškus. Tačiau mūsų darbe algoritmų greitis nelabai svarbus, mūsų darbo tikslas gauti kuo detalesnį trimatį modelį.
- **Bendrumas**: algoritmai galėtų atrasti taškus skirtingose užduotyse, kaip trimačio modelio kūrime, objektų radime, segmentavime ir t.t.
- **Kiekis**: algoritmas turi atrasti daug atitikmenų taškų net ir mažos rezoliucijos nuotraukoje. Tačiau priklausomai nuo užduoties gali reikėti daug taškų, kurie apima visą nuotrauką (objekto aptikimui) arba tik tam tikro nuotraukoje esančio objekto kraštinius taškus.



4 pav. Trimačio modelio sukūrimo pavyzdys naudojant tas pačias įvedimo nuotraukas. Kairėje pusėje pavaizduotas tikslus trimatis modelis, o dešinėje dėl netikslaus atitikmenų radimo rekonstruotas netikslus trimatis modelis

Šiandien yra nemažai sprendimų, kaip būtų galima išgauti raktinius taškus ir deskriptorius, naudojant tradicinius (angl. hand crafted) ar apmokomus algoritmus (angl. learned).

Tradiciniai detektoriai ir deskriptoriai

Yra nemažai tradicinio tipo algoritmų, kurių kiekvienas iš jų turi skirtingas savybes, kaip gaunami ypatybių taškai ir jų deskriptoriai. Pateiksime keletą populiarių algoritmų, kurie yra dar iki šiol naudojami tyrimams ir rekonstruoti trimačius modelius.

SIFT [37] – (angl. Scale Invariant Feature Transform) algoritme vykdomi du etapai: Masto erdvės kraštutinumo aptikimas, kur Gauso skirtumas (angl. Difference of Gaussian) yra taikomas identifikuoti taškų variantus mastelo keitimui. Po to atliekamas lokalaus ekstremumo patikrinimas su gretimais pikseliais. Kitas etapas - vykdomas raktinių taškų lokalizavimas – atmeta mažo kontrasto taškus ir atmeta taškus, kurie nėra ant kraštų naudojant Hessian matricą. Deskriptorius kuriamas dviem etapais: orientacijos priskyrimas, kuris formuoja orientacijos histogramas iš

vietinio gradiento, kad galėtume nusakyti taško dominuojančią kryptį. Po to taško deskriptoriaus vektorius yra konstruojamas pagal taškus ir aplinką.

SURF [44] detektorius, kuris yra paremtas pagal Hessian- Laplace detektorių, kur Hessian matrica prisiartina prie našumo naudojant dėžutės tipo filtro rinkinį ir joks lyginimas netaikomas pereinant nuo vienos skalės prie kitos. Deskriptorius apskaičiuojamas naudojant Haar filtro atsakymus naudojant integralo nuotraukas, kuris leidžia daug greičiau atrasti ypatybes.

KAZE [2] ypatybių detektorius turi panašią aptikimo veikimo eigą, kaip ir SURF algoritmas, tačiau 2D ypatybės yra aptinkamos ir aprašomo pagal netiesinio mastelio skalę netiesinio difuzinio filtravimu. Detektorius yra paremtas Hessian matricos determinantu. Kaze algoritme sudarant deskriptorių yra naudojamas M-SURF metodas.

AKAZE [3] algoritmas, kaip KAZE [2] taip pat remiasi netiesinio difuzinio filtravimu, tačiau netiesinio mastelio skalė rekonstruojama naudojant efektyvią struktūrą - greitai tiksli difuzija (angl. Fast Explicit Diffusion). Detektorius yra paremtas Hessian matricos determinantu. Deskriptorius paremtas modifikuoto vietinio skirtumo dvejetainiu (angl. Modified Local Difference Binary). Algoritmo ypatybės šiame metode neleidžia pakisti masteliui, rotacijai.

Gilaus apmokymo detektoriai ir deskriptoriai

Neuroninių tinklų apmokymas yra plačiai naudojamas objektų radimui, objektų segmentavimui, filtravimui ir t.t. Tačiau yra galimybė šiuos neuroninius tinklus sudaryti, tai kad būtų galima atrasti ypatybių taškus, sudaryti deskriptorius ar net surasti nuotraukos orientaciją trimatėje erdvėje. Pateiksime keletą neseniai sukurtų algoritmų:

GeoDesc [39] –neuroninio tinklo deskriptorius, kuriame integruoja geometrijos apribojimus iš kelių vaizdų rekonstrukcijos, kad būtų galima gauti naudos mokymosi procesui tobulinant mokymo duomenis.

ContextDesc [40] –neuroninio tinklo deskriptorius, kuriame sukuriama vieningą mokymosi sistemą, kuri panaudotų ir kauptų įvairiarūšę kontekstinę informaciją, kaip vizualinį kontekstą iš aukšto lygio nuotraukos vaizdavimo ir geometrinis kontekstas iš 2D ypatybės. Be to pateikia nuostolio funkciją, kuri vengia empirinės hiperparametrų paieškos ir pagerina konvergenciją.

LF-Net (Local Feature Network) [43] Šio algoritmo veikimas: pirmiausiai paleidžia detektorių ant pirmos nuotraukos ir randa maksimumus sudarytame balų plane. Po to vykdomas svorių optimizavimas, kad kai bus vykdomas detektorius ant antros nuotraukos sudarytų švarų atsakymo planą su maksimumais tinkamose vietose.

SuperPoint [22] – prižiūrimas (angl. supervised) neuroninis tinklo modelis, kuriame vykdomas ypatybių ir detektoriaus gavimas vykdant ant pilno dydžio nuotraukos. Pirmiausia naudojant VGG stiliaus koduotojas (angl. encoder), kuriame sumažinama įvestos nuotraukos dimensija. Po to neuroninis tinklas persiskiria į ypatybių gavimo ir deskriptoriaus dekoderius.

LIFT [63] – pirmasis giliojo apmokymo metodas, kuriame yra ypatybių gavimas (sukuria gerų ypatybių taškų balų planą), orientacijos apskaičiavimas (numatyti gabaliuko (angl. path) orientaciją) ir deskriptoriaus išgavimas. Balų planas (angl. score plan) yra naudojamas iškirpti gerus ypatybių taškus. Apmokyme naudojama daug žingsnių ir reikalauja naudoti struktūros iš judesio būdą. Be to LIFT sudaryta struktūra nesidalija skaičiavimais, tad programoms, kuriose vykdomi skaičiavimai realiuoju laiku vykdomi per lėtai.

D2-Net [18] – naudoja vieną konvoliucinį neuroninį tinklą, iš kurio yra gaunami ypatybės ir jų deskriptoriai. Yra sukuriamas apibūdinti – aptikti sprendimas, kuriame nevykdomas ypatybių gavimas naudojant mažo lygio informaciją. Pirmiausiai yra naudojamas gilieji konvoliuciniai neuroniniai tinklai sudaryti ypatybių planą, kurie buvo naudojami deskriptoriams sudaryti ir rasti raktinius taškus. Rezultate ypatybių detektorius yra stipriai susijęs su deskriptoriumi.

1.4.2. Atitikmenų radimas tarp nuotraukų poros ir blogų atitikmenų pašalinimas

Gavus ypatybių taškus ir jų deskriptorius yra vykdomas deskriptorių vektorių palyginimas naudojant artimiausio kaimyno paieška (angl. nearest neighbor search) [46], o tiksliau apytikslį artimiausią kaimyną (angl. approximate nearest neighbor). Artimiausio kaimyno paieška randa artimus taškus k tam tikrame taškų rinkinyje (duomenų bazėje, kuriame yra n taškų), pagal nurodytą tašką naudojant Euklido atstumo sprendimą. Šis būdas yra plačiai naudojamas, kaip atpažinimui, mašininio apmokyme, duomenų bazių užklausoms. Tačiau realioms problemoms spręsti yra naudojama didelių dimensijų vektoriai. Šis būdas nepriimtinas ir neefektyvus, tad yra naudojamas apytikslis (angl. approximate) sprendimas. Šių metodų grupė randa taškus esančius atstumu (1 + e) R nuo užklausos taško, kur R yra atstumas tarp užklausos taško ir tikrojo artimiausio kaimyno. Šis sprendimas turi nemažai algoritmų, su kuriais galima įvykdyti atitikmenų gavimą: FLANN [41], FAISS [25], kd-medis [7] ir .t.t

Gali atsirasti blogi atitikmenys (angl. outliers), kai nuotraukų pora stipriai skiriasi fotografavimo kampu, pasikartojančių tekstūrų ir pan. Blogų atitikmenų pašalinimui galima naudoti Ransac (angl. random sample consensus) [23] algoritmą. Šis sprendimas veikia, taip: pirmiausiai pakartotinai sugeneruojami modeliai iš mažų, atsitiktinių taško pogrupių. Po to yra apskaičiuojama kiek atitikmenų yra pagal visus taškus ir sprendžiama, kurį modelį pasirinkti, kuriame yra didžiausias kiekis. Yra nemažai sprendimų, kurie pagerina rezultatų kokybę PROSAC [12], MAGSAC [6], DEGENSAC [11] ir t.t.

1.4.3. Nuotraukų registravimas

Nuotraukų registravimas - tai pirmas progresuojančio žingsnio etapas, kuriame gauname absoliučią (rotaciją R ir transliaciją t) kameros poziciją. Apskaičiuoti naujos kameros poziciją yra vykdomas 2D-3D atitikmenų gavimas tarp naujos nuotraukos ypatybių taškų ir 3D debesies taškų. Šiai problemai yra naudojamas perspektyva iš n-tojo taško (angl. perspective-n-point) [38] sprendimas. Po kiekvieno sprendimo vykdymo galime gauti blogus atitikmenis, tad prieš tai vykdomam sprendimui yra pridedamas RANSAC algoritmas, kuriame vykdoma pašalinių atitikmenų pašalinimas.

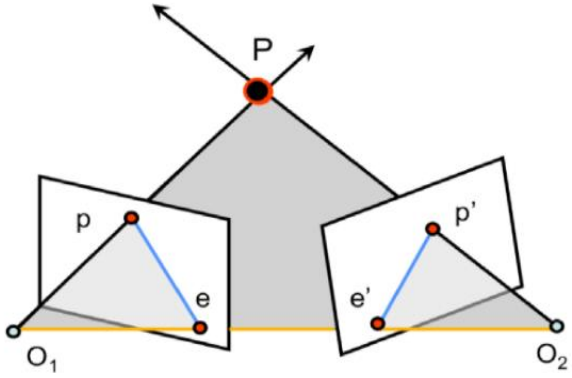
1.4.4. Trianguliacija

Trianguliacija [34] – metodas, sudaryti 3D debesies taškus. Šio sprendimo metu naudojama nuotraukų pora, kuri turi daug bendrų atitikmenų sudaryti 3D taškus ir kameros poziciją tarpusavyje parametrus. Išspręsti šią problemą yra naudojama (angl. epipolar constraint) epipolinis suvaržymas [16]. Pateiksim sąvokas, naudojamas iliustracijoje (pav.5):

- Epipolinė geometrija – geometrija, kuriame pateikiamos dvi kameros, kurios stebi tą pačią 3D tašką P ir jų projekcijos nuotraukose pateikiamos p ir p’.
- Bazinė linija – linija tarp kameros centrų, kurie pavaizduoti O1 ir O2 taškuose.
- Epipolijos plokštuma – plokštuma nusako kameros centras P taškas.
- Epipolės - Vietos kur bazinė linija kerta nuotraukos plokštumą (e ir e‘).
- Epipolijos linija – susijungimo linija tarp epipolijos plokštumos ir nuotraukos plokštumos. Leidžia identifikuoti taško poziciją kitoje nuotraukoje.

Surasti atitikmenį kitoje nuotraukoje nereikia ieškoti per visą nuotrauką – reikia panaudoti epipolinę liniją, taip pagerindami tikslumą ir sprendimo greitį, kuris vadinamas epipoliniu suvaržymu.

Rasti epipolines linijas ir epipoles reikia apskaičiuoti esminę ir fundamentalią matricas. Esminė matrica (angl. essential matrix) – joje pateikiama informacija, kaip pagal pirmąja nuotrauką naudojant transliaciją ir rotaciją yra pakreipta nuotrauka. Fundamentali matrica (angl. fundamental matrix) – tas pats, kaip ir esminė matrica, tačiau joje taip pat yra ir vidiniai abiejų kameros parametrai. Tačiau dėl netikslumų, taškai gali nekirsti epipolinės (angl. epipolar) linijos tarp nuotraukų poros, tad gauname reprojekcijos paklaidą.



5 pav. Epipolinis geometrijos struktūra. Pilka spalva pateikia epipolijos plokštuma, geltona spalva pateikia bazinė liniją ir mėlyna spalva pateikia epipolijos linijas.

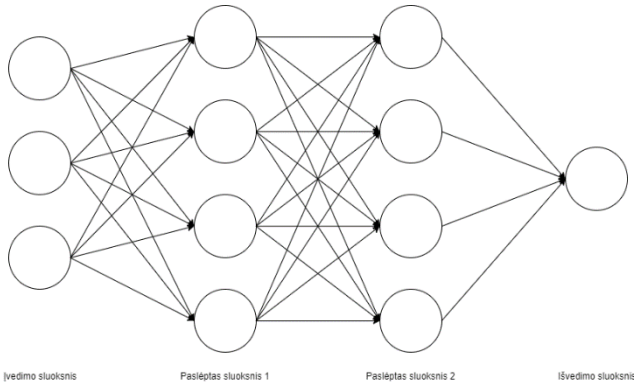
1.4.5. Paketų koregavimas

Po kiekvienos kameros pozicijos apskaičiavimo ir trianguliacijos gali sugeneruoti tam tikrus netikslumus. Sumažinti paklaidas yra naudojamas paketų koregavimas [59] (angl. bundle adjustment). Šis būdas apima pakartotinį kamerų ir 3D taškų debesies optimizavimą, kuriame vykdo reprojekcijos paklaidos mažinimą tarp nuotraukų pozicijų ir taškų. Reprojekcijos paklaida yra taškų atstumas tarp kiekvienos stebimos 2D ypatybės realios pozicijos ir apskaičiuoto jos rekonstruoto 3D atitinkamo taško atitikmens, remiantis dabartiniu kameros ir objekto parametrų įvertinimu. Paklaidos mažinimui yra naudojamas mažiausių kvadratų (angl. least-squares) algoritmas. Vienas iš populiarų algoritmų Levenberg-Marquardt [59] algoritmas. Progresuojančiame struktūros iš judesio metode paketų koregavimas yra vykdomas keliais būdais: vykdomas po kiekvienos inicijuotos nuotraukos ir pabaigoje, kai nebeturime nuotraukų sudaryti 3D modelį, yra įvykdomas globalus koregavimas.

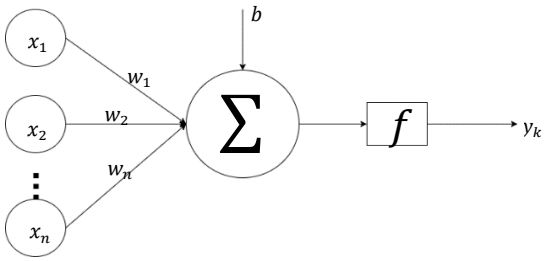
2. Neuroniniai tinklai

2.1. Dirbtiniai neuroniniai tinklai

Dirbtiniai neuroniniai tinklai (angl. artificial neural network - ANN)[31] – informacijos apdorojimo paradigma, kuria įkvepia biologinės nervų sistemos, kaip smegenų informacijos apdorojimas. Jį sudaro daugybė tarpusavyje sujungtų apdorojimo elementų (neuronų), kurie vieningai dirba sprendžia konkrečią problemą. Dirbtinis neuroninio tinklas (pav. 6) susideda iš 3 pagrindinių sluoksnių: įvesties sluoksnio, paslėpto sluoksnio ir išvedimo sluoksnio. Įvesties sluoksnyje laikoma duomenys, kurie bus naudojami apmokymui. Kiekvienas neuronas įvedimo sluoksnyje atstovauja unikalų atributą duomenų rinkinyje. Jokie skaičiavimai nevykdomai šiame sluoksnyje. Paslėptas sluoksnis yra tarp įvesties ir išvesties sluoksnių. Prieš kiekvieną rezultato perdavimą į kitą neurono sluoksnį yra įvykdomas aktyvacijos funkcijos. Išvedimo sluoksnyje gauna paskutinio paslėpto sluoksnio parametrus ir gražina modelio prognozę.



6 pav. Dirbtinio neuroninio tinklo pavyzdys



7 pav. Dirbtinio neurono pavyzdys

$$y = f(b + \sum_{i=1}^{n} x_i w_i) \tag{2.1}$$

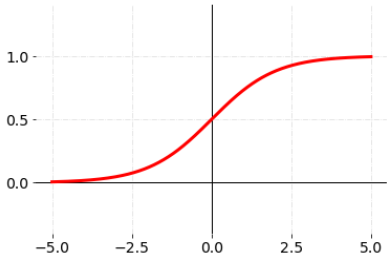
.

Neuroninis tinklas sudarytas neuronų. Iliustracijoje (pav. 7) pateikta dirbtinio neurono struktūros pavyzdys, kuriame gaunami trys elementai (x1,x2,x3), kurie gaunami iš apmokymo duomenų arba iš ankstesnio neuronų sluoksnio. Neuronas sudaugina elementus su jiems priskirtais svoriais, visus parametrus susumuoja ir po to yra vykdomas aktyvacijos funkcija, kurio metu pagal susumuotas įėjimo reikšmes yra įvykdoma funkcija, kurios metu parametrus transformuoja į išėjimo signalą ir perduoda į kitą neuronų sluoksnį. Svoriai (angl. weights) ir poslinkis (angl. bias) yra apmokami parametrai tinkle. Prieš neuroninio tinklo apmokymą yra užpildomi svoriais atsitiktiniais parametrais. Kai vykdomas apmokymas parametrai yra koreguojami link norimos parametrų ir teisingo išvesties. Svoris (angl. weight) kontroliuoja signalą (prisijungimo stiprumą) tarp neuronų. Kitais žodžiais, svoris nustato, kokią įtaką turės įvestis į išvedimo parametras. Jeigu svoris yra mažas (netoli 0), tada įvestis nepasikeis išvesties parametrą, tačiau didesnis svorio elementas tada žymiai bus pakeičiamas išvestis. Poslinkis (angl. bias), tai papildomas parametras, kuris naudojamas sureguliuoti išvedimą kartu su neurono išvesties suma. Be to poslinkis leidžia pastumti aktyvacijos funkciją į dešinę ar į kairę. Pagal ankstesnį sluoksnį poslinkio parametrai nėra įtakojami, tad jos vertė visados yra 1 arba kita konstanta. Be poslinkio, joks sluoksnis negalėtų sudaryti išvesties kitam sluoksniui, kad skirtųsi nuo 0, jeigu elemento reikšmė būtų 0.

2.1.1. Aktyvacijos funkcijos

Aktyvacijos funkcijos – tai matematiniai lygtys, kurios nustato neuroninių tinklų išvestį. Šios funkcijos yra prijungiamos prie kiekvieno neurono tinkle ir nustato kada turi būti de aktyvuotas ar aktyvuotas neuronas remiantis kaip svarbus neurono išvesti modelio prognozavime. Be to šios funkcijos padeda normalizuoti kiekvieną neurono išvesti tarp 1 ir 0 arba tarp -1 ir 1. Šios funkcijos turi būti labai našios ir padėtų sumažinti skaičiavimo laiką. Jeigu jų nepanaudosime tinklų sluoksniuose (2.1 formulėje atmetus f funkciją) išvestį gautume -∞ iki +∞, tad išvestis būtų transformuota į tiesinę funkciją. Tad sudarytas neuroninis tinklas negalėtų atpažinti sudėtingų raštų, kaip nuotraukas, video įrašai ir t.t. nes kiekvienas analizuojamas duomenų rinkinių elementai apibūdinami keliais matmenimis, kuriu negalime pavaizduoti su tiesinėmis transformacijomis. Tad yra naudojamas nelinijinės (angl. non-linear) funkcijos, kuriuo metu galima pateikti tikslesnė prognozes, naudoti sudėtingus duomenis. Pateiksime keletą populiariausių aktyvacijos funkcijų: Sigmoid (8 pav), Tanh (9 pav.), ReLU (10 pav.) ir Leaky ReLU (11 pav.).

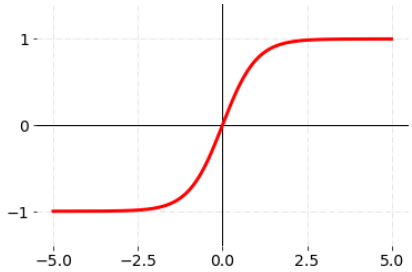
Sigmoid aktyvacijos funkcija (pav. 8 ir išraiška pateikta 2.2 punkte), kuriuo metu paima parametrą ir suspaudžia tarp 0 ir 1, kai dideli neigiami skaičiai tampa 0, o dideli pozityvūs skaičiai tampa 1. Ji geriausiai veikia, kai naudojama logistinėse regresijos modelis vykdyti dvejetainę klasifikaciją. Be to naudojamas išvesties sluoksnyje numatyti tikimybę. Tačiau toks intervalas pasunkina gradientų atnaujinimą. Ši funkcija turi nykstančio gradiento problemą, kuri kyla dėl didelio įvesties konvertavimo pradžioje 0 ir 1 pabaigoje . Nykstančio gradiento problema (angl. vanishing gradient problem) atsiranda, kai labai didelis or labai mažas įvesties regionai priskiriami labai mažam diapazonui, tad jeigu turime keleto sluoksnių tinklą, pirmajame sluoksnyje mes naudodami didelį įvesties regioną konvertuoajme į mažesnį išvesties regioną, kuris bus konvertuojamas į dar mažesnį regioną ir t.t. Tad didelis parametrų pakeitimas pirmajame sluosknyje, išvedime niekas nepakeis. Kita problema – išvestys nėra centruoti į nulį (angl. non zero centered problem). Kaip ir minėjome šios aktyvacijos išvestis yra tarp 0 ir 1, tai reiškia kad po sigmoid įvykdymo išvestis visada tampa teigiamas



$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.2}$$

8 pav. Sigmoid aktyvacijos funkcijos modelis

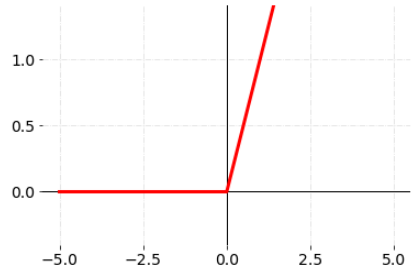
Tanh aktyvacijos funkcija (pav. 9 ir išraiška pateikta 2.3 punkte) labai panaši į sigmoid funkciją, tačiau modelis yra sucentruojamas tarp -1 ir 1. Tahn funkcija daugiausiai naudojama, kur reikia naudoti klasifikaciją tarp dviejų klasių, kaip ir sigmoid. Pagrindinis pranašumas tas, kad neigiamos įvestys bus pateiktos neigiamoje ašyje, o nulio įvestys – šalia nulio ašies Be to naudojant šią funkciją iškyla nykstančio gradiento (angl. vanishing gradient) problema.



$$f(x) = tanh(x) \tag{2.3}$$

9 pav. Tanh aktyvacijos funkcijos modelis

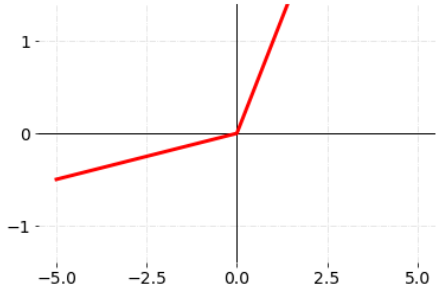
ReLU (angl. Rectified Linear units) aktyvacijos funkcija yra labai paprasta ir efektyvi, kuri naudojama beveik visose konvoliuciniuose neuroniniuose tinkluose (pav. 10 ir išraiška pateikta 2.4 punkte). Ši funkcija išvengia ir pašalina nykstančio gradiento problemą. Naudojant Sigmoid ar Tahn funkcijas turint didelį neuroninį tinklą, kuriame daug neuronų, apmokymas tampa labai ilgas, nes bus įvykdomos visos neuronų aktyvacijos apibūdinti tinklo išvestį. Tad šios funkcijos privalumas – de aktyvuoti keletą neuronų sudarant daug lengvesnį neuronų tinklą. Be to naudoja daug paprastesnę matematinę operaciją negu Sigmoid ar Tahn. Tačiau ši aktyvacijos funkcija turi ir problemų, kuri dar vadinama “dying ReLU”. Ši problema atsiranda, kai neigiamus įvestis paverčia nuliais. Ši problema atsiranda, kai mokymosi lygis yra per didelis arba kai yra didelis neigiamas poslinkis (angl. bias).



$$f(x) = max(x, 0) \tag{2.4}$$

10 pav. ReLU aktyvacijos funkcijos modelis

Leaky ReLU aktyvacijos funkcija (pav.11 ir išraiška pateikta 2.5 punkte) yra modifikuota ReLU funkciją, kurioje išsprendžiama “dying ReLU” problema. ReLU grąžina 0, jeigu įvestis yra neigiama, tad neuronas tampa neaktyvus ir neprisideda prie gradiento srauto. Ši funkcija išsprendžia šia problemą, leisdami labai mažą kiekį (šalia nulio) neigiamus parametrus, taip išspręsdami ReLU problemą, tad išvengia negyvų neuronų susidarymo problemą. Jeigu apmokymo metu ReLU silpnai mokinasi, galima naudoti šią aktyvacijos funkciją.



$$f(x) = max(0.1x, 0) \tag{2.5}$$

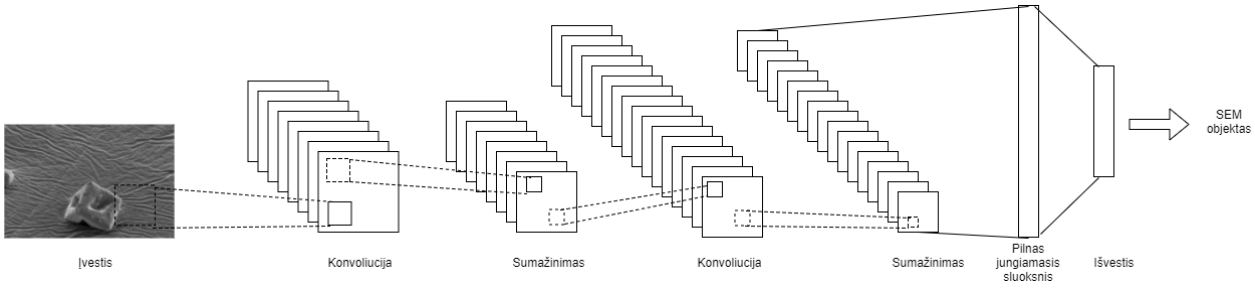
11 pav. Leaky ReLU aktyvacijos funkcijos modelis

2.2. Konvoliuciniai neuroniniai tinklai

Konvoliuciniai neuroniniai tinklai yra gilieji dirbtiniai neuroniniai tinklai, kurie naudojami klasifikuoti nuotraukas (kas toje nuotraukoje atvaizduojama), surikiuoti nuotraukas pagal tam tikras klasifikacijas, aptikti nuotraukoje tam tikrą objektą, kaip gatvės ženklus, žmogų, mašiną, smegenų auglį ir t.t. Šie tinklai gana panašūs į praeitame poskyryje aptartus neuroninius tinklus. Tačiau konvoliuciniai neuroniniai tinklai turi ir savas savybes:

- Pirmiausiai sluoksniai yra trijų dimensijų: pločio, aukščio ir gylio.
- Antra, neuronai vienoje sluoksnyje nesujungti su kitais neuronai kitame sluoksnyje.
- Trečia, kad paskutinis išvedamas sluoksnis yra sudaromas, kaip vieno vektoriaus ilgio nurodant gylio dimensiją, kuriame pateikti tam tikros klases tikimybių rezultatai.

Pateiktoje iliustracijoje (pav. 12) pavaizduojame konvoliucinio neuroninio tinklo struktūros pavyzdį. Kiekvieno sluoksnio dalies struktūrą mes aptarsime kitose poskyriuose.

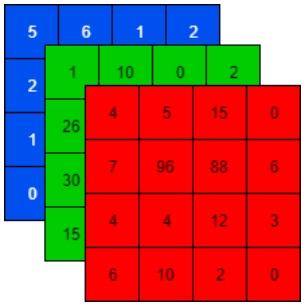


12 pav. Konvoliucinio neuroninio tinklo struktūros pavyzdys

2.2.1. Struktūra

Įvestis:

Konvoliuciniai neuroniniai tinklai visados savo įvestyje naudojama tam tikro tipo nuotraukas. Kiekvieną nuotrauką galime atvaizduoti, kaip pikselių reikšmių matricą. Daugiausiai naudojama 8 bitų dydžio pikseliai, tada yra labai lengvai atvaizduoti parametrus, kurie nurodomi [0,255] rėžiu. Kiekvienos nuotraukos turi spalvų kanalų (angl. channel) sistemą. Pavyzdžiui jeigu tam tikra nuotrauka, kuri nufotografuota fotoaparatu turi raudonos, žalios, mėlynos spalvos (angl. rgb) sistemos struktūrą, tada yra sukuriama matricų struktūrą, kur kiekvienas spalvos sluoksnis yra atskirtas matricomis. Pavyzdžiui turime nuotrauką 4x4 pikselių, tada turėsime 3 matricas, kurios susijusios su kiekvienos nuotraukos spalvų kanalu. Iliustracijoje (pav. 13) pateiktos pavyzdinės nuotraukos matricų struktūra.



13 pav. 4x4 dydžio nuotraukų matricos, kuri turi 3 spalvų kanalus

Konvoliucinis sluoksnis:

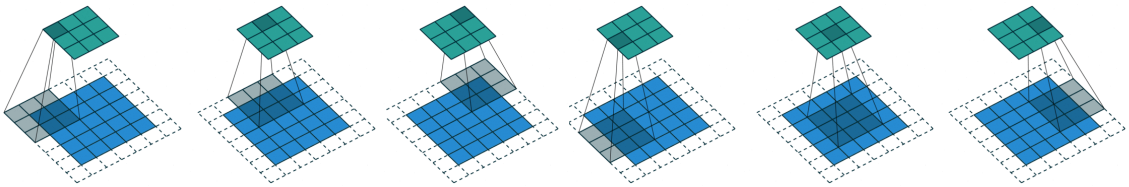
Šio sluoksnio paskirtis - išgauti ko daugiau ypatybių (angl. features) iš nuotraukų. Ypatybė (angl. feature), tai yra tam tikras naudingas modelis, su kuriuo galima daryti įvairias nuotraukos analizes. Pavyzdžiui žmogaus aptikimo metu mes ieškome žmogaus silueto, veido, akių, kurie bus panaudojami kaip ypatybės, kurios aptiks ir išmokys skirtingi konvoliuciniai sluoksniai. Rasti ypatybes naudojama filtrai (dar kitaip vadinami branduoliai angl. kernels). Filtrai transformuoja pikselių informaciją išgauti ypatybes. Filtrus galima naudoti įvairaus dydžio, tačiau daugiausiai naudojama mažo dydžio filtrai, su kuriais gausime daug lokalios informacijos, o naudojant didesnius filtrus surinksime globalią informaciją. Vykdant slinkimo veiksmą, kuriuo metu filtras slenka per įvesties matricą ir suskaičiuoja ypatybių žemėlapį (angl. feature map). Konvoliucinio sluoksnio parametrų dydis priklauso nuo keleto hiperparametrų:

- Filtro dydžio (aukščio F_H ir pločio F_w) parametrų.
- Filtrų kiekio (K) – šis elementas nusako, kiek ypatybių žemėlapių bus sukurta.
- Žingsnis (S) - šis elementas nusako, per kiek matricos pikselių bus pastumtas filtras. Jeigu žingsnio elementas lygus 2, tai filtras peršoks 2 pikselius kiekvieną kartą kol nepastums visos įvesties matricos.
- Nulių užpildymas (P) – šio elemento paskirtis užpildyti nulius aplink įvestą matricos kraštus. Šio užpildymo esmė - galėtume panaudoti filtrus nuotraukos matricos riboje esantiems elementams rasti.

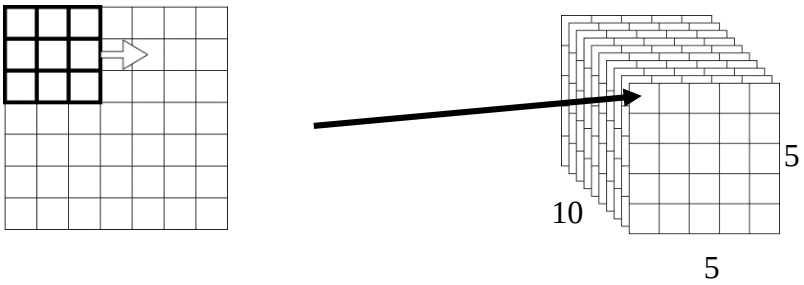
Apskaičiuoti koks bus ypatybių konvoliucnio sluoksnis dydis, kai panaudosime įvestį kurio dimensijos [W (plotis) x H (aukštis) x D (gylis)], rezultato dimensijos bus [$C_H \; x \; C_w \; x \; K$], kur

$$C_H = \frac{H - F_H + 2P}{S} + 1 \qquad C_w = \frac{W - F_w + 2P}{S} + 1 \qquad \textbf{(2.5)}$$

Pateikiame pavyzdį: turime 7x7 įvesties matricą ir 3x3 filtrą su 1 žingsniu be nulių užpildymo, tai rezultatas bus 5x5 matrica. Pateiktame iliustracijoje (pav. 14) pavaizduotas modelis, kuriame naudojama 10 filtrų.



14 pav. Konvoliucinio sluoksnio skilties sudarymas, kuriame naudojama žingsniai (angl. stride) ir nulių užpildymas (angl. padding). [19]



15 pav. Konvoliucinis sluoksnio pavyzdys, kur naudojama 7x7 įvesties matrica (kairėje pusėje) ir 3x3 filtras (viduje – patamsinta matrica) su 1 žingsnio judėjimu ir be nulių užpildymo

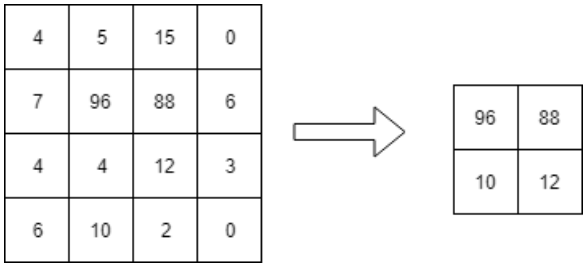
Parametrų dalijimasis tarp konvoliucinių sluoksnių yra labai svarbus etapas. Be parametrų dalijimosi išmokinant neuroninį tinklą reikalautų labai daug parametrų. Šią problemą išspręsti reikia gauti tam tikrame regione svarbią ypatybę (angl. feature), tada būtų labai naudinga ir kitose konvoliucinio sluoksnio skilčių regionuose panaudoti, tad nereikėtų apmokyti tuos pačias ypatybes daugelį kartų. Pavyzdžiui pateiktoje iliustracijoje (pav.) turime (3*3*3+1)*(5*5*10) = 7000 , kur poslinkio elementas lygus 1. Panaudojus parametrų dalijimosi būdą, tada gausime (3*3*3+1)*10 = 280 parametrų.

Sumažinimo sluoksnis:

Kitas svarbus sluoksnis – sumažinimo sluoksnis, kuris yra padedamas po konvoliucinio sluoksnio. Šis sluoksnis naudojamas sumažinti įvesties (sluoksnio) dimensiją (aukštį, plotį), tačiau nepaveikia gylio dimensijos ir veikia atskirai tarp kiekvieno konvoliucinio sluoksnio skilčių. Jis padeda sumažinti skaičiavimo parametrus ir sumažina triukšmą (angl. over-fitting) duomenyse (šią problemą aptarsime kitame poskyryje). Egzistuoja keletas operacijų, su kuriomis galima įvykdyti šį metodą: MaxPooling (maksimalios reikšmė radimas, kuris yra daugiausiai naudojamas neuroniniams tinklams kurti), SumPooling (sumos operacija) ir MeanPooling (vidurkio operacija). Kaip ir konvoliucinio sluoksnio sudaryme naudojamas tam tikro dydžio filtras, kuris slenka per įvesties matricą, taip sudarydami naują matricą. Pateiktoje iliustracijoje (pav. 16) pavaizduosime, kaip veikia sumažinimo sluoksnio metodas naudojant maksimalios reikšmės operaciją (angl. max pooling). Šis sluoksnis priklauso nuo kelių hiperparametrų: filtro dimensijos (F) ir žingsnių (S). Apskaičiuoti koks bus ypatybių konvoliucinio sluoksnio dimensija po sumažinimo, kai panaudosime įvestį kurio dimensijos [W (plotis) x H (aukštis) x D (gylis)], rezultato dimensijos bus [$C_H \; x \; C_w \; x \; D$], kur

$$C_H = \frac{H - F_H}{S} + 1 \qquad C_w = \frac{W - F_w}{S} + 1 \qquad \textbf{(2.6)}$$

Daugiausiai naudojama filtras, kurio dydis yra 2x2 su 2 žingsnių parametru. Naudojant didesnius filtrus gali sunaikinti informaciją.

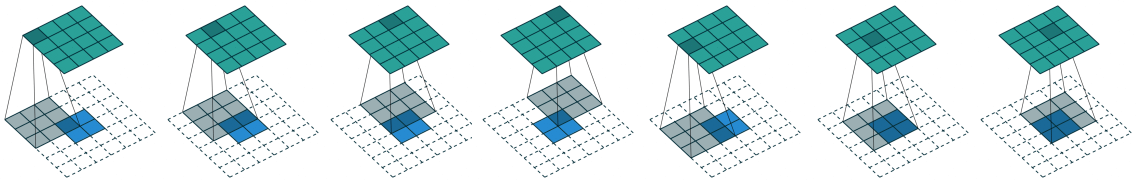


16 pav. Sumažinimo sluoksnio rezultatas, kuriame naudojama 2x2 filtras ir 2 žingsnių peršokimo parametras

Padidinimo sluoksnis:

Šis sluoksnis yra naudojamas ypatybių žemėlapio padidinimui. Šis sluoksnis yra plačiai naudojamas semantiniam segmentavimo būdui įvykdyti (plačiau apie šį metodą aptarsime kitame skyriuje). Įvykdyti šį sluoksnį reikia pasirinkti iš kelių metodų, kaip padidinti žemėlapio matricą: Arčiausio kaimyno interpoliacija, Bi-linijinė interpoliacija, Bi-kubinė interpoliacija, MaxUnpooling būdai [19]. Tačiau daugiausiai yra naudojami perkeltiniai konvoliuciniai (angl. transposed convolution) metodai [19]. Šis metodas nenaudoja iš anksto sudarytos interpoliacijos. Šis būdas yra labai panašus į konvoliucinio sluoksnio sudarymą – naudoja filtrus ir žingsnių parametrą, tačiau be nulių užpildymo. Iliustracijoje (pav. 17) pateiktas pavyzdys, kaip vykdomas

perkeltinis konvoliucinis metodas, kuriame naudojamas žingsnis, kurio parametras – 1, nulių užpildymą nenaudojame.



17 pav. Perkėlimo sluoksnis, kuriame žingsniai - 1 (angl. stride), nulių užpildomo nenaudojame (angl. padding). [19]

DropOut sluoksnis:

Šio sluoksnio paskirtis modifikuoti patį tinklą. Šis būdas naudoja tikimybės parametrą p, kuriuo jis po kiekvieno partijos (angl. batch) įvykdymo, neurono apmokymo metu atsitiktinai yra pašalinami su tikimybe 1-p mazgai, o su tikimybe p yra nepašalinami atsitiktiniai mazgai. Be to šis sluoksnis neleidžia, kai kuriems neuronams dominuoti visą apmokymo metu, kuriuose gali būti triukšmo. Kai testuojame neuroninį tinklą šis sluoksnis nėra vykdomas.

Pilnas jungiamasis sluoksnis:

Būdas, kai neuronai iš vieno sluoksnio yra sujungiami su kitais neuronais kitame tinklo sluoksnyje - pvz. pavaizduotas pav. kur antrajame ir trečiajame paslėptuose sluoksniuose pilnai sujungiami neuronai.

2.3. Neuroninio tinklo apmokymas

Neuroninio tinklo apmokymo tikslas sumažinti skirtumą tarp neuroninio tinklo išvesties ir tikrųjų duomenų. Be to modelis turi gauti detalius rezultatus kai naudojami duomenys, kurie nebuvo panaudojami tinklo apmokyme. Apmokyti neuroninį tinklą galime keliais būdais:

- Prižiūrimas mokymas (angl. supervised) – apmokamas modelis naudojant pažymėtus duomenų rinkinius. Mes perskiriame savo duomenis į mokymo duomenų rinkinį ir testavimo duomenų rinkinį, kur apmokymo rinkinį naudosime apmokyti neuroninį tinklą, o kitas veikia kaip nauji (nematyti) duomenys prognozuoti rezultatus apmokymo metu.
- Neprižiūrimas mokymas (angl. unsupervised) - metodas, kuriuo metu naudoja duomenis, kurie nėra pažymėti ar klasifikuoti. Tai reiškia, kad nėra jokių mokymo duomenų rinkinių, tad pats modelis turi sustruktūrizuoti ir rasti paslėptas struktūras, panašumus, raštus suskirstyti į tam tikras kategorijas.
- Beveik prižiūrimas mokymas (angl. semi supervised) - hibridinis metodas (prižiūrimo ir neprižiūrimo mokymosi metodų). Šis būdas remiasi nedideliu kiekiu pažymėtų duomenų ir didelio kiekio nepažymėtų duomenų apmokyti neuroninį tinklą. Pažymėti duomenys naudojami apmokyti modelį ir toliau iš dalies apmokytas modelis yra naudojamas pažymėti nepažymėtus duomenis. Gautus rezultatus yra naudojami apmokyti tinklą kartu su pažymėtais duomenimis.
- Mokymo pastiprinimu mokymas (angl. reinforcement) – metodas, kuriuo metu yra apmokamas agentas, kaip pasirinkti patį našiausią veiksmą iš jo žinomų veiksmų tam tikroje aplinkoje, kad gautume maksimalų rezultatą.

Epocha ir partijos dydis

Šie parametrai nustato, kokiu dažniu yra tiekiami apmokymo duomenys yra perduodami apmokymui. Partijos dydis (angl. batch), tai hiperparametras, kuris nustato kiek bus naudojama duomenų apmokyti neuroninį tinklą prieš atnaujinant vidinius modelio parametrus. Epocha (angl. epoch), tai hiperparametras, kuris nusako kiek kartų neuroninis tinklas bus apmokytas naudojant duomenų rinkinį.

Partijos normalizavimas

Partijos normalizavimas (angl. batch normalization) - metodas normalizuoti duotą aktyvaciją prieš perduodant į kitą neuroninio tinklo sluoksnį. Šis metodas padeda apmokyti greičiau ir tiksliau tinklą. Šis metodas išsprendžia vieną problemą - vidinį nepriklausomą poslinkį (angl. internal covariate shift). Pasikeitus įvesties pasiskirstymui paslėpti sluoksniai bando išmokti prisitaikyti prie naujo įvesties pasiskirstymo, tad sulėtina neuronų tinklo apmokymą. Ši problema atsiranda, kai įvesties pasiskirstymas smarkiai skiriasi nuo anksčiau matyto įvesties.

Modelio optimizavimas

Neuroninių tinklų apmokyme yra naudojama tam tikri optimizavimo algoritmai, kurie atnaujina tinklų atributus, kaip svorius, mokymosi greitį, kad būtų galima sumažinti nuostolių funkciją. Pagrindinis optimizavimo metodas: gradientų nusileidimas (angl. gradient descent). Šis metodas paskaičiuoja mažus pakeitimus kiekvienam svoriui ir nustato kaip jie įtakoja nuostolių funkcijai (angl. loss function). Po to pakeičia kiekvieną svorį individualiai pagal gradientą. Keičiant svorius per greitai gali kliudyti minimizuoti nuostolius, tad yra pateikiamas parametras mokymosi greitis (angl. learning rate), kuris naudojamas sudauginti su gradiento parametru, kad užtikrintų visų svorių pakeitimai būtų maži. Iš šio optimizavimo metodo buvo sudaryti daugiau šio tipo metodų: SGD, Adam, Adagrad, RMSprop ir t.t

Priekinis sklidimas

Priekinis sklidimas (angl. forward progation) – procesas, kuris tiekia įvesties vertes į neuroninį tinklą ir gaunama išvestis, kurį vadinsime prognoze. Pirmiausiai mes įvedame įvesties parametrus į neuroninio tinklo pirmąjį sluoksnį, kuriame nevykdomi jokios operacijos. Sekančiame sluoksnyje paimamos vertės iš praeito sluoksnio ir įvykdomas apskaičiavimai ir panaudojus aktyvacijos funkcijas perduodami vertės į kitą neuroninio tinklo sluoksnį. Taip skaičiavimai kartojasi, kol pasieksime paskutinį sluoksnį, kur gauname prognozę (rezultatą). Šį procesą vykdome apmokymo metu, taip pat ai vykdome prognozavimą.

Atgalinis sklidimas

Atgalinis sklidimas (angl. back-propagation) – apmokyti neuronų tinklą kiekvienos iteracijos metu pirmiausiai yra vykdomas priekinis sklidimas, kuris perduoda įvestis iš vieno sluoksnio į kitą. Po to yra įvykdomas ši procesas. Pirmiausiai apskaičiuojamas gradientai kuriuo metu bando atnaujinti svorius ir poslinkius pagal gradientų algoritmus. Skaičiavimas vyksta nuo paskutinio atgal iki pirmojo sluoksnio.

Nuostolio funkcijos

Nuostolio funkcijos (angl. loss function)- metodas, kuris vertina, kaip gerai sudarytas neuroninis tinklas naudojant pateiktą duomenų rinkinį. Jeigu prognozė toli nukrypusi, tai nuostolio funkcija išves aukštą rezultatą, jeigu prognozė gera pateiks mažesnę vertę.

Nepakankamumas

Nepakankamumas (angl. underfittng) nurodo į modelį, kuris negali modeliuoti mokymo duomenų ir generalizuoti (angl. generalize) naujus duomenų rinkinių (kai neuroninis tinklas veikia gerai ant neapmokytų duomenų, galime sakyti, kad gerai apibendrina - generalizuoja naudojant duomenų rinkinį modelio apmokyme).

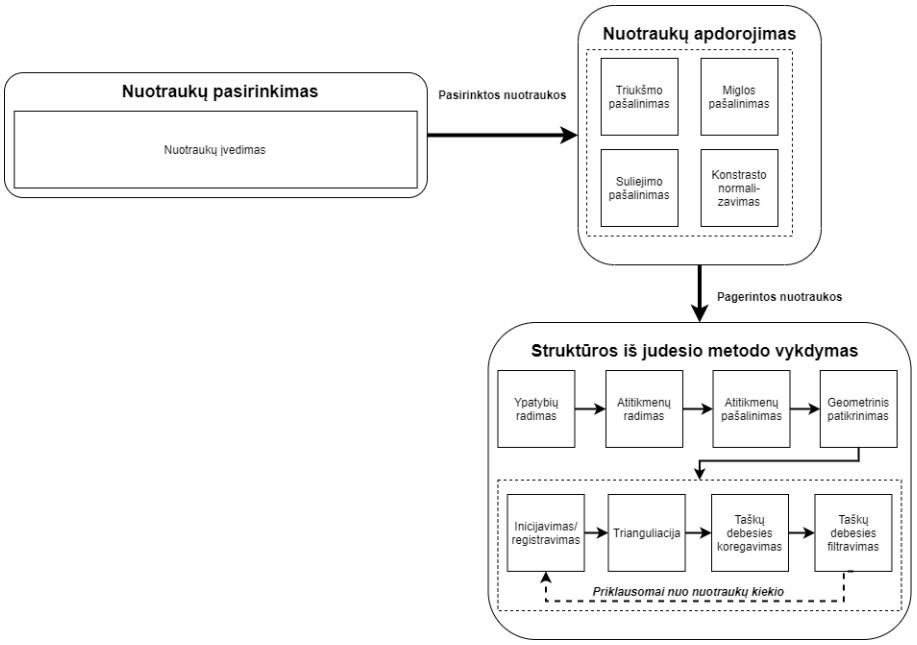
Perteklius

Perteklius(angl. overfitting) atsiranda, kai vykdomas neuroninio tinklo modelio apmokymas naudojant duomenis, kurie per daug gerai išmoksta, tačiau kai vykdomas testavimas - sužlunga, kai pateikiami nauji duomenys iš tos pačios srities ir negali generalizuotis (angl. generalize). Viena iš priežasčių tinklas yra per daug sudėtingas, nes gali išmokti nereikalingą detalumą, triukšmą duomenyse, taip kad neigiamai paveikia modelio našumą naujiems duomenims.
Pertekliaus problemos pašalinimo būdai:

- Modelio supaprastinimas, pašalinant sluoksnius ar sumažinti neuronų kiekį, taip kad neuroninis tinklas būtų mažesnis.
- Ankstus sustabdymo metodas, kuriuo metu sustabdo apmokymą, kai nepagerėja nuostolių funkcija pagal neuroninio tinklo patvirtinimo (angl. validation) duomenis ir net jeigu dar liko vykdyti epochų.
- Duomenų augmentavimas. Naudodami nuotraukų apdorojimo būdus, kaip vartymas, sukimas, mastelio keitimas, ryškumo keitimas, triukšmo pridėjimas ir t.t. Padidindami apmokamų duomenų kiekį galime sumažinti perteklių, nes modelis nebegalės perpildyti visų pateiktų įvesčių ir priverstas apibendrinti.
- Naudoti reguliavimus. Reguliavimas, tai sprendimas sumažinti modelio sudėtingumą pridedant tam tikrą sankciją nuostolio funkcijai. Pagrindinės sankcijos yra žinomos, kaip L1, L2, dropout reguliavimai. L2 skatina svorio vertes pasiekti nulį, o L1 skatina svorio vertes būti nulinėmis. Naudojant „dropout“ sluoksnį (reguliavimą). Šis būdas apmokymo metu modifikuoja neuroninį tinklą, kai kiekvienos iteracijos metu, atsitiktinai atmeta neuronus pagal tikimybę P, taip paprastindami neuroninį tinklą. Neuronų deaktyvacija yra taikoma, kai vykdomas priekinis sklidimo procesas ir svorių atnaujinimas.

3. Apibendrinimas

Prieš tyrimo aprašymą, šiame skyriuje pateiksime šio darbo architektūros modelį, kuri siekia atsakyti, kaip mes sudarysime tikslų trimatį. Trumpai aprašysime kiekvieną architektūros dalį ir pateiksime tikslą ir rezultatą. Be to paanalizuosime naudojamus duomenis.



18 pav. Modelio struktūra

Modelis

Pateiktame paveikslėlyje (pav. 18) pavaizduotas trimačio objekto sukūrimo architektūros modelis. Modelis susidaro iš 3 dalių: **nuotraukų pasirinkimas**, **nuotraukų apdorojimas**, **struktūros iš judesio metodo vykdymas**,. Tad trumpai aprašykime apie kiekvieną dalį.

Nuotraukų pasirinkimas

Trimačio modelio tikslumas labai priklauso nuo pasirinktų nuotraukų kiekio (kaip nuotraukos išdėstytos trimatėje erdvėje) ir jų kokybės. Šioje dalyje nevykdysime jokių sprendimų, kurie atrinktų geras nuotraukas iš sąrašo, tačiau ateinančiuose dalyse vykdysime tam tikrus nuotraukų apdorojimo, filtravimo ir porų pasirinkimo būdus.

Nuotraukų apdorojimas

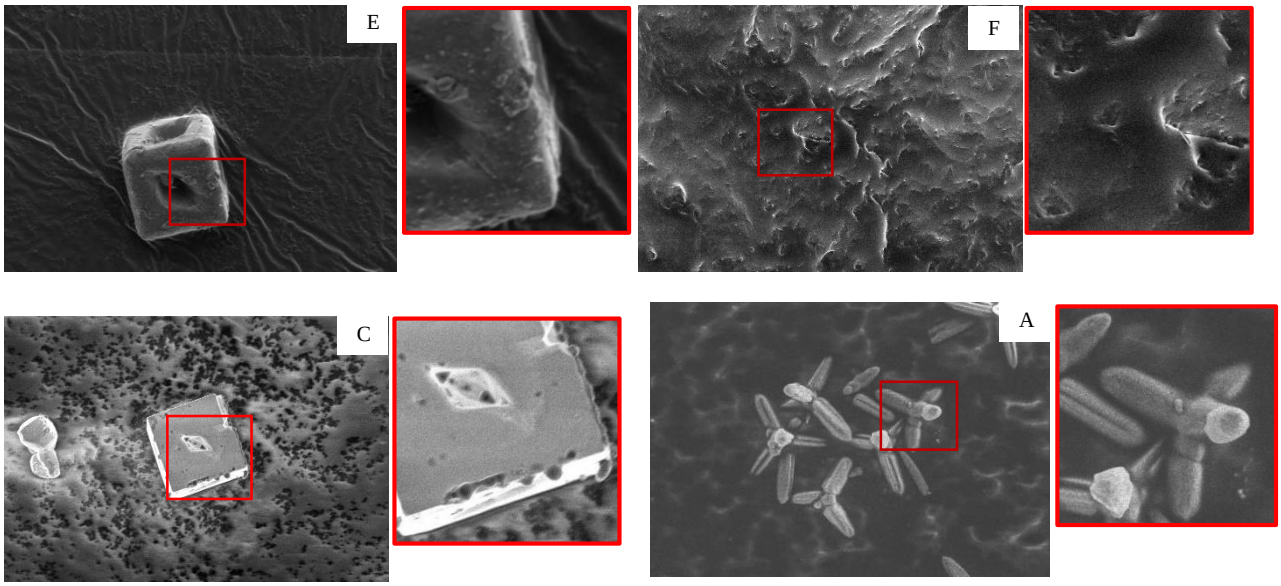
Gaunamos iš skenavimo mikroskopo ne visados gauname idealias SEM nuotraukas. Šioje dalyje įvykdysime nuotraukų apdorojimo žingsnius, kaip triukšmo pašalinimo, kontrasto normalizavimo ir miglos pašalinimo algoritmus. Taikant šiuos algoritmus nustatysime, kaip efektyviai paveikia trimačio modelio tikslumą.

Struktūra iš judesio būdas

Pagrindinė šio dalies rezultatas - gauti tikslius nuotraukų išsidėstymą trimatėje erdvėje parametrus (išoriniai) (plačiau aptarėme 1 skyriuje) ir kameros parametrus (vidiniai). Yra nemažai būdų, kaip gauti rezultatą, tad didžioji dalis skyrelio bus išanalizuoti egzistuojančius algoritmus ir įvertinti, kaip veikia naudojant apdorotas ir neapdorotas SEM nuotraukas.

3.1. Naudojamų duomenų analizavimas

Pirmiausiai pateiksime, kokius duomenis mes naudosime eksperimentų testavimui ir vertinimui. Šiame darbe mes naudosime keletą skirtingų SEM nuotraukų duomenų rinkinių (pav.19), kurie gauti iš Vilniaus Universiteto chemijos fakulteto duomenų bazės. Prie kiekvienos SEM nuotraukos yra pateikta, kaip buvo pasuktas mėginys pagal x, y, z rotacijos parametrus (**Papildymas A** skyrelio lentelėje), kuriuos naudosime pagrįsti, kaip sudarytas struktūros iš judesio (angl. structure from motion) sprendimas vykdymo metu, gauna tikslius reliatyvius kameros parametrus ir kaip tiksliai galime surekonstruoti trimatį modelį.



19 pav. Pateiktos SEM nuotraukos, kuriuose pateikta triukšmo (F),apšvietimo (C), miglos (E) ir mažo kontrasto (A) problemas

4. Nuotraukų pagerinimo sprendimas

Šioje dalyje pirmiausiai pateiksime, kokios problemos SEM nuotraukose galime iššvelgti. Po to pateiksime sprendimus, kaip būtų galima modifikuoti nuotraukas, taip kad gautume geresnius atitikmenis tarp nuotraukos porų. Kiekvienoje dalyje išsamiai aprašysime pateiktą sprendimą ir pateiksime testavimo rezultatus ir palyginsime su egzistuojančiais sprendimais.

4.1. Problema

Pagrindinė trimačio modelio rekonstrukcijos problema – įvesties duomenų kokybė. Vykdant ypatybių taškų (angl. feature point) aptikimą ir deskriptoriaus (angl. descriptor) (tam tikras nuotraukos lopinėlis aplink ypatybių tašką) sudarymą, kurie naudojami atitikmenims tarp nuotraukų porų rasti, jeigu gauname labai skirtingus deskriptorius ir nors atitikmenų gavimo algoritmai labai pritaikyti įvairiems pokyčiams atpažinti, vis tiek gauname daug nereikalingų ir blogų atitikmenų. Mūsų darbe naudojamos skenavimo elektronų nuotraukos ne visados gaunamos idealios kokybės. Dėl atsispindėjusių spindulių nuo tiriamojo objekto paviršiaus iš skenavimo aparato ir kitų priežasčių (aptarėme 1 skyrelyje) nuotraukose susidaro skirtingo ypatybes: kontrastas, migla, suliejimas, netolygūs šviesių ir tamsių taškų plotai ir artefaktai. Be to dauguma analizuojamų egzempliorių turi pakartojančius nuotraukų pikselių plotus, kuriuose nėra jokių išsiskiriančių ypatybių (angl. feature point). Paveikslėliuose (pav.) pateikti problemų pavyzdžiai, su kuriais susiduriami didžioji dalis SEM nuotraukų:

- (A) paveikslėlyje pateikta kontrasto ir miglos problemos pavyzdys. Pateiktoje nuotraukoje matome, kad paviršius mikroskopiniai iškilimai sunkiai matomi.
- (E) paveikslėlyje pateikta suliejimo/be fokuso (angl. out of focus) pavyzdys. Nuotraukoje matome labai neryškūs analizuojamos objekto paviršiaus kraštinės. Be to yra gana nemaža dalis ant objekto šviesių ir tamsių plotų.
- (F) paveikslėlyje pateikta triukšmo pavyzdys. Nuotraukoje pateikta druskos ir pipiro triukšmai.
- (C) paveikslėlyje pateikta šviesių plotų problemos pavyzdys. Ant analizuojamo objekto didžioji dalis paviršiaus yra gana šviesi ir pats paviršius yra gana lygus taip sumažina gerų ypatybių taškų gavimą.

Sprendimas

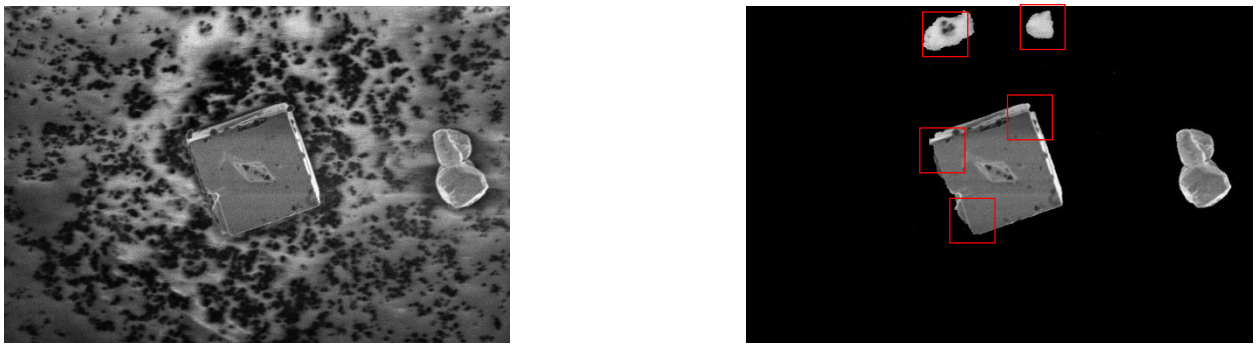
Išspręsti šią problemą mes ištestuosime du sudarytus sprendimus, kurie padėtų gauti geresnius ypatybių atitikmenis tarp nuotraukų porų:

- nuotraukų iškirpimas - mes naudosime semantinį segmentavimo būdą, kuriame aptiksime objektą pikselio tikslumu ir pašalinsime SEM nuotraukų fonus.
- nuotraukų apdorojimo pagerinimas - mes sudarysime sprendimą, kuriame pašaliname triukšmą, pageriname kontrastą ir kitas ypatybes.

4.2. Objektų aptikimas

Trimačių modelių kūrimui naudojant SEM nuotraukas, daugiausiai dėmesio skiriama analizuojamam objektui, o ne fonui. Įvykdžius bendrų taškų gavimą gauname daug nereikalingų taškų, kurie susieti su fonu arba su analizuojamo objekto šešėliais negu su pačiu analizuojamu objektu. Tokie gauti taškai sukelia labai daug problemų: daugiau reikalauja resursų apdoroti duomenis, kurie po to yra nereikalingi objekto analizavimui. Taip pat gali atsitikti, kad trimatis modelis gali būti atvaizduojamas netiksliai dėl blogai atitinkančių taškų kiekio rastų tarp nuotraukų porų.

Nereikalingų bendrų taškų pašalinimui galima išspręsti panaudojus tam tikrus objekto radimo algoritmus, kaip GrabCut [48]. Šis reikalauja vartotojo įvesties – apibrėžti stačiakampį apie objektą. Už stačiakampio yra nustatoma, kad tai yra fonas, o vidiniai stačiakampyje esantys pikseliai yra nežinomi. Pagal pažymėtą yra nustatomas pikselių spalvų modelis. Šis algoritmas naudoja grafų struktūrą, kuriame kiekvienas mazgas yra nuotraukos pikselis. Be to yra pridedami du mazgai: šaltinio mazgas ir fono mazgas. Šie mazgai jungia skirtingus duomenis: fono mazgas – fono pikselius, o šaltinio mazgas- ieškomą objektą. Nustatyti, kaip būtų gali atskirti į dvi dalis, tai yra vykdomas šalia esančių pikselių palyginimas ir kraštų informaciją, iš kurių gauname svorius. Jeigu yra didžiulis pikselio spalvos skirtumas tarp dviejų šalia esančių taškų, tai gauname labai mažą svorį, jeigu ne gauname didesnį svorį. Taip kiekvieną pikselį apskaičiavę užpildome grafą svoriais ir po to yra vykdomas grafo atskyrimas į dvi dalis pagal svorių sumą šaltinio mazgą ir fono mazgą. Po atskyrimo visi pikseliai, kurie yra šaltinio mazge gauname analizuojamą objektą, o fono mazge – foną. Panaudojus SEM nuotraukas pratestuoti šį algoritmą, automatiškai be vartotojo pagalbos. Nustatėme, kad kiekvienos SEM nuotraukų objektas yra centre. Tačiau patestavus rezultatai tikslūs nesigavo. Iliustracijoje (pav. 20). pavaizduotos vietos kur stipriai apkarpomos kraštinės, kurios yra labai svarbios atitikmenų gavimui. Be to fonas ir analizuojamas objektas turi panašias spalvas, tai būtų gana sudėtinga tiksliai atrasti gerą rezultatą. Šią problemą galime išspręsti panaudodami semantinį segmentavimo būdą, kuris naudoja gilųjį neuroninį tinklo apmokymą (angl. deep learning).



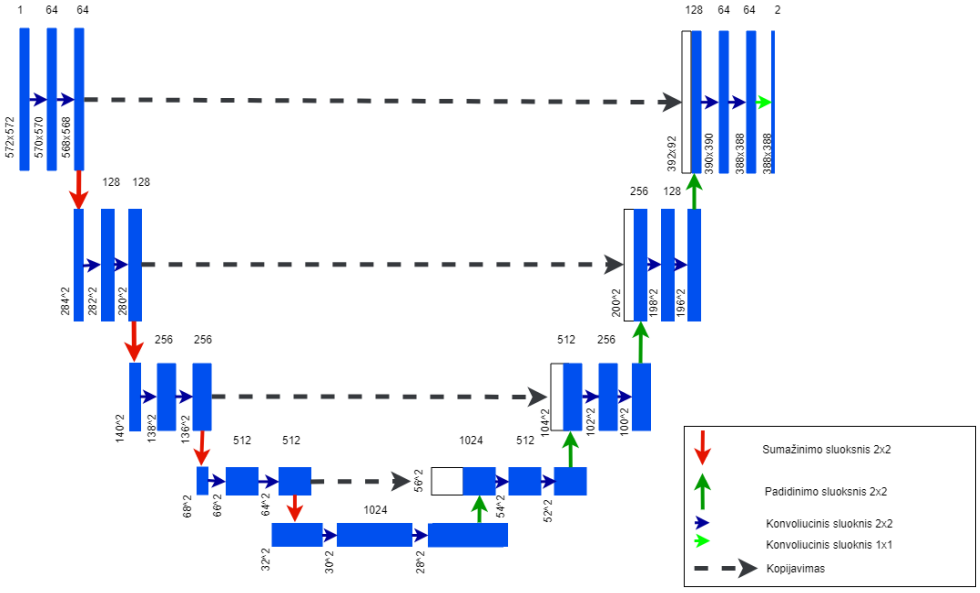
20 pav. Objekto aptikimas naudojant GraphCut algoritmą rezultatas. Raudoni stačiakampiai reiškia vietas, kuriuose blogai aptinka kraštines (nevisi objekto pikseliai gauti)

4.3. Semantinis segmentavimas

Semantinis segmentavimas, tai vienas iš objekto aptikimo būdų, kuriuo metu kiekvienos nuotraukos pikseliai priskiriama tam tikrai klasei. Šį būdą labai plačiai naudojama objektų patikimui, kaip veido aptikimui (akių rainelės aptikimas, nosies, ausų, pačios veido formos

aptikimas ir t.t.), pėsčiųjų aptikimui, kuris naudojamas sekimo kameroms, aptikti objektus iš satelito nuotraukų. Be to yra labai plačiai naudojamas medicinoje tikslesniems ir greitesniems aptikimams negu aptikti objektą tam tikros specialybės daktarui (aptikti auglį ar kitą patologiją, išmatuoti tam tikrų audinių, vietų plotus ir t.t.). Šiame darbe mes naudosime vieną iš populiariausių semantinio segmentavimo neuroninio tinklo architektūrų - U-Net [47]. Šis modelis yra plačiai naudojamas medicinoje, nes galime tiksliai apmokyti tinklą naudojant ganėtinai mažo dydžio nuotraukų sąrašą. Be to yra plačiai naudojama žmogaus pozos apskaičiavimui, objektų aptikimui ir be to semantiniam segmentavimui. Jeigu duomenų nepakanka galima įvykdyti nuotraukų augmentavimo būdą, kuriuo metu sugeneruojame didelius kiekius modifikuotų nuotraukų naudojant kompiuterinės regos algoritmus, kaip triukšmo uždėjimas, pašalinimas, nuotraukos pasukimas įvairiais kampais ir t.t.

Kaip ir kiekvienoje konvoliucinio neuroninio tinklo architektūroje naudojame konvoliucinio sluoksnio sudarymą, ypatybių žemėlapio sumažinimą, taip padidindami ypatybių žemėlapio kiekį kiekviename sluoksnyje. Tačiau ši struktūra panaši į U formą. Iliustracijoje (pav. 21) pateiktas šio modelio struktūra. Ši architektūra susideda iš užkoduotojo ir dekoduotojo tinklų. Pirmiausiai yra vykdomas užkoduotojo dalis, kuriame mažinamas ypatybių žemėlapis rasti aukšto lygio informaciją. Kiekviename sluoksnyje gautus ypatybių žemėlapius (angl. feature maps) nukopijuojame, kuriuos po to panaudosime nuotraukos atkūrimui. Kita dalis, kurią vykdysime - dekodavimas. Dekoduotojo paskirti atkurti nuotraukos detales, kuriuose atvaizduojami klasifikuoti pikseliai. Iš praeitų sluoksnių gautų ypatybių žemėlapių (aukštesniuose), kuriame apibūdama tam tikras įvesties detales, kaip kampai, kraštinės ir t.t yra sujungiamas su kitu sluoksnio informacija, kuris bus padidinama iki tikros nuotraukos dydžio.



21 pav. U-Net kovoliucinio tinklo struktūros modelis

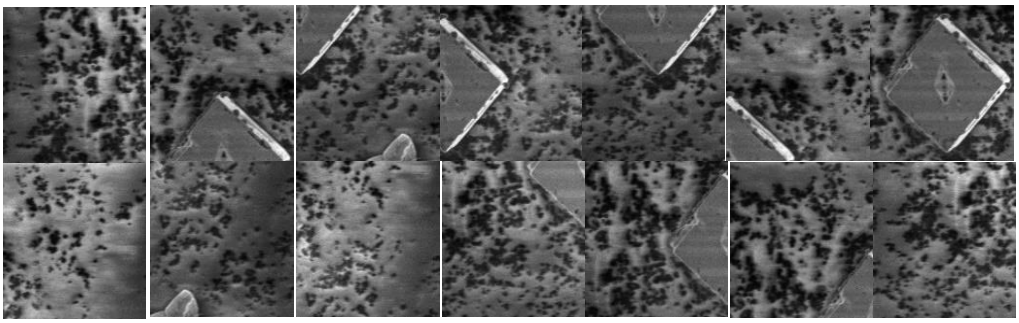
4.4. Patobulinimas

Įvykdysime U-Net architektūros patobulinimą. Pirmoji problema, su kuria susiduriame, tai kad autoriaus pateiktoje architektūroje [47] įvesties ir gauto rezultato nuotraukos dimensijos yra skirtingos [47]. Ši problema iškyla, nes kai vykdome konvoliucinių sluoksnių sudarymą, ypatybių žemėlapio dimensija mažėja, nes nėra naudojama nulių užpildymas. Tad prieš kiekvieną

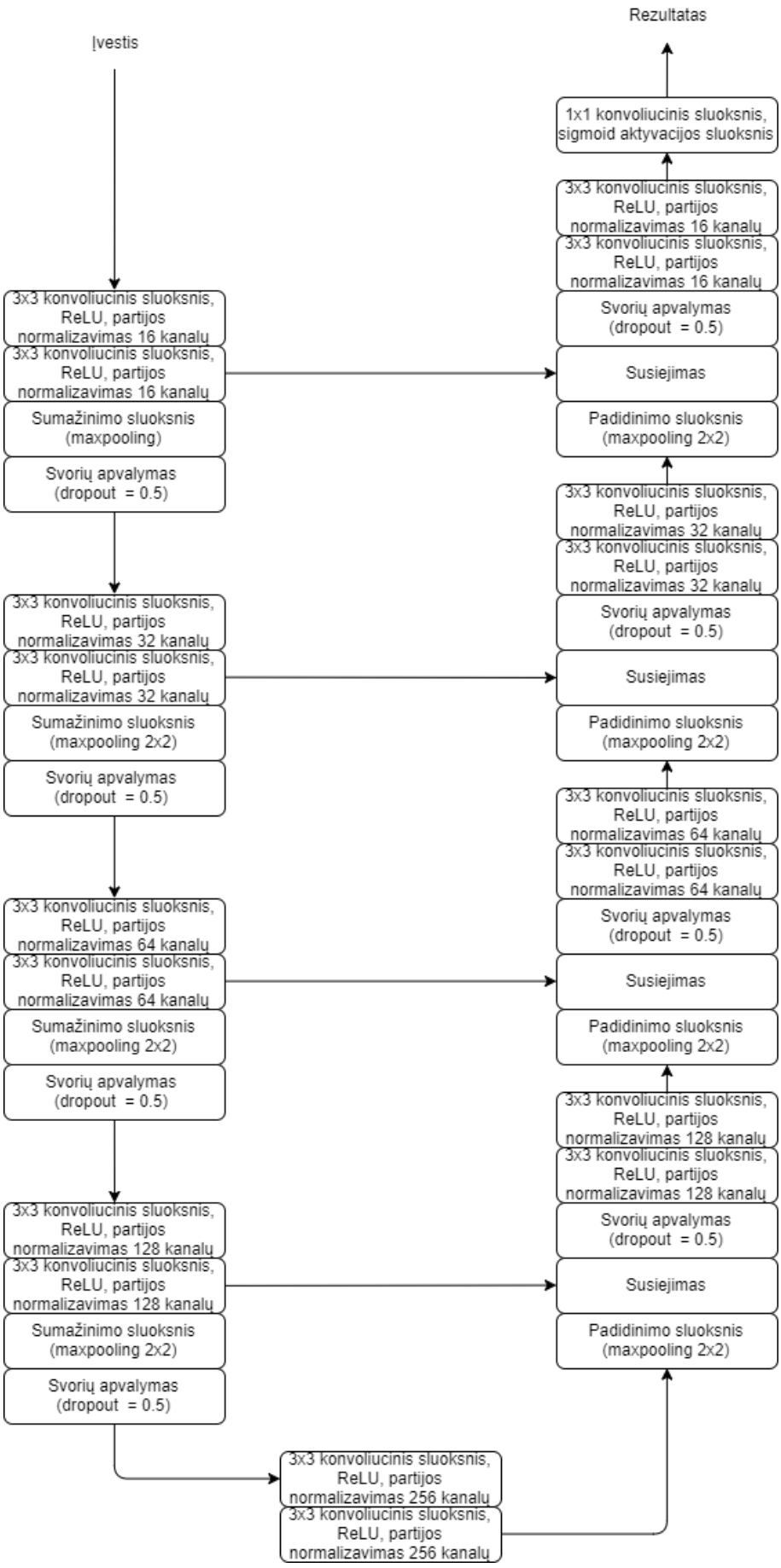
konvoliucinio įvykdymą reikėtų vykdyti nulių užpildymą. Kita problema, kuri gali iškilti, tai skaičiavimo mašinos vaizdo plokštės atminties trūkumas. Šiame darbe mes naudojame didelės rezoliucijos nuotraukas, tik gana galingi kompiuteriai, kurie turi didelės atminties vaizdo plokštę, gali įvykdyti tokių dimensijų semantinį segmentavimą. Yra keletas būdų, kaip išspręsti problemą:

- Pirmiausia galime sumažinti rezoliuciją tam tikrą kartų pvz. per pus ir po semantinio segmentavimo įvykdymo gražinti pirmutinę nuotraukos rezoliucijos dydį. Šis būdas pagreitintų skaičiavimus ir nereikėtų daug resursų, tačiau mes prarasime nemažai informacijos ir tikslumo.
- Antra, jeigu naudojamos SEM nuotraukos analizuojamas objektas yra viduryje galime automatiškai iškirpti nuotrauką tam tikra dimensija, tačiau kitais atvejais, kai objektas yra nuotraukos kraštuose, galime prarasti svarbią informaciją.
- Trečia, galime padalyti nuotrauką į tam tikras lygias dalis, kurie kai kuriuose nuotraukose persidengia dalis pikselių. Po semantinio segmentavimo įvykdymo sudarytas dalis suliejame – jeigu dvejose dalies nuotraukose turime tą patį plotelį ir gali atsitikti, kad viename iš nuotraukų persidengimo plotelių semantinio segmentavimo metu negauname, o kitoje nuotraukoje gauname šią informaciją, galime įvykdyti abiejų nuotraukų rezultatų suliejimą, taip gaudami gana gerą rezultatą. Viena iš problemų, kurių gali iškilti, tai kad po padalijimo tam tikrose nuotraukos kampuose yra nurodomi svarbūs analizuojamo objekto kraštinės ir gali atsitikti, kad po semantinio segmentavimo neaptiksime tų dalių. Šią problemą galime išspręsti panaudodami nulių užpildymą, kad filtras lengviau įsisavintų informaciją ir apmokyti neuroninį tinklą tokio tipo nuotraukomis.

Tad mes naudosime trečiąjį būdą, nes norime išlaikyti tas pačias nuotraukos dimensijas kiekvienose nuotraukose ir neprarasti ypatybių taškų kiekio. Šiame darbe naudosime nuotraukų duomenų rinkinius, kurie turi 1536x1024 dydžio dimensijos nuotraukas. Kiekvienas lopinėlis yra sudarytas 512x512 dydžio ir kiekvienoje nuotraukoje turime 256x256 dydžio persidengimo pikselius. Sudaryti šiuos nuotraukų lopinėlis yra slenkamas 256x256 dydžiu eiliškumo tvarka per visą nuotrauką, taip sudarydami 512x512 dydžio lopinėlius. Gali atsitikti, kad slinkimo metu liks dalis neišanalizuotos nuotraukos. Jeigu taip atsitiks, nuo nuotraukos pabaigos bus paimtas nuotraukos lopinėlis. Iliustracijoje (pav. 22) pateiktas kubo formos nuotraukos rezultatas.



22 pav. Vienos iš SEM nuotraukų perskyrimas į mažas dalis rezultatas



23 pav. Modifikuoto U-Net neuroninio tinklo architektūra

Pateiksime modifikuoto modelio architektūrą, kuri pateikta (pav. 23) iliustracijoje. Kaip ir autoriaus pateiktame modelyje [47] naudosime kodavimo/dekodavimo būdą. Kiekvienas konvoliucinis sluoksnis naudoja 3x3 dydžio filtrus. Kiekvieno etapo blokas susideda iš 2 konvoliucinių sluoksnių. Po kiekvieno konvoliucinio sluoksnio yra įvykdoma partijos normalizavimas (angl. batch normalization), kurio metu normalizuojamas konvoliucinis sluoksnis, taip padidindami veikimo greitį ir stabilumą ir taip pat yra įvykdomas aktyvacijos funkcija - ištaisytas linijinis blokas (angl. rectified linear unit - ReLU). Pereinant iš vieno bloko sluoksnio į kitą yra vykdomas sluoksnio sumažinimas naudojant didžiausios reikšmės radimo (angl. MaxPooling) metodą, kuris naudoja 2x2 filtro matricą. Dar prieš pereinant į kitą bloką yra pritaikomas apskaičiuotų svorių apvalymo (angl. dropout) sluoksnis su parametru 0.5 (toks parametro dydis yra pasirinktas, nes daugumose tyrimuose geriausiai parodo rezultatą). Tarp kodavimo ir dekodavimo blokų – sujungiamasis blokas, kuriame yra pateikti 2 konvoliuciniai sluoksniai, tik be padidinimo (angl. maxpooling) ir apvalymo (angl. dropout) sluoksnių. Tinklo pirmutiniame bloke dėl naudojamų resursų kiekio naudosime ne 64, o 16 ypatybių žemėlapio filtro dydį. Tolesniuose užkodavimo žingsnių sluoksniuose, vykdant kiekvieną bloką ypatybių žemėlapiai dvigubinsis, tol kol pabaigsime kodavimo pusės skaičiavimus, o dekodavimo pusėje yra mažinama per pus kas kartą kai pereinama į kitą bloką. Kiekvieno dekodavimo bloko pradžioje yra vykdomas padidinimo sluoksnio metodas, kuris naudoja perkeltinį konvoliucinį (angl. transposed convolution) skaičiavimą. Prieš kodavimo pusėje perėjimą į kitą bloką yra išsaugojamas ypatybių žemėlapis, kuris yra perduodamas į dekodavimo pusę, kuriame yra sudedamas su tapačiame bloko padėtyje esančiu sluoksniu. Po to yra įvykdomas apvalymo (angl. dropout) sluoksnio metodas ir įvykdomas konvoliucinis sluoksnis. Paskutiniame sluoksnyje mes naudosime Sigmoid aktyvacijos sluoksnį, kuriame išskirsto duomenis į klases. Šiame darbe mes turėsime tik dvi klases - aplinką ir SEM objektą. Visas neuroninis tinklas turės 2164305 parametrų.

Šio neuroninio tinklo sukūrimui naudojame Python programavimo kalbą kartu su Tensorflow[1] ir Keras[10] bibliotekomis. Parašytas programinis kodas yra perkeltas į Jupyter Notebook tipo dokumentą, kad galėtume lengviau paleisti programos kodą skaičiavimų debesies kompiuteryje.

4.4.1. Duomenų paruošimas ir augmentavimas

Apmokyti semantinį segmentavimo neuroninį tinklą, reikia dviejų duomenų tipų: pačių nuotraukų ir klasių nuotraukos dar kitaip vadinamos anotacijos kaukės (angl. mask), kuriose kiekvienas pikselis yra priskirtas tam tikrai klasei. Sudaryti nuotraukų anotacijos objektų kaukes (angl. mask) naudosime labelme programą. Ši programa leidžia lengvai ir tiksliai naudojant poligonų žymėjimą sudaryti reikalingus duomenis. Kiekvieną nuotrauką mes pažymime dvejomis klasėmis: pirma klasė – fonas, kurio kiekvienas pikselis pažymėtas 0 (juoda), antra klasė SEM objektas, kurio kiekvienas pikselis 255 (balta). Naudojant pateiktus analizuojamas duomenų rinkinių nuotraukas tiksliai neužtektų apmokyti. Šią problemą galime išspręsti, įvykdant duomenų augmentavimą. Duomenų augmentavimas padeda sumažinti perpildymo (angl. overfitting) problemą ir padeda neuroniniui tinklui būti daug lankstesniam įvedimo nuotraukoms. Augmentuoti nuotraukas naudosime imgaug biblioteką. Ši biblioteka plačiai naudojama mašininio apmokymo projektams papildyti naujomis nuotraukomis. Ši biblioteka leidžia pasukti, iškreipti nuotrauką tam tikru kampu, iškirpti, uždėti triukšmą ar pašalinti, padidinti ar sumažinti nuotraukos ryškumą ir t.t.

Apmokyti neuroninį tinklą mes kiekvieną duomenų rinkinyje esančią nuotrauką padalijame į tam tikras dalis, taip kaip ankstesniame poskyryje pateiktame būde ir jos dalys yra paveiktos tam

tikro augmentavimo. Be to neuroninio tinklo apmokymui papildysime duomenimis sudarydami SEM nuotraukas, taip kad nuotraukos centre būtų atvaizduojamas objektas. Kiekvienos gautos nuotraukos dimensija yra 676x676 (prieš semantinio vykdymo apmokymą mes sumažinsime rezoliuciją iki 512x512).

Pateiksime tipus, kokiais būdais mes augmentuosime nuotraukas:

- Horizontalus apvertimas bus įvykdytas būdas 50 procentų nuotraukoms.
- Vertikalus apvertimas bus įvykdytas būdas 20 procentų nuotraukoms.
- Nuotraukos pasukimas nuo -45 iki +45 laipsnių.

Neuroninio tinklo apmokymui buvo patestuojama ir su kitais augmentavimo būdais, kaip triukšmo pašalinimu ir padidinimu, kraštinių paryškinimu, tačiau geresnio apmokyto tinklo negavome. Po įvykdyto augmentavimo sugeneravome 1627 nuotraukas kuriame:

- Pirmajam kubo formos objektui sugeneravome 580 nuotraukos lopinėlius ir 120 apkarpytas nuotraukas.
- Antrajam kubo formos objektui 480 nuotraukos lopinėlius ir 108 apkarpytas nuotraukas.
- Spyglių formos objektui gavo 285 nuotraukos lopinėlius ir 54 apkarpytas nuotraukas.

Be to, kaip mes iškerpame nuotraukas į lopinėlius gauname dalis, kuriuose nėra informacijos apie analizuojamą objektą. Tada mes automatiškai išfiltruojame pagal sudarytas objektų kaukes (angl. masks). Jeigu nuotraukose nėra jokios baltos spalvos pikselių, kurie nusako kad pavaizduotas objektas mes pašaliname nuotrauką ir jos objekto kaukę. Išfiltravę duomenis mes gavome 1253 nuotraukas.

Apibendrinant semantinio segmentavimo vykdymą naudojant SEM nuotraukas, pateikiame žingsnius gauti rezultatus:

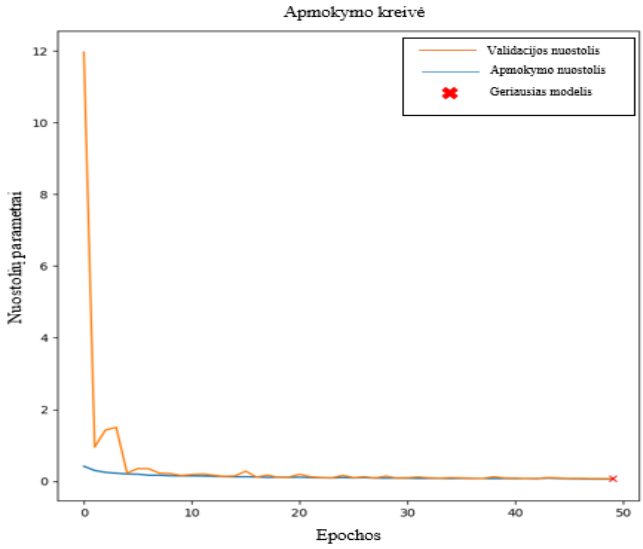
- Kiekvieną nuotrauką padalijame į 512x512 atskiras dalis, kuriame persidengia apie 256x256 pikselių. Išsaugojame informaciją apie šių plotelių poziciją pagal įvestą nuotrauką.
- Įvykdome semantinį segmentavimą būdą su sudarytai nuotraukos ploteliais.
- Gavus visas nuotraukos plotelių rezultatus, mes šias dalis sujungiame į vieną nuotrauką, pagal prieš tai išsaugotas pozicijas. Rezultatą gauname tokio pačio dydžio nuotrauką, tik joje visi pikseliai sužymėti klasėmis.
- Po to yra vykdomas fono pašalinimas pagal įvestos nuotrauką ir semantinio segmentavimo būde gautą nuotrauką, kuriame balta spalvos pikseliai reiškia, kad egzistuoja SEM objektas, o juodos spalvos pikseliai – fonas.
- Kai kuriuose analizuojamo objekto nuotraukose yra pavaizduoti papildomi maži objektai, kurių mums nereikės surekonstruoti. Tai įvykdysime kontūrų aptikimo būdą, pritaikę OpenCV bibliotekoje esančia funkcija, kuri suranda visus nuotraukoje egzistuojančius objektus. Šis paprasčiausiai sujungia visus šalia esančius pikselius pagal spalvą ir intensyvumą.

4.4.2. Neuroninio tinklo apmokymas

Pateiksime kokius hiperparametrus mes naudojame apmokyti neuroninį tinklą. Pirmiausiai pateiksime kokius nustatymus mes naudojame:

- Mokymosi greičio sumažinimas, kai neuroninis tinklas nustoja tobulėti. Šio neuroninio tinklo apmokymo pradžios parametras lygus 0.00001. Jeigu po 3 kartų epochos apmokymo nepavyko patobulinti, tai šis elementas yra pakeliamas 0.1 laipsniu.
- Neuroninio tinklo apmokymo sustabdymas, kai neuroninis tinklas nustoja tobulėti. Jeigu 6 (patience=6) kartus nepavyko patobulinti, tai neuroninio tinklas bus sustabdomas.

Neuroninio tinklo apmokymas užtruko apie 4 valandas naudojant 1253 512x512 dimensijos nuotraukas. Apmokymui mes įvykdėme 70 epochų, o partijos dydis lygus 10 (maksimalus dydžio parametras, kurį galėjome panaudoti, nes skaičiavimo kompiuteris nepajėgia daugiau nuotraukų patalpinti į vaizdo korto atmintį). Epocha, tai būdas, kuriuo metu visi duomenys perduodami nuo pradžios ir nuo pabaigos per visą neuroninį tinklą. Šį elementą naudojame, nes apmokymams turime gana limituotą duomenų rinkinį, jeigu vieną kartą pereisim apmokyti neuroninį tinklą, tai to nepakaks – neuroninio tinklo svoriai (parametrai) nepakankamai tikslūs bus įvestiems duomenims. Tad dėl šios priežasties perduodame tuos pačius duomenis daug kartų neuroniniam tinklui. Partijos dydis, tai dydis, kuris nusako kiek įvesties elementų bus panaudojama apmokyti neuroninį tinklą. Šis elementas yra naudojamas, nes nėra įmanoma visus apmokymo duomenis apmokyti vienu metu, tad dalijame duomenis į tam tikras dalis. Svorių apvalinimą naudosime dropout=0.5 (angl. dropout). Šis elementas, kaip minėjome sumažina per daug tinkančius prie neuroninio tinklo elementus ir naudojama Adam optimizavimas. Nuostolio apskaičiavimui tarp apmokamų ir patvirtinančių duomenų buvo naudojamas „binary_crossentropy“ būdas, nes mes turime tik dvi klases – tai SEM objektas - 1 ir fonas - 0. Kaip minėjome praeitame skyriuje nuostolis interpretuoja, kaip mūsų neuroninio tinklo sudarytas modelis veikia tarp apmokamų ir patvirtintų duomenų. Sudarytas nuotraukas mes perskiriame santykiu 80% (apmokami duomenis) ir 20% (patvirtinti duomenys). Pateiksime iliustraciją (pav. 24), kurioje pateikiama, kaip apmokamas neuroninis tinklas pagal mūsų duomenų nuostolį. Pateiktoje iliustracijoje matome, kad 59 epochoje neuroninio tinklo apmokymas sustojo ir neuroninio tinklo tikslumas pasiektas - 85.34 procentai. Buvo atvejų, kai apmokamas neuroninis tinklas kitais duomenimis pasiekė gana aukštą apie 96.68 procento tikslumą, tačiau patestavus tinklą su kitomis nuotraukomis, gaudavome gana blogus rezultatus – daugumose nuotraukose daug likdavo fono pikselių. Ši problema dar vadinama - per daug gerai apsimokančio neuroninio tinklo problema (angl. overfitting), kurią plačiau aprašėme 2.3 poskyryje.



24 pav. Neuroninio tinklo tikslumas apmokius 70 epochų rezultatas

4.4.3. Vertinimas

Semantinio segmentavimo rezultatų tikslumo patikrinimui yra naudojami keletas vertinimo būdų, kaip pikselių tikslumas, vidutinis pikselių tikslumas ir susikirtimas per sąjunga (angl. IoU) ir vidurkio susikirtimas per sąjungą (angl. meanIoU) būdas. Pateiksime formules su kuriomis mes vertinsime, kaip tiksliai suklasifikavo nuotrauką:

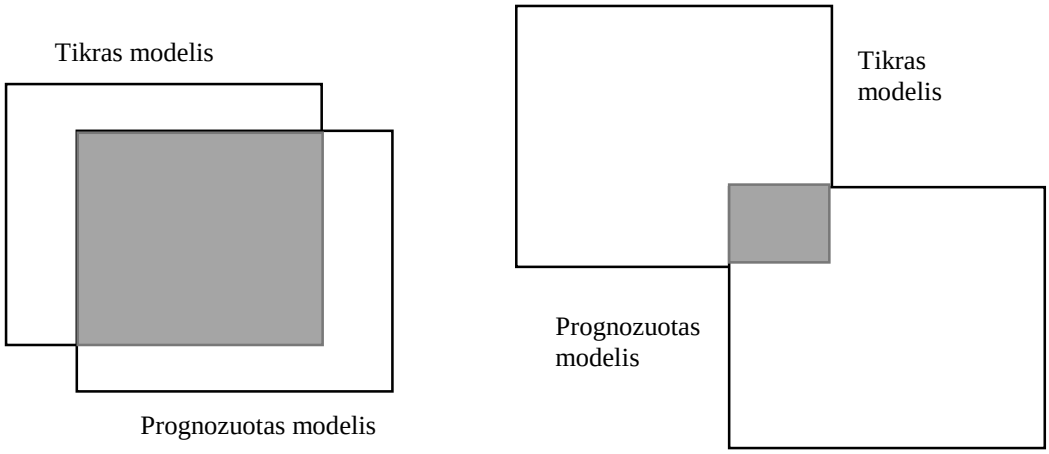
$$\sum_{i} \frac{n_{ii}}{t_i} \tag{3.}$$

Tai pikselio tikslumo formulė (3.), kuri pateikia santykį tarp n_{ii} , kur nurodo gerai gautus klasifikuotus pikselius pagal klasę, o t_i nurodo visos nuotraukos pikselius pagal klasę i.

$$mIoU = \frac{1}{n_{cl}} \sum_{i} \frac{n_{ii}}{(t_i + \sum_j n_{ji} - n_{ii})} \tag{3.}$$

Tai vidurkio susikirtimas per sąjunga formulė, kuri leidžia įvertinti procentą tarp tikrojo modelio kaukės (angl. mask) ir mūsų gauto prognozuoto rezultato. Parametras n_{cl} nurodo kiek klasių yra tikrajame modelyje, n_{ji} nurodo pikselių kiekį, kur šie klasifikuojasi ne i klasėje, bet j klasėje. Iliustracijoje (pav. 25) pateiksime pavyzdžius, kaip naudojant tikrojo modelio kaukę (angl. mask) ir gauto suklasifikuoto rezultato palyginimas naudojant sąjungą. Ko didesnis IoU parametras, to daugiau sutampa pikselių (kaip kairėje pavaizduotame paveikslėlyje).

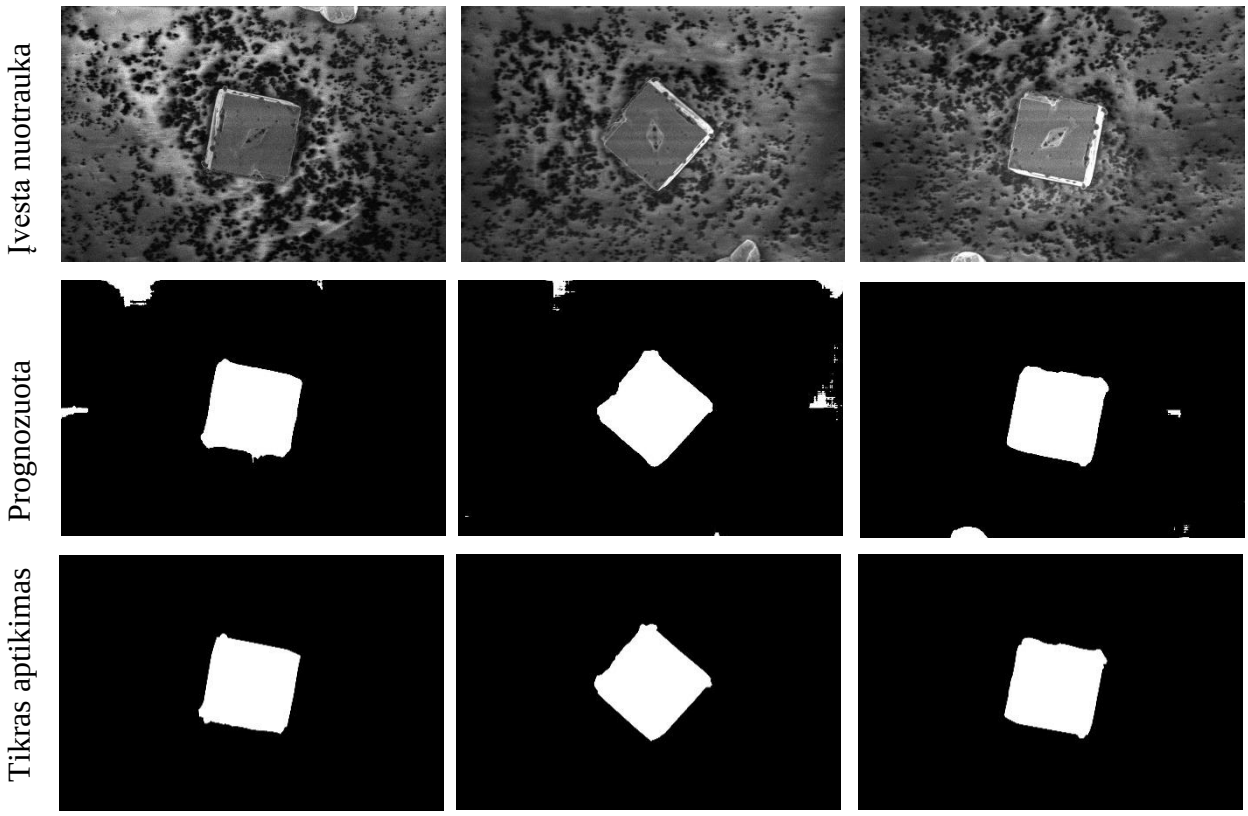
Šiame poskyryje pateiksime semantinio segmentavimo rezultatus naudojant modifikuotą U-Net neuroninį tinklą. Rezultatuose mes pateikiame po 3 rezultatus iš 7 analizuojamų duomenų rinkinių, kurios pateiktos (pav. 26 pav. 27 pav. 28) iliustracijose. Kiekvienose rezultatų iliustracijose mes pateiksime įvesties nuotrauką, prognozuotą modelį naudojant apmokytą tinklą ir taip pat pateiksime po filtravimo įvykdymo, kuriame buvo vykdomas užduotis rasti didžiausią plotą, kuris nusako, kad yra mūsų rezultatas, pašalindamas nereikalingus artifaktus ir kitus objektus. Kiekvienam objektui atvaizduosime geriausią objekto aptikimą ir blogiausią, pagal praeitame poskyryje pateiktas formules. Prie to pačio yra pridedama ir autoriaus pažymėto tikrųjų modelių nuotraukos. Po to pateikiame lentelę (lentelė 1), kurioje yra pateikti vidutinis visų kiekvienoje duomenų rinkinyje esančių nuotraukų tikslumas pagal praeitame poskyryje pateiktas formules. Rezultatuose matome, kad gana tiksliai gauname rezultatus.



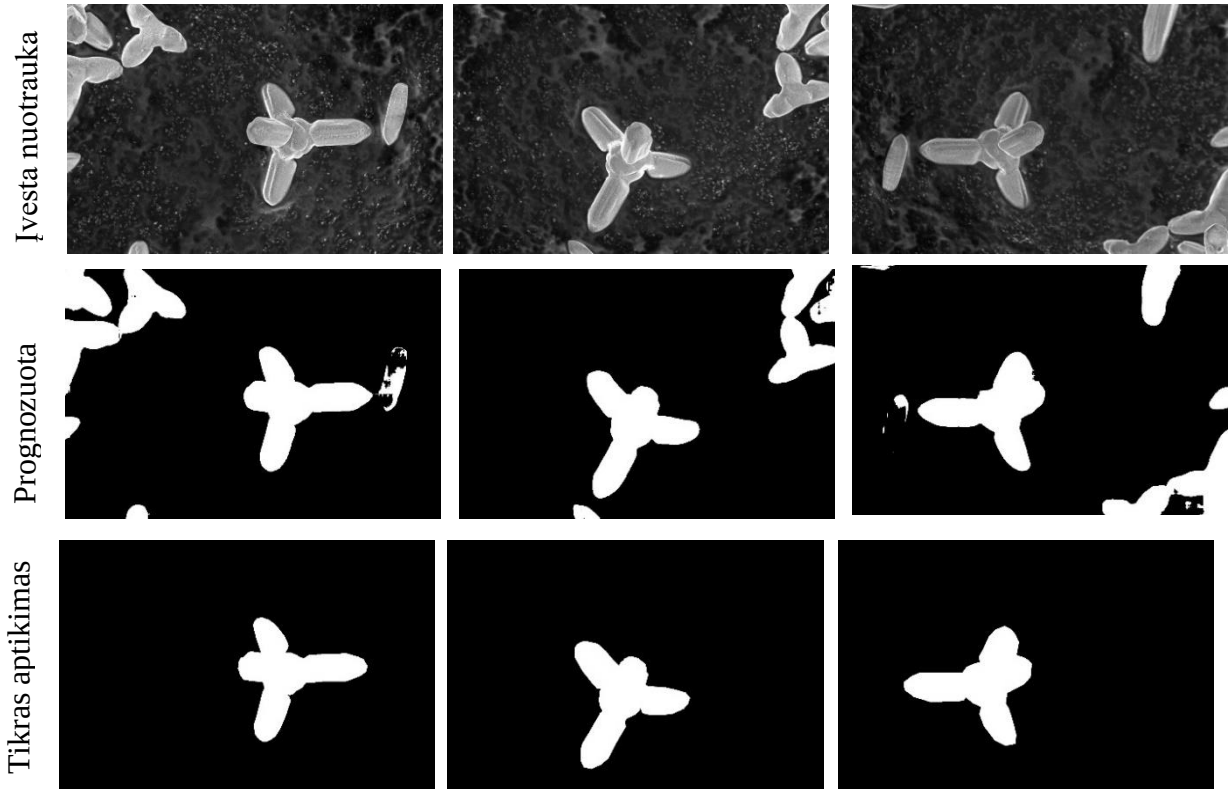
25 pav. Šiuose paveikslėliuose, tiek kairėje ir dešinėje pateikti suklasifikuotų nuotraukų ir tikrojo suklasifikavimo modelių susikirtimas. Pilka sritis reiškia, kiek abejos įvestys susijusios vienas su kita.

1 lentelė. Pateikti kiekvienos duomenų rinkinio vidutiniai rezultatai pagal visas turimas kiekviename rinkinyje esančias nuotraukas

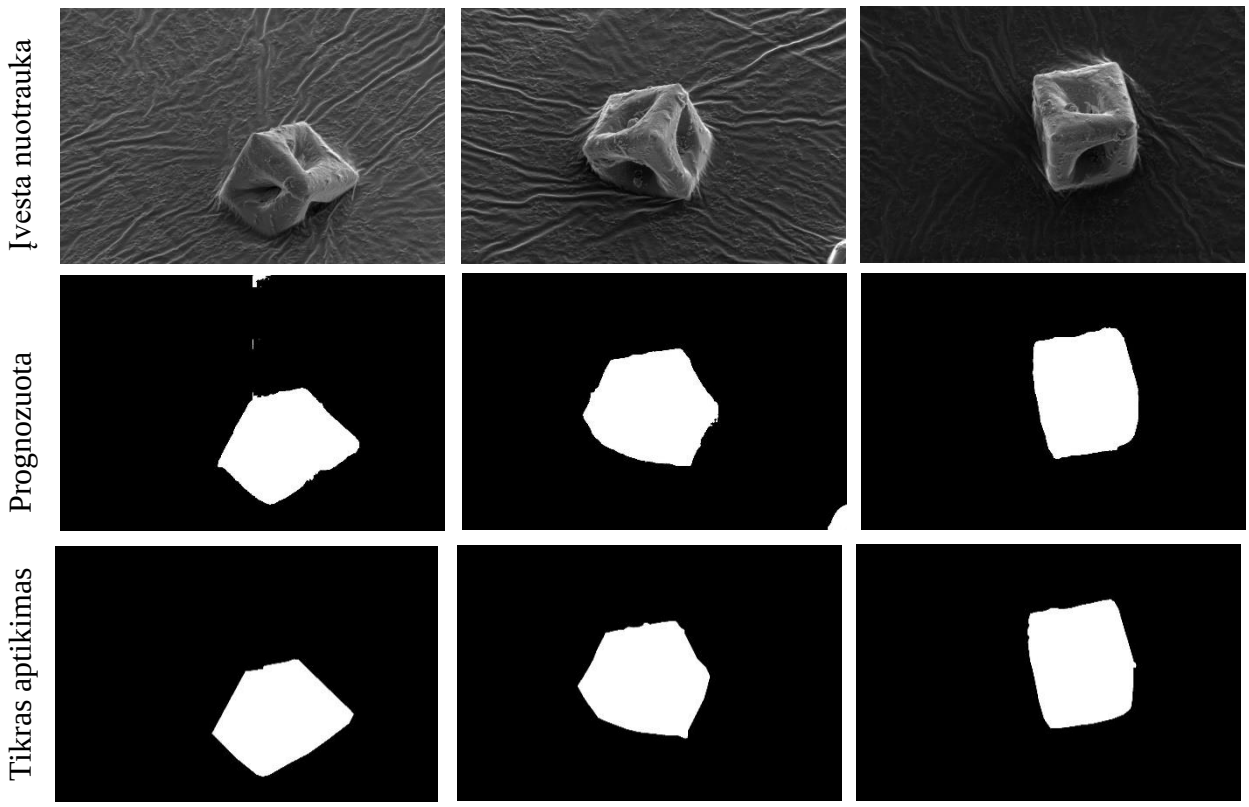
Duomenų rinkinys	**Pikselių tikslumas**	**IoU**
C	86.7	0.8423
D	92.4	0.8947
E	94.5	0.9203



26 pav. C nuotraukų objekto aptikimo rezultatai vykdant semantinį segmentavimą. Pateikiama įvesties nuotraukos, prognozuotas rezultatas ir tikrojo aptikimo rezultatas



27 pav. D nuotraukų objekto aptikimo rezultatai vykdant semantinį segmentavimą. Pateikiama įvesties nuotraukos, prognozuotas rezultatas ir tikrojo aptikimo rezultatas

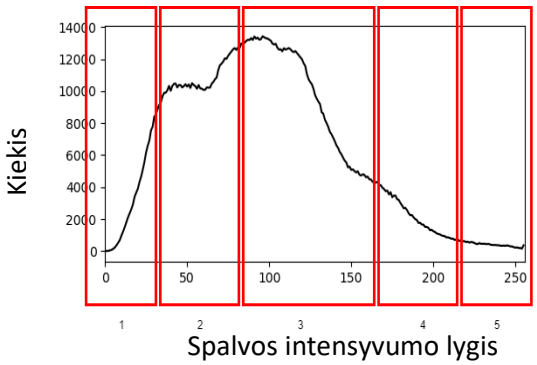


28 pav. E nuotraukų objekto aptikimo rezultatai vykdant semantinį segmentavimą. Pateikiama įvesties nuotraukos, prognozuotas rezultatas ir tikrojo aptikimo rezultatas

4.5. Nuotraukų pagerinimo sprendimas

Kita problema, kuri gali sukelti problemų surekonstruoti tikslų trimatį modelį, tai įvesties duomenys – SEM nuotraukos. Šiose nuotraukose gali būti daug triukšmo, šešėlių, šviesių vietų, artifaktų dėl kurių negalėsime gauti ypatybių ir taip pat gerų atitikmenų tarp porų nuotraukų. Šioje dalyje pritaikysime keletą atvejų, kaip galėtume pagerinti nuotraukos kokybę ir pateiksime rezultatus. Viena iš šios problemos sprendimų yra panaudoti nuotraukos spalvos intensyvumo normalizavimą. Šioje iliustracijoje (pav. 29) pateikiame, kaip sudaryta nuotraukos atspalvių spalvos naudojant histogramos informaciją. Nuotraukos histograma yra padalyta į 5 dalis:

1. Juodos spalvos plotai – juodžiausi pikseliai visoje nuotraukoje. Bet kurie pikseliai šioje skyriuje yra visiškai juodo spalvos.
2. Šešėlių spalvos plotai – daug šviesesnės juodos spalvos pikseliai. Galima pamatyti daugiau detalių šiame ryškumo skyriuje.
3. Vidutiniai tonai – dauguma šių nuotraukos pikselių turėtų būti nuotraukoje.
4. Ryškūs plotai – daug ryškesnė už šviesių pikselių plotus. Panašaus tipo pikseliai, kaip ir šešėlių plotai, tik kad čia vaizduojama ryškūs plotai ir galima pamatyti tam tikras modelio detales.
5. Baltos spalvos plotai – ryškiausi pikselių plotai. Šiame plote negalime jokios informacijos apie vaizduojamą modelį pamatyti.



29 pav. Nuotraukos histogramos pavyzdys suskirstytas į 5 dalis pagal nuotraukų savybes

Išspręsti kontrasto normalizavimą, galime panaudoti pritaikę Riboto kontrasto adaptyvios histogramos išlyginimą (angl. Contrast Limited Adaptive Histogram Equalization - CLAHE)[28], kuriuo metu bando padidinti nuotraukos kontrastą. Šis gauti rezultatą sprendimas įvykdo keletą žingsnių: pirmiausiai nuotrauką perskiria į keletą mažų regionų, kad nereikėtų visą paveikslėlį vienu metu modifikuoti. Po to kiekvienam regionui yra apskaičiuojama histogramos kontrasto funkcijos. Po to gretimi regionai yra sujungiami naudojant linijinę interpoliaciją, kad būtų pašalinamos dirbtinai sudarytos ribos.

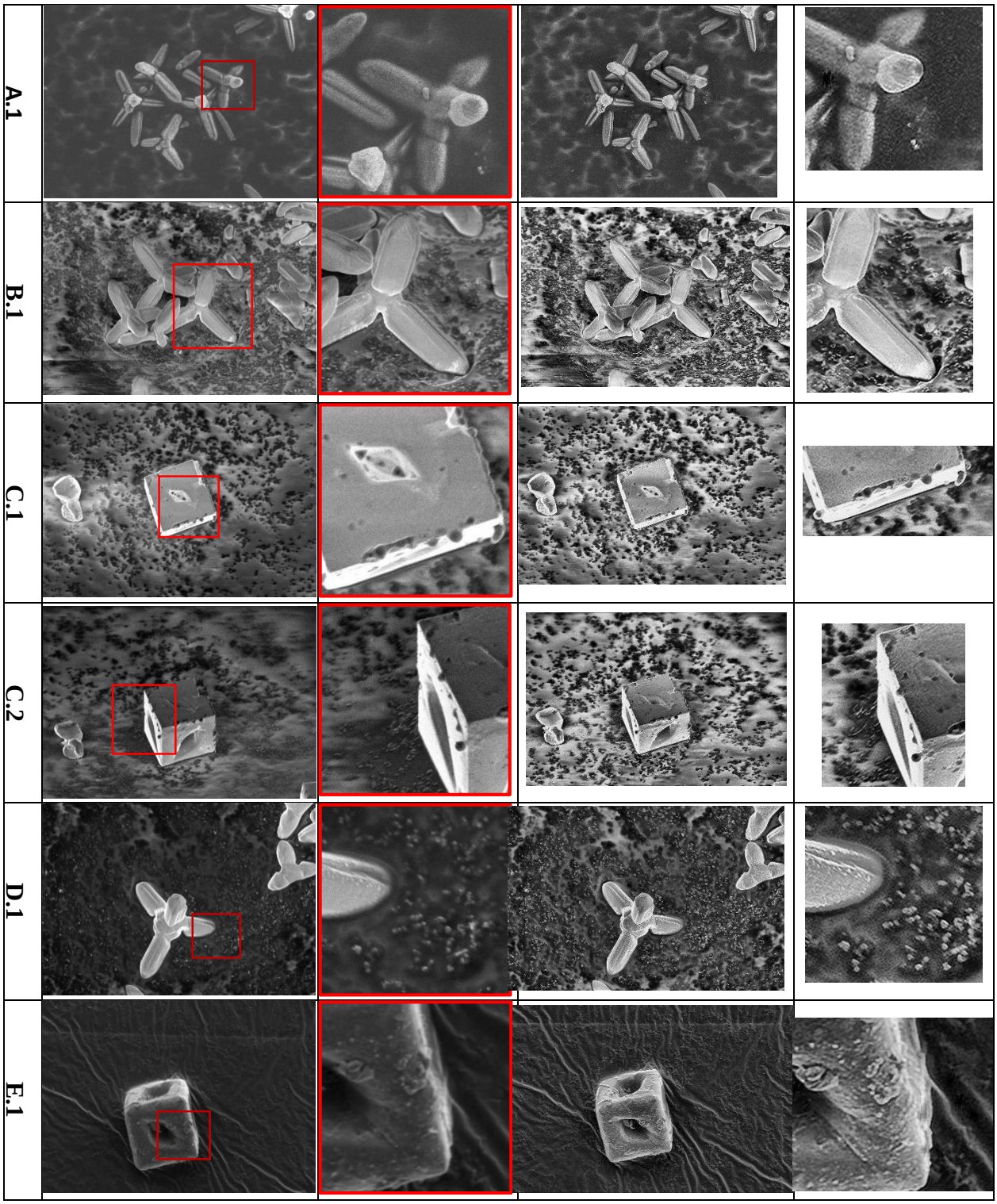
Kita problema – tai triukšmas SEM nuotraukose. Darbo pradžioje pateikėme priežastis kodėl susidaro triukšmas nuotraukos (pirmajame skyrelyje). Išspręsti šią problemą mes panaudosime jau sudarytą neuroninį tinklą, kuriame buvo apmokinta pagal įvairių lygių triukšmų mikroskopines nuotraukas pašalinti triukšmus [66]. Analizuojamoje darbe autorius [66] naudoja keletą triukšmo pašalinimo algoritmų, kurie buvo analizuojami, kaip gerai atlieka užduotį. Autorius darbe išskiria vieną neuroninį tinklą - Noise2Noise[33], kurį ir mes naudosime ir mūsų darbe. Šis neuroninio tinklo modelio apmokyme nenaudoja švarias nuotraukas, bet statistiškai nustato, kaip nuotraukos turi būti be triukšmo. Be to autorius neuroninio tinklo architektūroje įvykdė keletą modifikacijų: buvo pridėtas partijos normalizacijos sluoksnis po kiekvieno konvoliucinio sluoksnio (angl. batch normalization), kuris padeda stabilizuoti neuroninių tinklo apmokymo procesą. Po to yra pridedama ir Tahn aktyvacijos funkcija prieš tinklo išvestį.

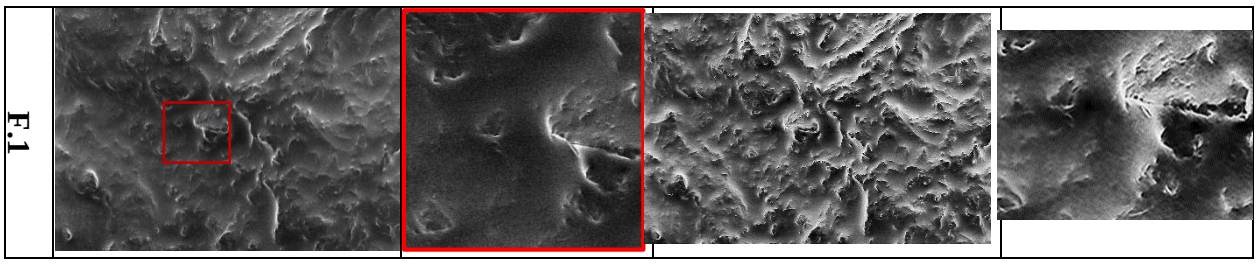
Kita sprendžiama problema – tai nuotraukų neužfokusavimas. Šiame darbe naudosime sukurtą DeblurGanV2 [29] algoritmą. Šis neuroninis tinklas yra priskiriamas prie generacinių tinklų (angl. generative adversarial networks)[17]. Šio tipo neuroniniai tinklai susideda iš dviejų modelių: generatorius G, kuris generuoja naujus pavyzdžius ir diskriminatorius D - vykdomas klasifikavimas, kuriame tikrinama ar sugeneruotas pavyzdys tikras ar netikras. Analizuojamoje algoritme – DeblurGanV2 architektūra susideda iš generatoriaus, kuris sudarytas pagal ypatybės piramidės tinklą (angl. Feature Pyramid Network)[35], kuris daugiausiai buvo naudojamas objektų aptikimui, o po to buvo pritaikomas reliatyvistinio tipo diskriminatorius, kuris apskaičiuoja galimybę, kad tikri duomenys yra daug daugiau realūs negu netikri duomenys. Šio tipo diskriminatorius parodo daug geresnį stabilumą ir apskaičiavimo efektyvumą.

Apmokyti šį neuroninį tinklą buvo naudojamas SEM duomenų rinkinys [5]. Pateiktame rinkinyje yra apie 18,577 nuotraukų, kuriame kiekviena nuotrauka yra 1024x728 rezoliucijos dydžio ir yra išsiskirstomos į 10 kategorijas. Išsamesnė informacija apie SEM nuotraukas pateikta šiame darbe [5]. Tačiau šiame darbe mes nenaudosime visų nuotraukų rinkinių, nes daugumose pateiktose SEM nuotraukų pikseliuose nepateikia gerų ypatybių, raštų (angl. features). Tad šiame darbe mes pasirinkome „nano laidų“ ir „raštuotų paviršių“ SEM rinkinius, iš kurių sudarėme SEM

1478 nuotraukas. Prieš neuroninio tinklo apmokymą, įvykdome SEM nuotraukų modifikavimą, kuriame ant kiekvienos nuotraukos yra uždedama gauso tipo migla.

Išanalizavus keletą nuotraukų apdorojimo sprendinių mes sudarysime veiksmų eilę, kuriame eilės tvarka bus įvykdomi šie sprendimai: pirmiausiai yra įvykdomas histogramos išlyginimas, po to vykdomas fokusavimo uždėjimas naudojant DeblurGanV2 sprendimą ir pabaigoje pašaliname triukšmus vykdant Noise2Noise sprendimo būdą. Naudojant pateiktą sprendimo būdą mes įvykdysime keletą testavimo užduočių naudojant SEM nuotraukas. Pateiktame paveikslėlyje (pav.30) pavaizduojami įvesties ir rezultatų nuotraukos.





30 pav. Pateikto nuotraukų apdorojimo rezultatai naudojant SEM nuotraukas

Pateiktuose rezultatuose (pav. 30) matome sudaryto sprendimo rezultatai, tačiau nevisados galime gerus rezultatus. PVZ. E.1 paveiklėlyje matome drastiškus pasikeitimus į gerąją pusę: pateikiama detalesnis paviršius susinormalizuojasi visas paveikslėlis. Tačiau A.1 paveikslėlyje matome blogesnius rezultatus už įvesties nuotrauką: matome daug daugiau artifaktų ir triukšmo.

5. Struktūra iš judesio būdo sprendimas

Surekonstruoti trimatį modelį naudojant (angl. structure from motion) reikalinga įvykdyti nemažai sudėtingų sprendimų. Svarbiausias sprendimas – gerų atitikmenų (panašumo) gavimas tarp nuotraukų porų (aptarėme 1 skyrelyje). Pagal šių parametrų tikslumą galime rasti analizuojamo objekto trimatėje erdvėje kamerų išsidėstymo parametrus, kurie reikalingi vykdyti pilno (angl. multi view stereo) trimačio modelio sukūrimą. Prieš pateiktą pilną sprendimą, šioje dalyje pateiksime egzistuojančių ypatybių taškų aptikimo ir deskriptoriaus sugeneravimo, atitikmenų radimo ir jų filtravimo algoritmų palyginimus. Vertinime mes apskaičiuosime reliatyvios rotacijos kameros parametrus tarp nuotraukų poros 2D plokštumoje ir apskaičiuosime paklaidą pagal turimas SEM nuotraukų pasukimą tam tikru kampu. Kiekvieną testavimo dalį pateiksime 3 skirtingas nuotraukų išraiškas:

- Pirmajame teste naudosime nemodifikuotas SEM nuotraukas.
- Antrajame teste mes naudosime nuotraukas, kuriuose įvykdomi nuotraukų apdorojimo būdai.
- Trečiajame teste mes naudosime nuotraukas, kuriuose panaudosime tik semantinį segmentavimo būdą išgauti analizuojamą objektą.

5.1. Atitikmenų gavimo testavimas

Pateiksime egzistuojančių ypatybių gavimo, deskriptoriaus sudarymo ir blogų atitikmenų pašalinimo algoritmų palyginimą. Atitikmenų gavimui mes naudosime FLANN (angl. Fast Library for Approximate Nearest Neighbors)[41] biblioteką, kuri lygins deskriptorių vektorius tarp nuotraukų porų naudojant atstūmus. Ypatybių taškų gavimui ir deskriptorių sudarymui naudosime šiuos algoritmus: SIFT, SURF, AKAZE. Atitikmenų pašalinimui mes naudosime šiuos algoritmus: Ransac, Ldmes**,**

Naudojant SEM nuotraukas testavimui neturime tikrų rezultatų, pagal kuriuos galėtumėme remtis ar sudaryti sprendimai patikslina parametrus (rezultatus). Naudojant egzistuojančias programas VisualSFM[61], Colmap[15] sudaryti 3D modelį ir gauti globalios kameros parametrus ne visados gauname, tad šiame darbe truputėlį supaprastinsime mūsų testavimą: pirmiausiai (lentelėse – 4,5,6) vykdysime atitikmenų pašalinimo algoritmų palyginimą naudojant SIFT ypatybių gavimo algoritmą. Šių algoritmų palyginime mes vykdysime reliatyvios rotacijos gavimo sprendimą (pasukimas tarp dviejų nuotraukų) pagal 2D plokštumą. Kitame testavime mes vykdysime ypatybių taškų gavimo sprendimų palyginimą. Gavus iš praeito testavimo rezultatus - pasirinksime geriausią atitikmenų pašalinimo algoritmą ir naudodami įvairius deskriptoriaus ir raktinių taškų gavimo sprendimus, kaip ir praeitame testavime vykdysime reliatyvios rotacijos apskaičiavimą. Šiuos testavimus pateiksime lentelėse (7,8). Be to kiekvienoje testavimo dalyje naudosime vis skirtingas nuotraukų apdorojimo sprendimus.

5.2. Reliatyvios rotacijos kameros parametrų testavimas

Prieš Norint įvykdyti rotacijos kameros parametrų testavimą turime apskaičiuoti rotaciją tarp dviejų nuotraukų. Kai kuriuose naudojamose SEM nuotraukose yra nurodyti x, y, z ašies kampų parametrai. Apskaičiuoti tikrojo modelio rotacijos parametrus tarp dviejų SEM nuotraukų, atimsime kiekvienos ašies parametrus gaudami rotacijos skirtumą tarp dviejų nuotraukų. Po to paversime gautus parametrus į ašies rotacijos matricas pagal 1.2-1.4 punktuose pateiktomis formulėmis. Po to apskaičiuosime bendrą rotacijos matricos parametrą naudojant 1.5 punkte esančia formule. Gavus parametrus išsaugosime JSON tipo failuose. Kiekvieną apskaičiuotą matricą yra gana sudėtinga išsaugoti failuose, duomenų bazėse. Išspręsti šią problemą yra naudojama kvaternijono parametrų išraiškai. Pateiksime kvaternijono išraišką:

$$\boldsymbol{q = w + xi + yj + zk} \tag{5.1}$$

Pats kvaternijono išraiška sudaryta iš 4 elementų - wxyz, kur šie parametrai yra realieji skaičiai, o $i^2 = j^2 = k^2 = ijk = -1$. Atversti iš kvartenijono išraišką į rotacijos matricą pateikiame šioje formulėje:

$$\boldsymbol{R(q)} = \begin{bmatrix} \boldsymbol{w^2 + x^2 - y^2 - z^2} & \boldsymbol{2xy - 2wz} & \boldsymbol{2xz + 2wy} \\ \boldsymbol{2xy + 2wz} & \boldsymbol{w^2 - x^2 + y^2 - z^2} & \boldsymbol{2yz - 2wx} \\ \boldsymbol{2xz - 2wy} & \boldsymbol{2yz + 2wx} & \boldsymbol{w^2 - x^2 - y^2 + z^2} \end{bmatrix} \tag{5.2}$$

Kad nereikėtų pačiam šios formulės sudaryti, panaudosime pyquaternion biblioteką [62], kuri paverčia šiuos parametrus į rotacijos matricą.

$$\boldsymbol{R_{error}} = \sum_{\boldsymbol{i}}^{\boldsymbol{n}} \sum_{\boldsymbol{j}}^{\boldsymbol{n}} |\Delta \boldsymbol{R_{gt}(i,j)} - \Delta \boldsymbol{R_{est}(i,j)}| \tag{5.3}$$

- R_{error} – tai parametras, kuris nusako kiek nutolęs nuo tikslaus objekto parametrų.
- R_{gt} – tai turimo modelio relatyvios rotacijos matrica naudojant egzistuojančius nuotraukos xyz ašies parametrus.
- R_{est} – tai apskaičiuotos reliatyvios rotacijos matrica.
- i, j – tai nuotraukos indeksas

Apskaičiuoti reliatyvios rotacijos paklaidą pagal analizuojamas SEM nuotraukas, mes naudosime (5.3) išraišką. Rasti analizuojamų nuotraukų reliatyvios rotacijos kameros parametrus naudosime skirtingas ypatybių taškų aptikimo, deskriptoriaus sudarymo ir atitikmenų radimo ir jų filtravimo algoritmus. Sprendimo testavime naudosime C, E, F SEM nuotraukų rinkinius (plačiau aprašyta 3 skyrelyje). Testavime mes stebėsime, kaip modifikuotos nuotraukos pagerina ar pablogina gauti tikslesnius reliatyvios rotacijos parametrus.

5.3. Rezultatai

5.3.1. Atitikmenų pašalinimo rezultatai

4 lentelė. Pateikiami atitikmenų pašalinimo rezultatai naudojant skirtingus algoritmus naudojant skirtingas SEM duomenų rinkinius, kuriose naudojamos nemodifikuotos nuotraukos be pagrindo pašalinimo sprendimo. Lentelėje pateikiami reliatyvios rotacijos rezultatai.

N.Rinkinys Algoritmas	C(nuo 0 iki 10 laipsnių)	D (paimama nuo 0 iki 10 laipsnių)	E (paimama nuo 0 iki 20 laipsnių)	F (nuo 0 iki 60 laipsnių)
RANSAC	xm498.4983317115418 ym1458.2900942715573 zm798.2135785728696	xm2052.3040599543338 ym3335.807418432605 zm772.4368578380564	xm5312.918435857119 ym2490.449970386065 zm461.0469386826871	xm202.88184054782474 ym378.750687620116 zm60.658924237302216
LMEDS	Xm181.1061156877576 ym1594.7000392606883 zm78.50114621531205	xm625.8213584609958 ym4122.447176968299 zm79.41085149566405	xm280.6695677183257 ym2052.110047456462 zm51.009849444857586	xm9.039943703286836 ym295.8640442888429 zm7.177890416963108

Lentelėje pateikiama rotacijos paklaidos suma (kiek laipsnių buvo nukrypęs nuo tikrojo kampo suma). Matome, kad LMEDS pirmauja pagal tikslumą, tik D SEM nuotraukų rinkinyje pagal ym (y ašies) yra daugiau pakrypęs. Viena iš priežasčių, kad RANSAC algoritmas nusileidžia LMEDS tuo, kad RANSAC daugiau palieka neatitikmenų.

5 lentelė. Pateikiami atitikmenų pašalinimo rezultatai naudojant skirtingus algoritmus naudojant skirtingas SEM duomenų rinkinius kuriose naudojamos nemodifikuotos nuotraukos su pagrindo pašalinimo sprendimu. Lentelėje pateikiami reliatyvios rotacijos rezultatai.

N. Rinkinys Algoritmas	C(nuo 0 iki 10 laipsnių)	D (paimama nuo 0 iki 10 laipsnių)	E (paimama nuo 0 iki 20 laipsnių)	F (nuo 0 iki 60 laipsnių)
RANSAC	xm4522.545435345 ym4565.5645345345 zm532.5434543453	xm6456.3348778656 ym354.7567865323 zm4353.54645687	xm211.6453453453 ym354.750687620116 zm86.996598989	xm4522.545435345 ym4565.5645345345 zm532.5434543453
LMEDS	xm837.85456465 ym4562.4645686 zm86.6546546565	xm295.54353453383 ym456.5435348346 zm74.45348438456	xm12.5468786753 ym302.575375353565 zm9.5787878678565	xm837.85456465 ym4562.4645686 zm86.6546546565

6 lentelė. Pateikiami atitikmenų pašalinimo rezultatai naudojant skirtingus algoritmus naudojant skirtingas SEM duomenų rinkinius, kuriose naudojamos modifikuotos nuotraukos be pagrindo pašalinimo sprendimo. Lentelėje pateikiami reliatyvios rotacijos rezultatai.

N. Rinkinys Algoritmas	C(nuo 0 iki 10 /laipsnių)	D (paimama nuo 0 iki 10 laipsnių)	E (paimama nuo 0 iki 20 laipsnių)	F (nuo 0 iki 60 laipsnių)
RANSAC	xm2658.4654648885418 ym3868.568696255565 zm656.47856345	xm496.5464563 ym4565.5589123 zm532.45787864	xm6832.6546456 ym8524.456456 zm4728.656543	xm174.86786755653 ym486.786786752 zm214.56767686545
LMEDS	xm1022.5657782348 ym2685.3533453468 zm857.56434538868	xm427.4523243534 ym3846.45756543 zm74.5457575468	xm750.52533540 ym686.897898 zm557.6456546	xm47.87687656 ym269.89735 zm75.7235879786

5 ir 6 lentelėse pateikiama rotacijos paklaidos suma (kiek laipsnių buvo nukrypęs nuo tikrojo kampo suma). Matome, kad LMEDS pirmauja pagal tikslumą.

5.3.2. Raktinių taškų ir deskriptoriaus testavimo rezultatai

7 lentelė. Pateikta reliatyvios rotacijos paklaidos vykdant skirtingas ypatybių taškų gavimo sprendimus naudojant ir atitikmenų filtravimo algoritmą rezultatai, kai naudojamos nemodifikuotos nuotraukos be pagrindo pašalinimo sprendimo

N. Rinkinys Algoritmas	C(nuo 0 iki 10 laipsnių)	D (paimama nuo 0 iki 10 laipsnių)	E (paimama nuo 0 iki 20 laipsnių)	F (nuo 0 iki 60 laipsnių)
SURF	Xm4575.54355245 Ym3488.254253 zm656.254245	Xm456.73755 ym2475.3535 zm823.8868282	xm2358.2782783 ym354.28773783 zm453.8361525	xm574.278687378 ym449.9876727 zm2184.5345868
AKAZE	xm2824.53453838 ym228.5453453 zm528.53453778	xm725.46564565 ym5645.56456352 zm564.6546456456	xm2122.56434534 ym242.454545 zm51.87678678	xm565.4545345 ym288.687678 zm107.78686786

8 lentelė. Pateikta reliatyvios rotacijos paklaidos vykdant skirtingas ypatybių taškų gavimo sprendimus naudojant ir atitikmenų filtravimo algoritmą rezultatai, kai naudojamos nemodifikuotos nuotraukos su pagrindo pašalinimo sprendimu

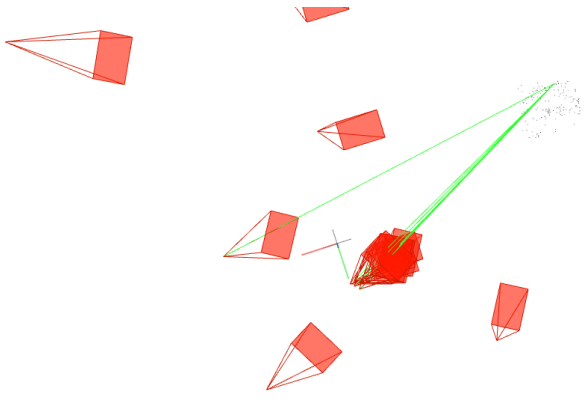
N. / Rinkinys / Algoritmas	C(nuo 0 iki 10 laipsnių)	D (paimama nuo 0 iki 10 laipsnių)	E (paimama nuo 0 iki 20 laipsnių)	F (nuo 0 iki 60 laipsnių)
SURF	Xm3212.6876752752 Ym5775.5757377 zm272.27278	Xm4878.86865643 ym875.786785645 zm88.68767856565	xm686.89485345 ym3574.5645648 zm878.45689	xm174.86786755653 ym3486.786786752 zm214.56767686545
AKAZE	xm4565.757554546 ym5425.46227878 zm455.787852	xm6245.4523243534 ym452.45756543 zm86.5457575468	xm293.65463453 ym4546.32322135 zm68.354568989	xm47.87687656 ym2691.89735 zm275.72355757

7 ir 8 lentelėse pateikiama rotacijos paklaidos suma (kiek laipsnių buvo nukrypęs nuo tikrojo kampo suma) vykdant skirtingas ypatybių gavimo algoritmus. Matome, kad AKAZE algoritmas veikia geriausiai, tačiau pagal ankstesnius 4,5,6 testavimus, kuriame buvo vykdomas SIFT algoritmas, gauname geriausius rezultatus.

5.4. Reto 3D modelio sudarymo testavimas

Šioje dalyje mes pažvelgsime reto 3D modelio sukūrimo rezultatus. Taikydami mūsų pateiktus sprendimus modifikuosime atviro kodo programą – COLMAP [15]. Ši programa leidžia įvykdyti visus struktūros iš judesio žingsnius, kaip aptikti ypatybių taškus, rasti atitikmenis tarp nuotraukų, trianguliacija, paketų koregavimas, kameros parametrų aptikimas. Be to yra galimybė sudaryti tankų taškų debesį ir konvertuoti taškų debesis ir kitus parametrus į kitų atvirų trimačio modelio rekonstrukcijos programų, kaip pvz. VisualSFM[61] failus. Visa programa parašyta C++ programavimo kalba. Vykdyti funkcijas yra galimybė naudoti grafinę programos langą arba kviesti funkcijas iš komandinės eilutės. Be to yra parašytos pakalbinės funkcijos, kurios leidžia Python programavimo kalba praplėsti programos funkcionalumą.

Pagal praeitų skyrelių analizavimo rezultatus, geriausius sprendinius pateiksime į struktūros iš judesio struktūrą. Pagrindinė problema testuojant naudojant SEM nuotraukas - tikslių duomenų turėjimas, kurie gaunami sudarant 3D modelį, su kurias būtų galima palyginti sugalvotus sprendimus. Tad šiame darbe mes truputėlį supaprastinsime vertinimą. Pateikiame vieną pavyzdį, vykdant E tipo duomenų sudarymui.



31 pav. Pateikiamas 3D modelio pavyzdys su kamerų išsidėstymu

Išvados ir rekomendacijos

Apibendrinant šį baigiamąjį magistrinį darbą, pateiksime keletą išvadų:

- Įvykdžius dvi skirtingas SEM nuotraukų apdorojimo būdus: semantiniu segmentavimu ir triukšmo, suliejimu pašalinimu ir miglos kontrasto padidinimu buvo gauti neblogi rezultatai.
- Įvykdžius skirtingų atitikmenų pašalinimo algoritmų palyginimą, kuriame buvo vykdomi reliatyvios rotacijos paklaidos apskaičiavimai, rezultatuose matome, kad RANSAC algoritmas veikia blogiausiai.
- Įvykdžius skirtingų ypatybių gavimo sprendimų palyginimą, kuriame buvo vykdomi reliatyvios rotacijos paklaidos apskaičiavimai, rezultatuose matome, kad SIFT algoritmas veikia geriausiai ir yra gauti geresni atitikmenys už SURF ir AKAZE.
- Vykdant struktūros iš judesio būdą rezultatuose matome, kad vis tiek reikia gana gerų SEM nuotraukų, kad būtų galima sudaryti 3D modelį. Vykdant nuotraukų apdorojimo sprendimus atitikmenuose gauname blogesnius rezultatus, nes CLAHE histogramos algoritmas pablogina atitikmenų gavimą ir deskriptoriaus sudarymą.
- Vykdant nuotraukų apdorojimo sprendimus negalėjome sudaryti 3D modelių dėl neefektyvių atitikmenų tarp nuotraukų.
- Mikroskopo pagalba gaunant SEM nuotraukas, kiekvienai iš jų galima įrašyti, kaip buvo pasuktas mėginys (x, y, z rotacijos parametrus). Turint daugiau nuotraukų būtų galima atlikti analizę, kokiu kampo žingsniu rekomenduotina daryti nuotraukas, kad 3D objektas jau būtų pakankamai tikslus, tad naudojamų nuotraukų kiekis nebūtų per daug, kad skaičiavimai neužtruktų per ilgai.

Ateities tyrimų planas

Šioje dalyje aprašomas galimas ateities tyrimų planas. Šios užduotys gali būti atliekamos kitose darbuose:

- Naudojant iš struktūros iš judesio gautus vidinius ir išorinius kameros parametrus įvykdyti stereoskopinį 3D rekonstrukcijos (angl. multi view stereo) sprendimą, kuriame sudaromas pilnas, tikslus 3D modelis.
- Pritaikyti giliąsias apmokamus tinklus sudaryti gylio žemėlapius (angl. depth map), kuriuo metu pagal 2 ar daugiau nuotraukų ir jų vidiniais ir išoriniais kamerų parametrais sudaromas sluoksnis su kuriuo galima sujungti į vieną 3D modelį panaudojus keletą tokių pačių sluoksnių.
- Išanalizuoti algoritmus, kaip būtų galima pagerinti gylio sluoksnio kokybę.
- Išspręsti nuotraukų parinkimo problemą prieš vykdant (angl. multi view stereo) sprendimą.
- Panaudoti super-rezoliucijos sprendimą, kuriame vykdomas gylio plokštumos sudarymo rezoliucijos padidinimas, kurio pagalba būtų galima gauti daug detalesnį 3D modelį.
- Išspręsti standartinio paketų koregavimo (angl. bundle adjustment) algoritmo optimizavimą, kad veiktų daug našiau ir tiksliau gautų nuotraukų išsidėstymą trimatėje erdvėje (išorinius) parametrus.
- Daugiau paanalizuoti kameros parametrų 3D erdvėje gavimą naudojant n taško perspektyvo (angl. perspective-n-point) sprendinius.
- Ištirti daugiau gylio žemėlapių sukūrimo ir filtravimo sprendimus, kuriuose naudojami neuroniniai tinklai. Be to pateikti sprendimą, kaip naudojant vieną ir porą nuotraukų sugeneruoti gylio žemėlapius ir neuroninių tinklų pagalbą sulieti juos taip pagerindami rezultatus.
- Daugiau ištirti ypatybių taškų gavimo, atitikmenų gavimo ir jų pašalinimo algoritmus.
- Daugiau ištirti nuotraukų apdorojimo sprendimus ir ištirti kitus neuroninių tinklų architektūras, kurie pagerintų SEM nuotraukos kokybę.
- Pritaikyti ir ištirti, kaip veikia naudojant progresuojančią 3D rekonstrukcijos metodą, kuriuo metu mes padalijame vykdymą į tam tikras dalis ir po to kiekvienos dalies trimačio modelio taškų debesis sujungiamas į vieną modelį.
- Pabandyti išbandyti sprendimus, kuriuose yra apmokamas neuroninis tinklas, kurio pagalba galima pagal vieną nuotrauką surekonstruoti trimatį modelį.
- Pateikti daugiau SEM nuotraukų duomenų rinkinių, kuriuose būtų daug įvairių nuotraukų skirtingomis šviesų kryptimis ir skirtingomis analizuojamo objekto perspektyvomis. Be to pateikti įvertinimo (testavimo) projektą tyrėjams, kuriame būtų vertinama du sprendimai: kaip įvykdant nuotraukų apdorojimo algoritmus pagerina nuotraukų kokybę ir kaip taikant trimačio sukūrimo metodus tiksliai surekonstruojamas 3D modelis.
- Sukurti WEB sistemą, kuriame vartotojams būtų patogiai sudaryti 3D modelį.

Literatūros sąrašas

[1] Martın Abadi, Ashish Agarwal, TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems, 2015, https://arxiv.org/pdf/1603.04467.pdf

[2] Pablo F. Alcantarilla, Adrien Bartoli, Andrew J. Davison, KAZE Features, 2012, http://isit.u-clermont1.fr/~ab/Publications/Alcantarilla_Bartoli_Davison_ECCV12.pdf

[3] Pablo F. Alcantarilla, Jesús Nuevo, Adrien Bartoli, Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces, 2013, http://www.robesafe.com/personal/pablo.alcantarilla/papers/Alcantarilla13bmvc.pdf

[4] Jonathan Atteberry, How Scanning Electron Microscopes Work, http://science.howstuffworks.com/scanning-electron-microscope2.htm

[5] Rossella Aversa, Mohammad Hadi Modarre, Stefano Cozzini, Regina Ciancio, Alberto Chiusole, The first annotated set of scanning electron microscopy images for nanoscience, 2018, https://www.nature.com/articles/sdata2018172

[6] Daniel Barath, Jiri Matas, Jana Noskova, MAGSAC: Marginalizing Sample Consensus, 2019, https://openaccess.thecvf.com/content_CVPR_2019/papers/Barath_MAGSAC_Marginalizing_Sample_Consensus_CVPR_2019_paper.pdf

[7] Jon Louis Bentley, Multidimensional Binary Search Trees Used for Associative Searching, 1975, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.160.335&rep=rep1&type=pdf

[8] JiaWang Bian, Wen-Yan Lin, Yasuyuki Matsushita, Sai-Kit Yeung, Tan Dat Nguyen, Ming-Ming Cheng, GMS: Grid-based Motion Statistics for Fast, Ultra-robust Feature Correspondence IEEE CVPR, 2017, http://jwbian.net/Papers/GMS_CVPR17.pdf

[9] Moses S. Charikar, Similarity estimation techniques from rounding algorithms,2002, https://dl.acm.org/doi/10.1145/509907.509965

[10] François Chollet, Keras, 2015, https://github.com/keras-team/keras

[11] Ondrej Chum, Two-view Geometry Estimation Unaffected by a Dominant Plane, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.466.2719&rep=rep1&type=pdf

[12] Ondrej Chum, Matching with PROSAC – Progressive Sample Consensus, 2005, https://dspace.cvut.cz/bitstream/handle/10467/9496/2005-Matching-with-PROSAC-progressive-sample-consensus.pdf;jsessionid=A6E065F62D4C7B7A4B46CD326ADB0A92?sequence=1

[13] Titus Cieslewski, Michael Bloesch, Davide Scaramuzza, Matching Features without Descriptors: Implicitly Matched Interest Points, 2018,https://arxiv.org/pdf/1811.10681.pdf

[14] Titus Cieslewski, Davide Scaramuzza, SIPS: Unsupervised Succinct Interest Points, 2018, https://arxiv.org/pdf/1805.01358.pdf

[15] COLMAP, https://colmap.github.io/

[16] Roberto Cipolla, Generalised Epipolar Constraints, 1999, http://www1.maths.lth.se/matematiklth/vision/publdb/reports/pdf/astrom-cipolla-etal-ijcv-99.pdf

[17] Antonia Creswe, Tom White, Generative Adversarial Networks: An Overview, 2017, https://arxiv.org/pdf/1710.07035.pdf

[18] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, Torsten Sattler, D2-Net: A Trainable CNN for Joint Detection and Description of Local Features ,2019, https://arxiv.org/abs/1905.03561

[19] Vincent Dumoulin, Francesco Visin, A guide to convolution arithmetic for deep learning, 2018, https://arxiv.org/pdf/1603.07285.pdf

[20] FEI Company, An introduction to electron microscopy, 2017 http://www.cqmfscience.com/wp-content/uploads/2017/03/N_Braidy_TEM_FEI.pdf

[21] FEI Company, All you wanted to know about Electron Microscopy, http://web.pdx.edu/~pmoeck/pdf/all%20you%20wanted%20to%20know%20about%20electron%20microscopy.pdf

[22] Daniel DeTone, Tomasz Malisiewicz, Andrew Rabinovich, SuperPoint: Self-Supervised Interest Point Detection and Description, 2018, https://arxiv.org/pdf/1712.07629.pdf

[23] Konstantinos G. Derpanis, Overview of the RANSAC Algorithm, 2010, http://www.cse.yorku.ca/~kosta/CompVis_Notes/ransac.pdf

[24] Wlodzimierz Drzazga, Jaroslaw Paluszynski and Witold Slowko, Three-dimensional characterization of microstructures in a SEM, 2006, http://iopscience.iop.org/article/10.1088/0957-0233/17/1/006

[25] Jeff Johnson, Matthijs Douze, Hervé JégouBillion-scale similarity search with GPUs, 2017, https://arxiv.org/abs/1702.08734

[26] Kevin de Haan, Zachary S. Ballard, Yair Rivenson, Yichen Wu, Aydogan Ozcan, Resolution enhancement in scanning electron microscopy using deep learning, 2019, https://www.nature.com/articles/s41598-019-48444-2.pdf

[27] Kun He, Yan Lu, Stan Sclaroff, Local Descriptors Optimized for Average Precision, 2018, http://openaccess.thecvf.com/content_cvpr_2018/CameraReady/0368.pdf

[28] Etta D. Pisano, Shuquan Zong, Bradley M. Hemminger, Marla Deluca, R. Eugene Johnston, Keith Muller, M. Patricia Braeuning, and Stephen M. Pize, Contrast Limited Adaptive Histogram Equalization Image Processing to Improve the Detection of Simulated Spiculations in Dense Mammograms, 1998, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3453156/pdf/10278_2009_Article_BF03178082.pdf

[29] Orest Kupyn, Tetiana Martyniuk, Junru Wu, Zhangyang Wang, DeblurGAN-v2: Deblurring (Orders-of-Magnitude) Faster and Better, 2019, https://arxiv.org/abs/1908.03826

[30] V. V. Kniaz, V. V. Fedorenko, N. A. Fomin, Deep learning for low textured iamge matching, 2018, https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLII-2/513/2018/isprs-archives-XLII-2-513-2018.pdf

[31] Jan Larsen, Introduction to Artificial Neural Networks, 1999, https://orbit.dtu.dk/files/2717948/imm2443.pdf

[32] Stefan Leutenegger, Margarita Chli, Roland Y. Siegwart, BRISK: Binary Robust Invariant Scalable Keypoints, 2011, https://www.robots.ox.ac.uk/~vgg/rg/papers/brisk.pdf

[33] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, Timo Aila , Noise2Noise: Learning Image Restoration without Clean Data,2018, https://arxiv.org/abs/1803.04189

[34] Na Li, Jiquan Yang, Triangulation Reconstruction for 3D Surface Based on Information Model, 2016, https://content.sciendo.com/downloadpdf/journals/cait/16/5/article-p27.xml

[35] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, Serge Belongie, Feature Pyramid Networks for Object Detection, 2017, https://arxiv.org/abs/1612.03144

[36] Wen-Yan Lin, Fan Wang, Ming-Ming Cheng, Sai-Kit Yeung, Philip H.S. Torr, CODE: Coherence Based Decision Boundaries for Feature Correspondence, 2018, https://ieeexplore.ieee.org/document/7815414

[37] Tony Lindeberg, Scale Invariant Feature Transform, 2012, http://www.diva-portal.org/smash/get/diva2:480321/FULLTEXT02.pdf

[38] Xiao Xin LU, A Review of Solutions for Perspective-n-Point Problem in Camera Pose Estimation, 2009 https://iopscience.iop.org/article/10.1088/1742-6596/1087/5/052009/pdf

[39] Zixin Luo, Tianwei Shen, Lei Zhou, Siyu Zhu, Runze Zhang, Yao Yao, Tian Fang and Long Quan, GeoDesc: Learning Local Descriptors by Integrating Geometry Constraints, 2018, https://arxiv.org/abs/1807.06294

[40] Zixin Luo, Tianwei Shen, Lei Zhou, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, Long Quan, ContextDesc: Local Descriptor Augmentation with Cross-Modality Context, 2019, https://arxiv.org/abs/1904.04084

[41] Marius Muja, David G. Lowe, Fast approximate nearest neighbors with automatic algorithm configuration, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.160.1721&rep=rep1&type=pdf

[42] Antonis Nanakoudis, SEM: types of electrons, their detection and the information they provide, 2017, http://blog.phenom-world.com/sem-electrons-detection-provide-information

[43] Yuki Ono, Eduard Trulls, Pascal Fua, Kwang Moo Yi, LF-Net: Learning Local Features from Images, 2018, https://arxiv.org/pdf/1805.09662.pdf

[44] Jacob Toft Pedersen, SURF: Feature detection and description, http://cs.au.dk/~jtp/SURF/report.pdf

[45] M Satya Prasad, David C Joy, Is SEM Noise Gaussian?, 2003,https://www.cambridge.org/core/services/aop-cambridge-core/content/view/B6DDC875EFF4339183917F1F74C82672/S1431927603444917a.pdf/div-class-title-is-sem-noise-gaussian-div.pdf

[46] Mohammad Reza Abbasifard ,A Survey on Nearest Neighbor Search Methods, 2014, https://www.researchgate.net/publication/270450288_A_Survey_on_Nearest_Neighbor_Search_Methods

[47] Olaf Ronneberger, Philipp Fischer, Thomas Brox, U-Net: Convolutional Networks for Biomedical Image Segmentation, 2015, https://arxiv.org/pdf/1505.04597.pdf

[48] Carsten Rother, Vladimir Kolmogorov, Andrew Blake, "GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts, 2001, https://www.microsoft.com/en-us/research/wp-content/uploads/2004/08/siggraph04-grabcut.pdf

[49] Ethan Rublee, Vincent Rabaud, Kurt Konolige, Gary Bradski, ORB: an efficient alternative to SIFT or SURF, 2011, http://www.willowgarage.com/sites/default/files/orb_final.pdf

[50] Torsten Sattler, Out with the Old? Convolutional Neural Networks for Feature Matching and Visual Localization, 2017, http://www.ifp.uni-stuttgart.de/phowo/2017/PDF/16-Sattler-Abstract.pdf

[51] Nikolay Savinov, Akihito Seki, L'ubor Ladický, Torsten Sattler, Marc Pollefeys, Quad-networks: unsupervised learning to rank for interest point detection, 2017, http://openaccess.thecvf.com/content_cvpr_2017/papers/Savinov_Quad-Networks_Unsupervised_Learning_CVPR_2017_paper.pdf

[52] Johannes L. Schonberger, Hans Hardmeier, Torsten Sattler, Marc Pollefeys, Comparative Evaluation of Hand-Crafted and Learned Local Features, 2017, https://www.cvg.ethz.ch/research/local-feature-evaluation/schoenberger2017comparative.pdf

[53] Takehito Seki , Yuichi Ikuhara , Naoya Shibata, Theoretical Framework of Statistical Noise in Scanning Transmission Electron Microscopy, 2018, https://www.sciencedirect.com/science/article/abs/pii/S0304399118300603

[54] Rajvi Shah, Aditya Deshpande, P J Narayanan, Multistage SFM: A Coarse-to-Fine Approach for 3D Reconstruction,2016, https://arxiv.org/pdf/1512.06235.pdf

[55] Gregory G. Slabaugh, Computing Euler angles from a rotation matrix, http://www.gregslabaugh.net/publications/euler.pdf

[56] A. P. Tafti, Jessica D. Holz, Ahmadreza Baghaie, Heather A. Owenb, Max M. He, Zeyun Yu, 3DSEM++: Adaptive and intelligent 3D SEM surface reconstruction, 2016, https://www.ncbi.nlm.nih.gov/pubmed/27200484

[57] Ahmad P. Tafti, Kirkpatrick, A.B., Alavi, Z., Owen, H.A. and Yu, Z., 2015, Recent advances in 3D SEM surface reconstruction, http://www.sciencedirect.com/science/article/pii/S0968432815300226

[58] Tinne Tuytelaars and Krystian Mikolajczyk, Local Invariant Feature Detectors: A Survey, 2008, http://homes.esat.kuleuven.be/~tuytelaa/FT_survey_interestpoints08.pdf

[59] Bill Triggs, Philip Mclauchlan, Richard Hartley, Andrew Fitzgibbon, Bundle Adjustment – A Modern Synthesis, 2010, https://hal.inria.fr/inria-00548290/document

[60] Aji Resindra Widya, Akihiko Torii and Masatoshi Okutomi Structure-from-Motion using Dense CNN Features with Keypoint Relocalization, 2018, https://arxiv.org/pdf/1805.03879.pdf

[61] Changchang Wu, VisualSFM, nuoroda į puslapį: http://ccwu.me/vsfm/doc.html

[62] Kieran Wynn, pyquaternion, http://kieranwynn.github.io/pyquaternion

[63] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua. LIFT: Learned Invariant Feature Transform, European Conference on Computer Vision (ECCV), 2016, https://arxiv.org/pdf/1603.09114.pdf

[64] Kwang Moo Yi, Eduard Trulls, Yuki Ono, Vincent Lepetit, Mathieu Salzmann, Pascal Fua, Learning to Find Good Correspondences, 2018, https://arxiv.org/pdf/1711.05971.pdf

[65] Linguang Zhang, Szymon Rusinkiewicz, Learning to Detect Features in Texture Images, 2018, https://gfx.cs.princeton.edu/pubs/Zhang_2018_LTD/keypoint-cvpr.pdf

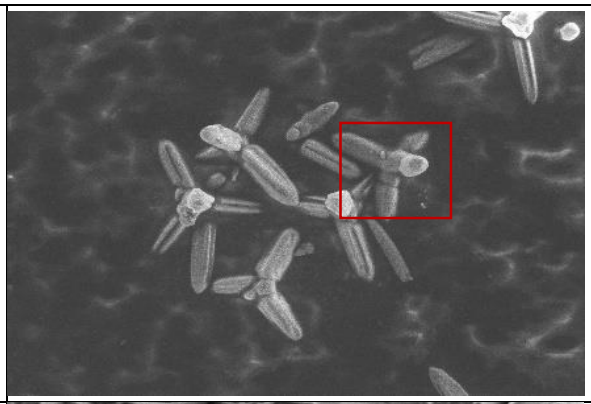
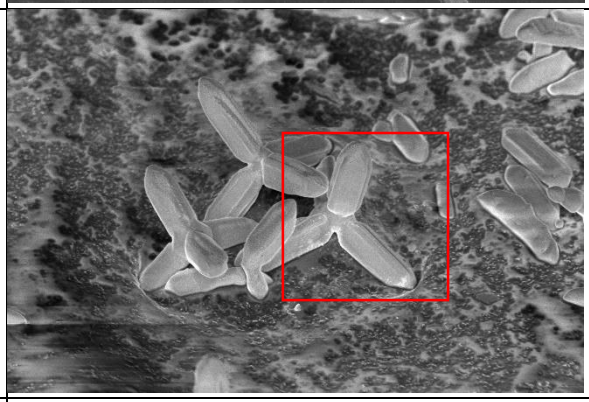
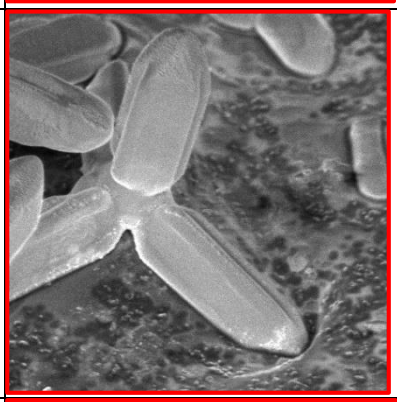
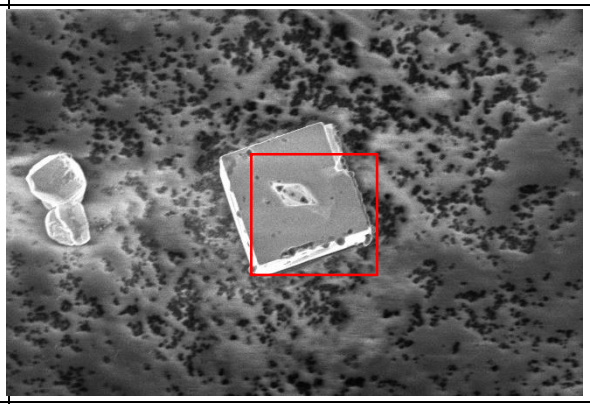
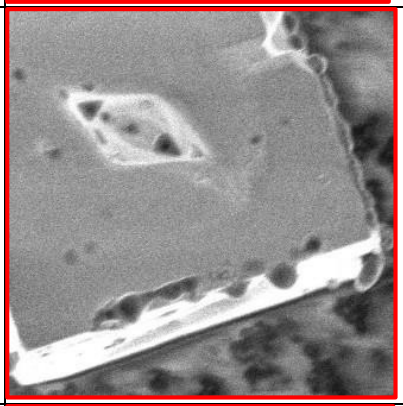
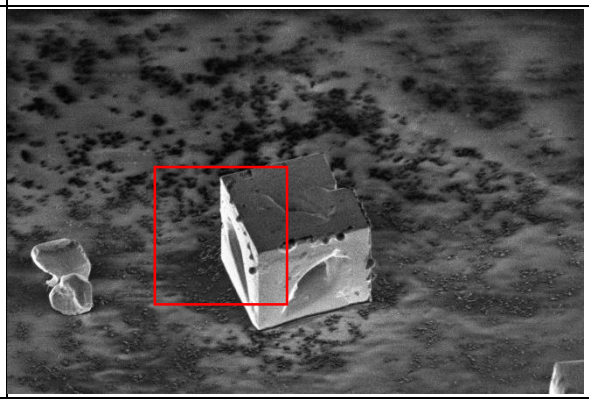
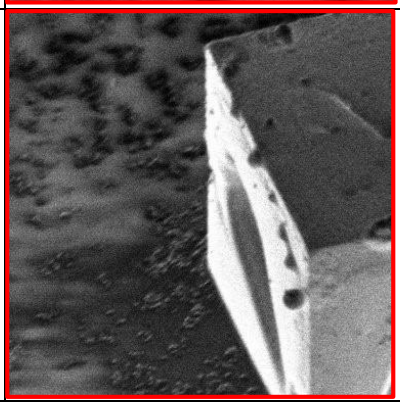
[66] Yide Zhang, Yinhao Zhu, Evan Nichols, Qingfei Wang, Siyuan Zhang, Cody Smith, Scott Howard, A Poisson-Gaussian Denoising Dataset with Real Fluorescence Microscopy Images, 2018, https://arxiv.org/abs/1812.10366

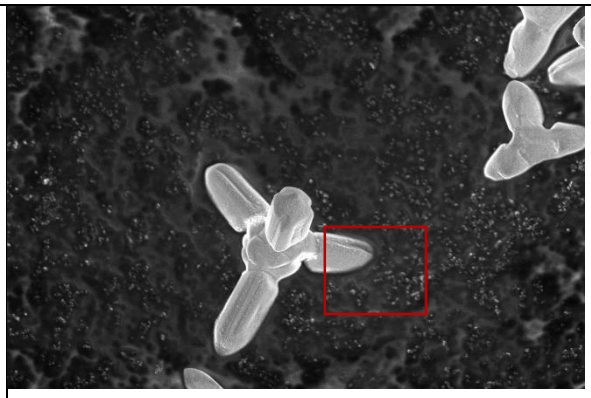
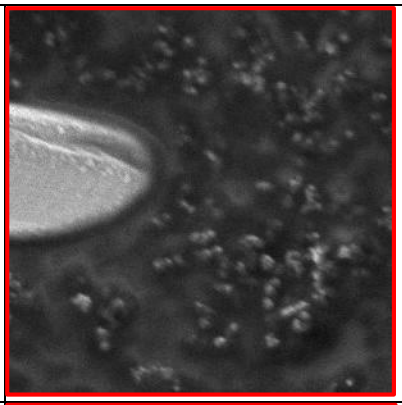
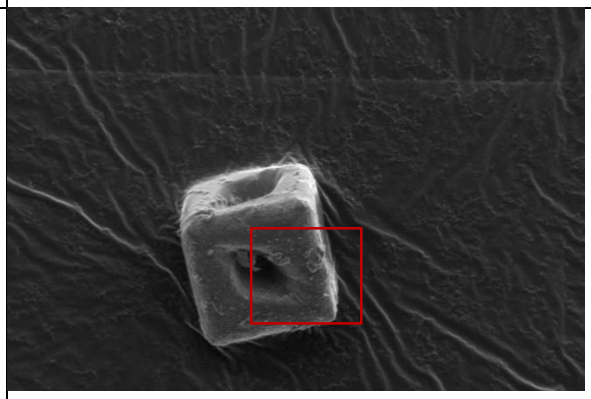
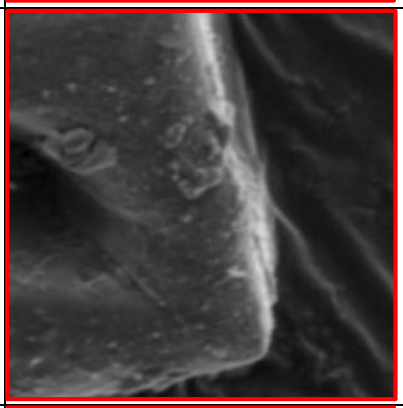
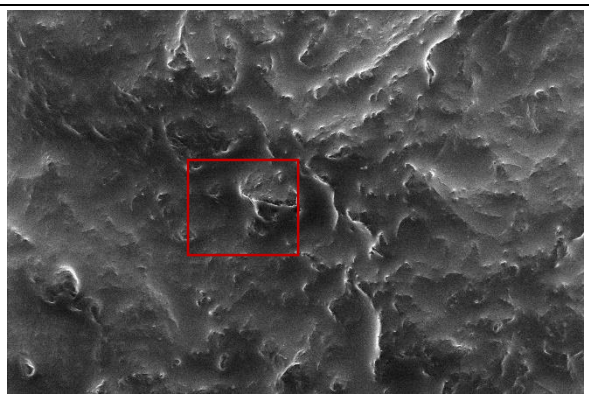
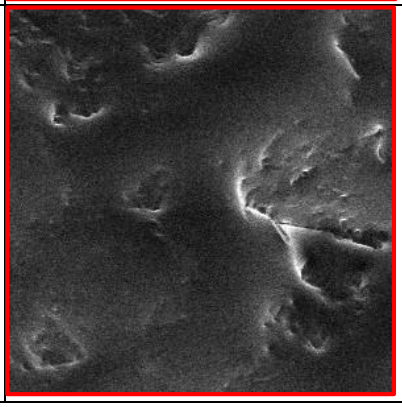
[67] Yasutaka Furukawa, Carlos Hernández: Multi-View Stereo: A Tutorial, 166 pages. Now Publishers Inc, 2015.

Priedai

A. SEM nuotraukos testavimui

1 lentelė. Pateikta keletas SEM nuotraukų rinkinių nuotraukų pavyzdžiai

A.1		
B.1		
C.1		
C.2		

D.1		
E.1		
F.1		

2 lentelė. Pateiktos naudojamos nuotraukos testavimui sudaryti 3D modelius

	Kiekis	X	Y	Z	Rezoliucija	Papildoma informacija
A	13	Visi 0 laipsnių kampu	Visi 0 laipsnių kampu	Kas 5 laipsnių (0 iki 60)	1280x960	Šiame objekte turi daug triukšmo ir miglos padarinių.
B	22	Visi 0 laipsnių kampu	Kas 90 laipsnių (0 iki 90)	Kas 5 laipsnių (0 iki 60)	1536x1024	Šis yra labai paprasto formos, neturi daug artefaktų ar triukšmų ir yra labai gerai atskiriamas nuo pačio fono. Šis objektas pasirinktas, nes kraštinių taškai yra labai panašūs ir tiksliai neatpažintų atitikmenų
C	27	Kas 10 laipsnių (0 iki 30)	Kas 60 laipsnių (30 iki 330)	Visi 5 laipsnių kampu	1536x1024	Ant kubo yra daug ryškių vietų - šviesios horizontalios linijos, triukšmo ir šešėlių, kurios gali pakenkti aptikti atitikmenis
D	37	Kas 5 laipsnių	Kas 60 laipsnių (30 iki 330)	Visi 8 laipsnių kampu	1536x1024	Šis objektas panašus į B dalies objektus, tačiau aplink analizuojamą objektą pateikta daug grubesnis paviršius.
E	37	Kas 10 laipsnių (0 iki 40)	Kas 60 laipsnių (0 iki 300)	Visi 5 laipsnių kampu	1536x1024	Šis objektas turi daug įdubimų, kuriuose yra daug šešėlių, tad būtų labai sunku išgauti daug atitikmenų iš šių nuotraukų, tad tik galėtume panaudoti šių objektų kraštines.
F	30	Kas 60 laipsnių (0 iki 300)	Kas 10 laipsnių (0 iki 40)	Visi 200 laipsnių kampu	1536x1024	Šiame nėra jokių pateiktų išsiskiriančių objektų. Pateiktas tik nelygus paviršius, kuriame yra daug triukšmo.