LATVIJAS UNIVERSITĀTE

DATORIKAS FAKULTĀTE

MĀKOŅU NOŅEMŠANA NO SATELĪTA ATTĒLIEM

MAĢISTRA DARBS

Autors: **Matīss Zērvēns**

Stud. apl. Nr. mz12038

Darba vadītājs: docents Dr. dat. Kārlis Freivalds

RĪGA 2020

ANOTĀCIJA

Jauni satelītu attēli ir praksē noderīgi, tos izmanto dažādās jomās, piemēram, kartogrāfijā. Šajā darbā tiek apskatīti Copernicus Sentinel-2 satelītu uzņemtie Zemes attēli. Tiek izpētīta satelītu programma, rīki un datu formāti. Tiek apskatīti lēmuma koku un Beijesa klasificēšanas algoritmi, kas ļauj izgūt mākoņus un citus reģionus no attēliem, izpētīti attēlu apstrādes algoritmi, kas ļauj saskaņot dažādu attēlu vizuālo izskatu pēc attēla histogrammas datiem.

Darba praktiskajā daļā tiek realizēta sistēma, kas ļauj izgūt lietotājam aktuālāko satelīta attēlu noklājumu pār ģeogrāfisku intereses reģionu. Attēli tiek veidoti kombinējot jaunākos satelīta uzņemtos datus kopā tā, lai gala attēlos nebūtu mākoņi. Tiek parādīti risinājumi kā ģenerētos attēlus pēc tam ir iespējams publicēt citos ĢIS risinājumos, kurus tālāk izmanto gala lietotāji.

Atslēgvārdi – Sentinel-2, mākoņu detektēšana, histogrammu saskaņošana, attēlu apvienošana, ĢIS.

ABSTRACT

Cloud removal from satellite images

Recent satellite images of Earth surface are useful in many practical applications such as cartography. This work focuses on Copernicus Sentinel-2 satellite produced images. Author researches the program, its provided software tools and data format of the satellite data.

Decision tree and Bayes classifiers are investigated that allow cloud and other object classification in satellite images. Also, image processing algorithms based on histogram analysis are looked at which allow color and lighting equalization between similar images.

In practice a system is created which allows user to get newest satellite images covering area of interest where they are combined so that they don’t have clouds in them. Author also provides information and tools which can be used to publish created images in other GIS solutions for end user consumption.

Keywords – Sentinel-2, cloud detection, histogram matching, image combination, GIS.

AUTOREFERĀTS

Tika izpētīti zinātniskajā literatūrā aprakstīti lēmumu koku un Beijesa klasificēšanas algoritmi, kas ļauj izgūt mākoņu un citu reģionu informāciju no satelīta attēliem. Papildus tika apskatīts attēlu histogrammu salīdzināšanas algoritms, kas ļauj izmainīt attēla vizuālo izskatu tuvāk citam attēlam. Liels darbā ieguldītais laiks bija veltīts Sentinel-2 satelītu programmas, tās pieejamo programmatūru un datu formātu izpētei. Autors iemācījās lietot pieejamos REST API, lai izgūtu nepieciešamos satelīta datus, to aprakstošo informāciju. Tika izpētīts datu granulu SAFE glabāšanas formāts, realizēta to nolasīšana un apstrāde autora veidotajā sistēmā.

Apskatītais lēmumu koka algoritms un histogrammu salīdzināšanas algoritms tika realizēts autora paša izveidotā sistēmā, kas apstrādā Sentinel-2 satelītu uzņemtos datus. Autora lielākais pienesums bija izveidot programmu, kas spēj automātiski apstrādāt lietotāja ievadīto ģeogrāfisko intereses reģionu, šim reģionam programma spēj atrast aktuālāko satelīta attēlu apkopojumu, kas nesatur mākoņu reģionus. Autors veica padziļinātu izpēti un pielietoja savu pieredzi darbā ar ĢIS sistēmām, lai izpētītu iespējamus risinājumus, kā noprogrammētās sistēmas rezultātus tālāk ir iespējams izmantot un publicēt citās ĢIS sistēmās. Ģenerētajiem rezultātiem tika veidoti ģeogrāfiskās informācijas piesaistes faili, lai tos tālāk būtu iespējams publicēt citās ĢIS sistēmās. Tika apskatīta problēma, ka attēliem ir atšķirīgas koordinātu sistēmas, tika piedāvāts risinājums pārprojicēt datus vienu sistēmu izmantojot GDAL vai Geoserver rīkus. Praksē tika realizēta WMS servisa izveide no sistēmas ģenerētajiem datiem, tika parādīts kā šos servisus tālāk var izmantot darbvirsmas programmatūrā QGIS un autora realizētajā tīmekļa lietotnes interaktīvajā kartē. Darbā veltītais praktiskais programmēšanas, testēšanas laiks mērāms divās nedēļās.

Daļa no darba rakstiskās daļas tika sākotnēji veikta kursa darba ietvaros. Izpētītā informācija un sastādītais plāns tika izmantots par pamatu praktiskajam darbam maģistra darba ietvaros. Pēc maģistra darba praktiskās daļas izstrādes, esošais teksts tika koriģēts un aktualizēts, jo liela daļa sākotnēji aprakstītā tomēr atšķīrās. Lielākās izmaiņas tika veiktas praktiskās daļas nodaļā, bet tika papildināta un ieviesta jauna informācija arī pārējās darba nodaļās. Pēc maģistra darba rakstiskās daļas izveides, viss darbs tika pārlasīts, izlabotas rakstiskās, stilistiskās kļūdas, papildinātas atsauces.

Pēc risinājuma izveides tika apzinātas problēmas un uzlabojumi, kurus nepieciešams veikt, lai attīstītu darbu nākotnē. Ir nepieciešams veikt attēlu pārprojicēšanu uz vienu sistēmu apstrādes laikā, jo pretējā gadījumā nav iespējams analizēt tādus attēlus, kuri atrodas uz blakus esošu

koordinātu sistēmu robežas. Netika izpētīts līdz galam risinājums par attēla saskaņošanu lielākiem intereses reģioniem, kur nepieciešams saskaņot blakus esošo attēlu fragmentu izskatu. Autoram ir skaidrs darbs ar Sentinel-2 satelītu programmas piedāvātajiem datiem, apgūto informāciju un realizētās risinājuma daļas varētu ieviest autora darba vietā nākotnē, kur uzņēmumam pašam vai tā klientiem šāda veida dati, varētu būt interesanti. Svarīgi ir saprast, ka ieviešot šāda veida risinājumu uzņēmumā pašiem, nav jāmaksā par to papildus izmantošanas maksas kā tas būtu, ja izmantotu kādu citu gatavu risinājumu, kas piedāvā līdzīgā veidā iegūtus datus.

Saturs

APZĪMĒJUMU SARAKSTS 7

IEVADS 9

1. SATELĪTA PROGRAMMAS, DATU PRODUKTI 12

1.1. Sentinel-2 satelītu programma 12

2. PRAKSĒ PIELIETOTIE ALGORITMI 19

2.1. Mākoņu detektēšanas algoritmi 19

2.1.1. Lēmumu koki 19

2.1.2. Klasiskā Beijesa klasificēšana 22

2.2. Kombinētu attēlu vizuālā izskata vienādošanas algoritmi 24

2.2.1. Attēla histogrammas jēdziens 24

2.2.2. Attēla histogrammas saskaņošanas algoritms 25

3. EKSISTĒJOŠI RISINĀJUMI, KAS APSTRĀDĀ SENTINEL-2 DATUS 27

3.1. EOxCloudless 27

3.2. SENTINELHub 28

3.3. Sentinel-2 satelītattēlu pamatkartes mozaīkas serviss 28

4. DARBĀ REALIZĒTAIS RISINĀJUMA APRAKSTS 30

4.1. Kopējās sistēmas vispārīgs apraksts 31

4.2. Sistēmas lietotāja ievaddati 33

4.3. Lietotāja pieprasījumu apstrāde 35

4.4. Intereses reģiona aizpildīšana ar satelīta datu reģioniem 35

4.5. Mākoņu reģionu detektēšana un piefiksēšana 38

4.6. Noklājuma novērtēšana 42

4.7. Attēlu apvienošana 45

4.8. Uzģenerētā attēla periodiska atjaunošana 45

4.9. Satelīta attēlu vizuālā izskata vienādošana 46

4.10. Attēlu publicēšana ĢIS 48

REZULTĀTI 54

SECINĀJUMI 61

IZMANTOTĀ LITERATŪRA UN AVOTI 62

APZĪMĒJUMU SARAKSTS

SAR – radara veids, kas ļauj iegūt zemes virsmas datus.

MSI – vairāku spektru instruments, kas izvietots uz satelīta, lai varētu uzņemt dažādu gaismas spektru datus.

RGB - kubveida krāsu reprezentāciju modelis, kurā kuba augstums, platums un dziļums reprezentē sarkano, zaļo un zilo krāsu (red, green, blue).

Varbūtiskā gradienta lejupeja – iteratīvs funkcijas optimizācijas algoritms.

Atbalsta vektoru klasificēšana – uzraudzītās mašīnmācīšanās algoritms, kas veic bināru objektu klasificēšanu.

Lēmumu koks – klasificēšanas algoritms, kas veic objekta klasificēšanu pēc vairākām pārbaudēm, kas tiek izkārtotas kokveida struktūrā.

Gadījuma lēmuma mežs – ansambļa veida klasificēšanas algoritms, kas veic klasificēšanu balstoties uz lielāko kopīgo balsi, kuru devuši vairāki lēmumu koki.

Beijesa klasifikatori – varbūtiski klasificēšanas algoritmi, kas balstīti uz Beijesa teorēmu.

Attēla histogramma – attēla pikseļu vērtību statistisks izkārtojums.

Spektrs – elektromagnētiskā lauka specifiskas frekvence vai frekvenču intervāls.

Granula – Sentinel-2 satelīta uzņemto datu mazākā datu vienība.

Intereses reģions – izveidotās sistēmas lietotāja ievadītais ģeogrāfiskais reģions.

API – programmatūra, kura nodrošina konkrētu funkcionalitāti, kuru var izmantot citu programmatūru realizācijā, kurām nepieciešama API nodrošinātā funkcionalitāte.

Copernicus programma – Eiropas savienības veidota Zemes virsmas novērošanas programma.

Sentinel – Copernicus programmas satelīts.

L1C – Sentinel-2 satelīta attēlu produkts, kas satur augšējās atmosfēras atstarojošo informāciju.

L2A – Sentinel-2 satelīta attēlu produkts, kas satur zemāko atmosfēras atstarojošo informāciju.

SAFE formāts – Sentinel uzņemto datu apmaiņas formāts, kas satur uzņemto spektru attēlu datus un aprakstošo metadatu informāciju XML formātā.

JPEG2000 – attēla informācijas glabāšanas formāts, kuru izmanto Sentinel-2 spektru glabāšanai.

JPEG75 – attēla informācijas glabāšanas formāts, kuram izmantota kompresija, kas zaudē daļu no sākotnējās informācijas.

GeoTIFF – ir attēla informācijas glabāšanas formāts, kurš satur papildus galveni, kurā glabā ģeogrāfisko informāciju par reģionu, kuru attēls reprezentē.

GIS jeb ĢIS – ģeogrāfiskās informācijas sistēmas, kas ir informācijas sistēmas, kas apstrādā ģeotelpiskus datus.

Ģeoreferencēts – datiem ir pievienota papildus informācija, kas definē datu ģeogrāfisko atrašanās vietu.

PB – peta baiti.

SQL – programmēšanas valoda, kas paredzēta datu ieguvei un manipulācijai no strukturētām datubāzēm.

HTTP – tīkla datu apmaiņas protokols.

HTTP POST – HTTP protokola metode, kas paredzēta, lai nosūtīt datus uz serveri, lai izveidotu vai manipulētu objekta datus.

HTTP GET – HTTP protokola metode, kas paredzēta, lai izgūtu datus no konkrēta tīmekļa resursa.

REST – tīmekļa servisu izveides standarts.

OGC – atvērto ĢIS tehnoloģiju standartu organizācija, kas veido un uztur ĢIS tehnoloģiju standartus.

WMS – tīmekļa karšu servisa standarts, kuru izveidojusi OGC.

WFS – tīmekļa objektu serviss, kas paredzēts ģeotelpisko datu, atribūtu ieguvi teksta formātā.

WMTS – tīmekļa karšu servisa standarts, strādā tikai ar ģenerēta attēlu keša datiem atšķirībā no WMS, kas ģenerē attēlus no datiem pēc pieprasījuma.

WKT – atvērts ģeotelpisko datu formāts, kurā iespējams glabāt vektora veida.

IEVADS

Mūsdienās Zemes satelīta attēli tiek plaši izmantoti daudzās nozarēs un lietojumos. Viens no satelīta attēlu izplatītākajiem lietojumiem ir tas, ka tie tiek izmantoti populārās tīmekļa lietotnēs un viedtālruņu lietotnēs. Tādas lietotnes kā Google maps, Apple maps ir gandrīz jebkuram cilvēkam, kuram ir viedtālrunis, pazīstamas. Šajās lietotnēs satelīta attēli tiek izmantoti, lai veidotu interaktīvu karti, kuru lietotājs var kustināt horizontāli un vertikāli, pietuvināt un attālināt. Lietotāji šāda veida kartes izmanto savos ikdienas darbos, piemēram, lai navigētu telpā, atrastu konkrētas vietas u.c.

Satelīta attēli, no tiem veidotās kartes, tiek aktīvi izmantotas arī daudzos citos specializētos lietojumos, atšķirīgās nozarēs, piemēram, tās izmanto mežsaimniecībā, lai veiktu mežu pārvaldību, veiktu koku sugu uzskaiti, detektētu nelegālu mežu izciršanu un daudzās citās nozarē svarīgās darbībās. Lauksaimniecībā šie dati var tikt izmantoti, lai varētu prognozēt gaidāmo ražu [1]. Kartogrāfijā satelīta attēli var tikt izmantoti, piemēram, lai detektētu nesenās izmaiņas apkārtējā vidē, kuras pēc tam pārzīmē citos datos, piemēram, vektoru kartēs. Šos datus arī izmanto dabas katastrofu un negadījumu seku mazināšanā, piemēram, plūdu, zemestrīču, ugunsgrēku gadījumos.

Visos augstāk minētajos lietojumos, Zemes satelītu datu uzņēmumi ir aktuāli un pielietojami tikai tad, ja dati ir pietiekoši nesen uzņemti. To pievienotā vērtība ātri sarūk daudzos no lietojumiem līdz ar to vecuma pieaugšanu. Lai risinātu šo problēmu, pēdējo gadu laikā ir nozīmīgi palielinājušies pieejamie satelīta dati, kurus nodrošina Amerikāņu un Eiropas satelītu programmas, kas speciāli veidotas, lai nodrošinātu regulāri atjaunotus satelītu uzņēmumus bez maksas speciālistiem no augstāk minētajām nozarēm [4, 5].

Augstāk minētās satelītu programmas iegūst visas Zemes satelīta attēlus dažu dienu laikā, iegūtie dati tiek uzkrāti, piemēram, Eiropas satelītu programma dažu gadu laikā jau uzkrājusi vairāk kā 25 PB datus [1]. Diemžēl, liela daļa no uzņemtajiem datiem nav pilnvērtīgi izmantojami, jo datu uzņemšanas laikā ir bijis mākoņu segas pārklājums virs reģiona, pār kuru gāja satelīts. Aptuveni divas trešdaļas no ikgadējiem uzņēmumiem, jeb ~ 66% ir pilnīgi vai daļēji ar mākoņu pārklājumu. Tā kā svarīgi ir, galvenokārt, tikai dati, kur nav bijuši mākoņi, tad ir nepieciešams izgūt no datiem tikai reģionus, kur nav mākoņu pārklājums. Pie tik regulāriem datu atjaunojumiem, nav praktiski, ka datu izgūšanu veic ar manuālu darbu, tāpēc ir svarīgas automātiskas metodes, algoritmi, kas spēj izgūt nepieciešamos reģionus cilvēku vietā.

Ir izpētīti un izstrādāti daudzi dažādi algoritmi un pieejas, kas tiek izmantotas, lai automātiski spētu izgūt no satelīta datiem tikai derīgās datu daļas. Algoritmi, galvenokārt, darbojas tā, ka tie datu reģionu sadala vairākās daļās, kur kāda no datu daļām ir tā saucamās skaidras debess reģioni, bet citi ir, piemēram, mākoņu klāti reģioni, sniega klāti reģioni, ūdens reģioni un citi. Lai gan algoritmu princips ir līdzīgs, praksē izmantotās pieejas nāk no ļoti daudzām atšķirīgām algoritmu, datorzinātnes nozarēm. Problēmas risināšanā tiek izmantoti algoritmi no datizracē bieži pielietotām metodēm, piemēram, klasiskā Beijesa metode [2] un lēmuma koki. Mūsdienās, viena no modernākajām pieejām ir izmantot dziļos neironu tīklus, lai risinātu problēmu, kas ir tikai viena no ļoti daudzām sarežģītajām problēmām, kuras tagad risina ar šīs metodes palīdzību. Tiek pielietotas arī daudzas citas mašīnmācīšanās metodes, lai risinātu mākoņu detektēšanu satelīta attēlos, ir iespējams pielietot tādas metodes kā varbūtiskā gradienta lejupeja (stochastic gradient descent), atbalsta vektoru klasificēšana, gadījuma lēmuma mežs (random forest) [3].

Bez mākoņu detektēšanas problēmas, darbā tiek apskatīta attēlu vizuālā izskata vienādošana. Satelīta dati tiek iegūti nemitīgi pa visu pasaules teritoriju īsā laika periodā. Satelītam kustoties lielā ātrumā virs Zemes, tiek uzņemti daudzi kvadrātveida datu reģioni, starp reģioniem var būt mainīgi apgaismojumi, var būt atšķirīgi ietekmējoši faktori, kas izmaina rezultējošos datus, piemēram, putekļi, tvaiks un citas parādības [6]. Ir nepieciešams pielietot attēlu datu normalizējošus algoritmus un metodes, lai apvienotie dati no daudzajiem datu reģioniem kopā izskatītos konsekventi. Lai risinātu šo problēmu, var pielietot vairākus attēlu apstrādes algoritmus, piemēram, uz attēla histogrammas analīzes balstītus algoritmus [61, 62].

Šī darba ietvaros tiek apskatīti pieejamie brīvpieejas satelīta datu servisa piegādātāji, veikta pieejamo datu un metadatu izpēte. Darbā primāri tiek apskatīta Eiropas Copernicus Sentinel-2 satelītu programma, to rezultātu datu produkti. Pēc tam tiek apskatīta esošā situācija zinātniskajā literatūrā un tīmekļa resursos, kas apraksta pielietojamās metodes mākoņu vai līdzīgu datu reģionu identificēšanu satelīta datos. Tad tiek apskatīta esošā situācija zinātniskajā literatūrā, kur apskatītas attēlu apstrādes metodes, kas risina attēlu vienādošanas problēmu pie dažādiem apgaismojumiem un citiem ietekmējošies faktoriem, kas izmaina gala satelīta attēlu. Pēc esošās izpētes apskatīšanas un izzināšanas, autors pielieto daļu no metodēm praksē, tiek izveidots risinājums, kas spēj automātiski izgūt satelīta datus no datu piegādātāja, datiem tiek pielietota realizēta metode, kas spēj izņemt no attēla mākoņiem klātos reģionus. Risinājumā tiek realizēti algoritmi, kas ļaus apvienot vairākus satelīta datu reģionus kopā, trūkstošos datu reģionus, kur bija mākoņi, risinājums

aizvieto ar vecākiem fragmentiem, kuros nav bijis mākoņu pārklājums. Tiek izpētīts, kā realizētās sistēmas ģenerētos attēlus pēc tam ir iespējams izmantot praktiski, piemēram, publicējot tos ar tīmekļa karšu servisa (WMS) [13] palīdzību, ģenerējot failiem ģeogrāfiskās informācijas piesaistes datus, lai tos varētu iesaukt dažādās ģeotelpiskās informācijas sistēmās (ĢIS) [14].

Darbs strukturēts četrās galvenajās daļās. Pirmajā daļā tiek apskatīti eksistējošie brīvpieejas satelīta datu piegādātāji, pieejamo datu, metadatu specifika, formāti. Otrajā daļā tiek apskatīti vairāki literatūrā aprakstītie algoritmi, kurus praksē pielieto, lai risinātu darbā pētītās problēmas. Trešajā daļā tiek apskatīti līdzīgi eksistējoši risinājumi šī darba ietvaros realizētajai sistēmai, to nodrošinātā funkcionalitāte. Ceturtajā daļā tiek sniegts detāls apraksts par autora praksē realizēto risinājumu, kas ļauj lietotājam izgūt pašu aktuālāko satelīta attēlu kombināciju, kas veidota tā, lai attēlos nebūtu mākoņu pārklājuma. Tiek sniegts apraksts par sistēmas veidotajām datu publicēšanas un integrēšanas iespējām citās ĢIS sistēmās.

1. SATELĪTA PROGRAMMAS, DATU PRODUKTI

Šajā nodaļā tiek apskatītas satelītu programmas, to piedāvātie servisi, programmatūra un datu produkti, kurus lietotāji var izmantot bez maksas. Darbā, galvenokārt, koncentrēsies uz Eiropas Savienības Copernicus programmas [20] Sentinel-2 misiju, kas tiek aprakstīta detālāk nodaļas nākamajā sadaļā.

1.1.Sentinel-2 satelītu programma

Šajā sadaļā tiek detāli apskatīta Eiropas Savienības Copernicus programmas Sentinel-2 misija, tās ģenerētie datu produkti un programmatūra, kas izmantojama, lai tos iegūtu tālākai apstrādei un analīzei.

Copernicus programma ir Zemes novērošanas programma, kuras mērķis ir nodrošināt tuvu reāllaika Zemes novērošanu, lai nodrošinātu dažāda veida datus, kas izmantojami dažādās nozarēs, piemēram, klimata pārmaiņu analīzei, dabas katastrofu seku mazināšanai un novēršanai. Programmu realizē Eiropas Savienības dalībvalstis ar mērķi nodrošināt tās pilsoņiem brīvu pieeju datiem un servisiem, ko programma nodrošina. Programmas mērķis ir nodrošināt gan zinātnes, gan industrijas attīstību. [21]

Programmas datus iegūst satelīti, kurus sauc par Sentinel. Tie riņķo Zemes orbītā un veic Zemes novērojumus ar speciālu instrumentu palīdzību, kas izvietoti uz tiem. Programmas pirmais satelīts Sentinel-1A tika palaists orbītā 2014. gadā. Nākamo 10 gadu laikā, programmas plānā ir palaist vēl vairāk kā 10 satelītus, kuri veidos satelītu konstelāciju, kas piegādās bieži uzņemtus, augstas kvalitātes datus par Zemi. Satelītu iegūtie dati tiek piedāvāti šādu nozaru lietojumiem:

- Atmosfēras monitorēšanai;
- Jūras floras monitorēšanai;
- Zemes monitorēšanai;
- Klimata pārmaiņu monitorēšanai;
- Ārkārtas situāciju risināšanai;
- Drošībai.

Pašlaik orbītā ir trīs veidu satelīti: Sentinel-1, Sentinel-2 un Sentinel-3 satelīti. Katrs no tiem iegūst citādāka veida datus, kas pielietojami dažādiem lietojumiem.

Sentinel-1 satelīti ir pirmie, kas tika palaisti orbītā. Tie ir aprīkoti ar SAR [22] tipa radara instrumentiem, kas ļauj iegūt Zemes virsmas attēlus neatkarīgi no laikapstākļiem un ārējā

apgaismojuma. Dati pielietojami dažādiem lietojumiem, piemēram, zemes, jūras floras monitorēšanai.

Sentinel-3 satelīti ir programmas visnesenāk palaistie satelīti, kuri aprīkoti vairākiem sensoriem, kas paredzēti plašam klāstam dažāda veida datu ieguvei, kas pielietojamie, piemēram, ūdens un zemes teritoriju virsmu temperatūru noteikšanai. Vairāku spektru instruments, kas paredzēts ūdens tvaika, hlorofila absorbcijas noteikšanai u.c. [24]

Sentinel-2 satelīti, kuru produkti interesē šī darba ietvaros, ir aprīkoti ar MSI instrumentiem, kas iegūst 13 dažādu spektru datus no Zemes virsmas. Šos uzņemtos datus ietekmē gan ārējais apgaismojums, gan citi faktori, piemēram, atmosfēras parādības kā piemēram mākoņi. Zemāk esošajā tabulā ir apkopoti visi 13 satelīta instrumenta uzņemtie spektri.

	S2A		S2B		
Band Number	Central wavelength (nm)	Bandwidth (nm)	Central wavelength (nm)	Bandwidth (nm)	Spatial resolution (m)
1	442.7	21	442.3	21	60
2	492.4	66	492.1	66	10
3	559.8	36	559.0	36	10
4	664.6	31	665.0	31	10
5	704.1	15	703.8	16	20
6	740.5	15	739.1	15	20
7	782.8	20	779.7	20	20
8	832.8	106	833.0	106	10
8a	864.7	21	864.0	22	20
9	945.1	20	943.2	21	60
10	1373.5	31	1376.9	30	60
11	1613.7	91	1610.4	94	20
12	2202.4	175	2185.7	185	20

1.1. attēls ***Sentinel-2 divu satelītu konstelācijas instrumentu uzņemto spektru specifikācija [23]***

Kā redzams tabulā, tad abi satelīti uzņem gandrīz identisku spektru datus, starp tiem ir nelielas atšķirības, kas izskaidrojamas ar nelielām variācijām satelīta materiālos. Tabulā ir arī norādīta datu izšķirtspēja, kurā konkrētā spektra dati ir pieejami. Jo mazāks izšķirtspējas skaitlis, jo detālāki ir konkrētā spektra dati. Sentinel-2 13 datu spektri ir sadalīti pa 3 izšķirtspējām: 10m, 20m un 60m. Izšķirtspēja nozīmē, ka tā ir mazākā vienība kuru ir iespējams izšķirt datos, piemēram, 20m izšķirtspējas gadījumā, vienā pikselī ir uzņemta 20m reālās pasaules informācija.

Katram no uzņemtajam datu spektram ir savs lietojums, tabulā zemāk ir skaidrojums kāda veida dati ir konkrētajā spektrā.

Sentinel-2 bands
Band 1 – Coastal aerosol
Band 2 – Blue
Band 3 – Green
Band 4 – Red
Band 5 – Vegetation red edge
Band 6 – Vegetation red edge
Band 7 – Vegetation red edge
Band 8 – NIR
Band 8A – Narrow NIR
Band 9 – Water vapour
Band 10 – SWIR – Cirrus
Band 11 – SWIR
Band 12 – SWIR

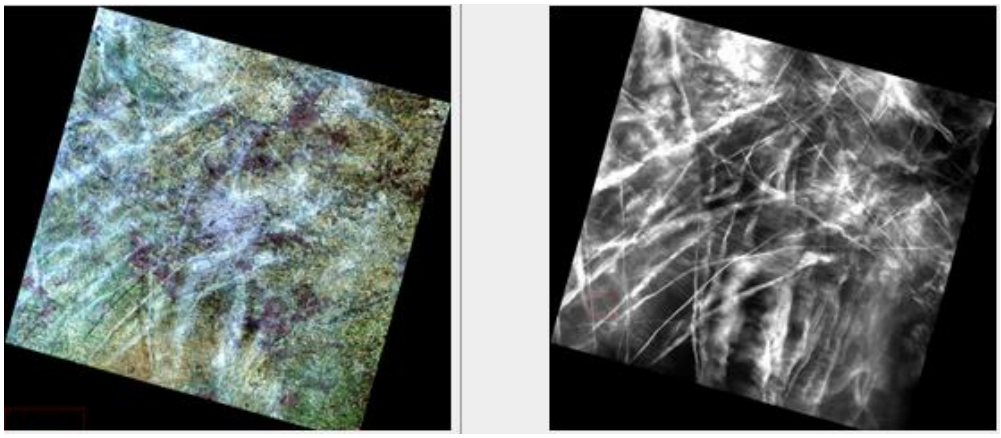
1.2. attēls **Sentinel-2 satelītu datu spektru skaidrojumi [11]**

Kā tabulā redzams, tad spektri Band 2 (tālāk darbā spektri tiks saīsināti Bn, kur n ir spektra kārtas numurs), B3 un B4 ir redzamās gaismas veidojošie spektri. Šo trīs spektru salikums būs darbā interesējošie attēli. Lai risinātu darbā apskatīto problēmu - mākoņu detektēšanu - tiks izmantoti ne tikai šo trīs spektru informācija, bet arī daudzi citi no šiem pieejamiem spektriem, piemēram, B10 spektrs ir svarīgs spektrs šai problēmai, kas ir specializēts, lai viegli detektētu specifiska tipa mākoņus, kurus sauc par spalvmākoņiem [25]. Spalvmākoņa attēla paraugs ir redzams attēlā zemāk.

1.3. attēls ***Spalvmākoņu piemērs[26]***

Spalvmākoņi veidojas augstu atmosfērā, no aptuveni 6000m, tie veidoti no smalkiem ledus kristāliem, kas veidojušies sasalstot ūdens pilieniem. Sentinel-2 B10 spektrs ir speciāli veidots tā,

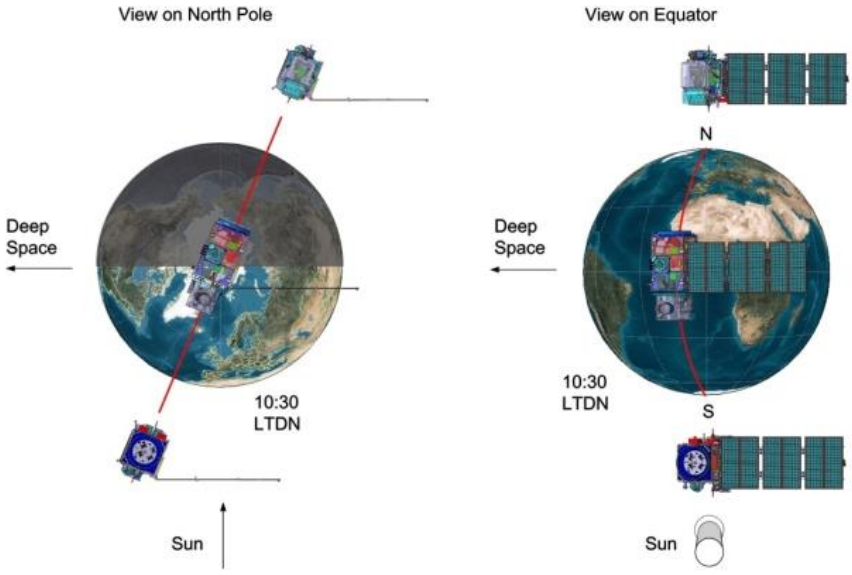
ka tas detektē datus augstu atmosfērā, kur veidojas šie spalvmākoņi, izmantojot šo spektru ir relatīvi viegli detektēt šī veida mākoņus [27]. Attēlos zemāk ir parādīts RGB attēls un šīm attēlam atbilstošais B10 spektra uzņēmums, dati iegūti no Landsat 8 satelīta programmas.



1.4. attēls **Landsat 8 attēlu paraugs. Kreisais attēls ir RGB spektru attēls, bet labais attēls ir B10 spektra dati[27]**

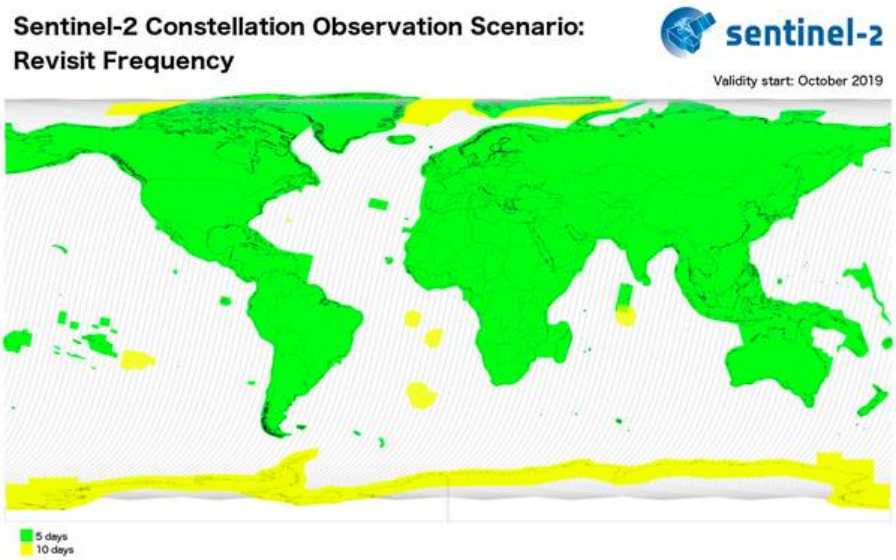
Kā redzams, tad ar šī spektra palīdzību ir ļoti viegli atrast konkrētā tipa mākoņus. Diemžēl, daudzi cita veida mākoņi neatrodas šajā augstumā, tie ir tuvāk Zemes virsmai, lai tos detektētu jāizmanto citi datu spektri un advancētāki algoritmi, kas tos analizē.

Sentinel-2 satelītu konstelācija veidota no diviem satelītiem, kas orbitē Zemi sinhronizēti ar Saules apgaismoto Zemes daļu. Abi satelīti orbitē vienā un tai pašā polārajā orbītā, bet tie ir novirzīti pa 180 grādiem viens no otra. Satelītu orbītas konfigurācija redzama attēlā zemāk.



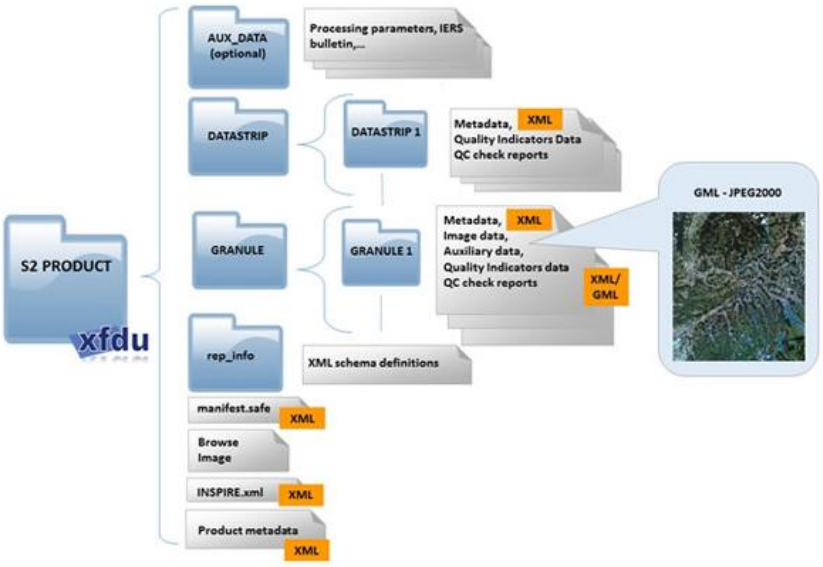
1.5. attēls **Sentinel-2 konstelācijas izvietojums [29]**

Satelīti kopējiem spēkiem darbojoties panāk, ka katra teritorija tiek apsekota ne ilgāk kā ik pēc 5 dienām. Zemāk esošajā attēlā tiek ilustrēts Sentinel-2 datu atjaunošanas biežums pasaules teritorijā.



1.6. attēls **Sentinel-2 datu atjaunošanas biežums pasaules teritorijā [10]**

Sentinel-2 satelītam kustoties polārā orbītā [28], uzņem datus pa joslu. Šīs joslas datus sadala 100x100 km lielos kvadrātos, kurus sauc par granulām. Šis granulas ir mazākā datu vienība, kas iegūstama no satelītu programmas datu izgūšanas programmām, datu servisiem. Dati ir saglabāti programmas SAFE formātā. Sentinel-2 datu formāta ilustrācija pieejama attēlā zemāk.



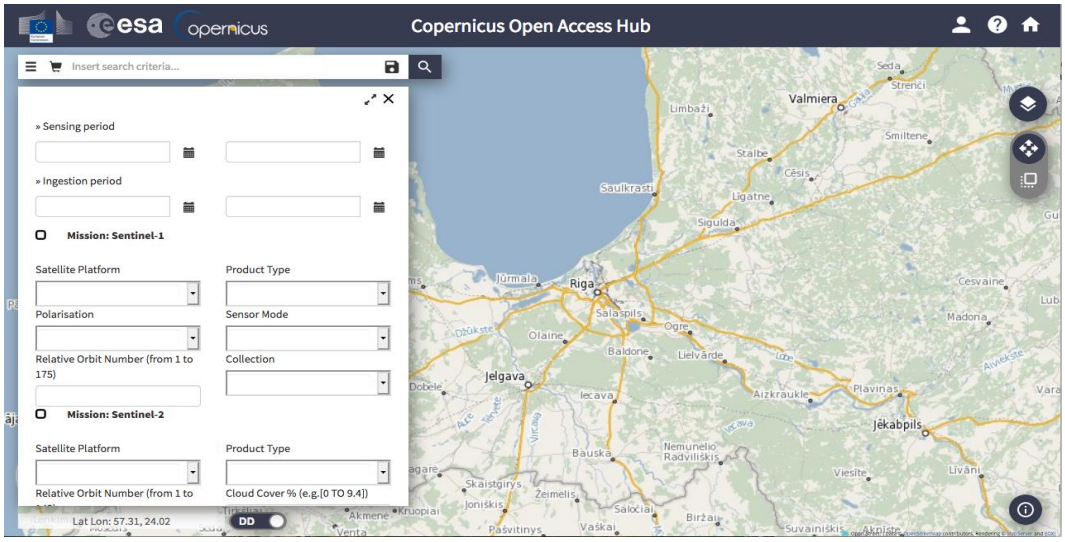
1.7. attēls **Sentinel-2 granulas datu formāta ilustrācija [30]**

Kā redzams, ir pieejams plašs klāsts gan datu, gan metadatu katrai datu granulai. Paši 13 spektru dati sadalīti katrs savā JPEG2000 attēla formātā. Ir divu veidu granulas, granula, kas satur augšējās atmosfēras datus, bet otra veida satur detālāku apakšējās atmosfēras informāciju. Pirmā veida granula aizņem aptuveni 500MB, bet otrā veida 800MB. Attēlā zemāk ir 10m izšķirtspējas granulas B2, B3, B4 spektru apvienotais attēls, kas uzņemts 2019. gada 3. augustā virs Rīgas.

1.8. attēls ***Sentinel-2 granulas RGB spektru attēla piemērs virs Rīgas 2019. gada augustā***

Augstāk redzamais granulas attēls un visi pārējie granulas dati tika iegūti izmantojot tīmekļa lietotni, kuru nodrošina satelītu programma.

Viens no satelīta datu izguves veidiem ir izmantojot grafisku tīmekļa lietotni, kas saucas Open Hub. Lai to varētu pilnvērtīgi lietot, lietotājam ir jāpiereģistrējas Copernicus Open Access Hub portālā. Reģistrēšanās ir bez maksas, pēc tam izmantojot reģistrēto lietotāju ir pieeja pie visiem programmas datiem bez maksas [31]. Grafiskās saskarnes Open Hub piemērs ir redzams attēlā zemāk.



1.9. attēls **Copernicus Open Hub grafiskās lietotnes piemērs**

Granulu datus var filtrēt izmantojot karti, kur var atlikt intereses reģionu, kreisajā izvēlnē papildus jānorāda filtrēšanas parametri, piemēram, kuru satelītu dati interesē, datu ieguves laika intervālu.

Otrs veids ir izmantojot API Hub, kas ir bez grafiskās saskarnes, tas ir paredzēts, galvenokārt programmātiskai datu izgūšanai, attiecīgi, to izmantos darbā realizētais gala risinājums.

2. PRAKSĒ PIELIETOTIE ALGORITMI

Šajā nodaļa tiek apskatīti pašlaik literatūrā dokumentētie algoritmi un metodes, kas risina kādu no darbā pētītajām problēmām

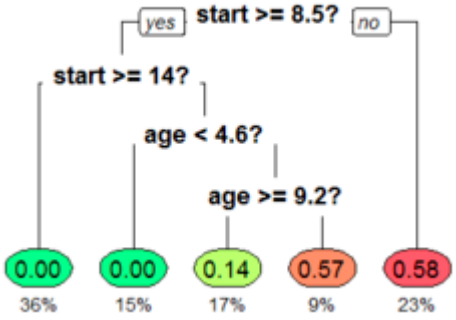
2.1.Mākoņu detektēšanas algoritmi

Šajā sadaļā tiek apskatīti literatūrā aprakstītie algoritmi, kas risina darbā svarīgāko problēmu, kas ir mākoņu detektēšana satelīta attēlā.

2.1.1. Lēmumu koki

Šajā sadaļā tiek apskatīts datizracē pielietots algoritms, kuru sauc par lēmuma koku. Šis ir relatīvi viegli saprotams algoritms.

Lēmuma koka ideja ir tāda, ka tiek uzbūvēti nosacījumi vairākos līmeņos, katrā līmenī ir fiksēts skaits ar nosacījumiem pār ievaddatu konkrētiem atribūtu kopu datiem. Kokam nultais ir sākuma līmenis, bet pēdējais ir beigu līmenis. Algoritms klasificē objektus balstoties uz to parametriem ejot no nulltā līmeņa cauri katram tālākam līmenim līdz nonāk līdz pēdējam līmenim, kurā tiek veikta gala klasifikācija. Attēlā zemāk ir sniegts vienkārša koka piemērs.



2.1. attēls **Lēmuma koka piemērs, kas nosaka konkrētas slimības iestāšanās varbūtību pēc operācijas [33]**

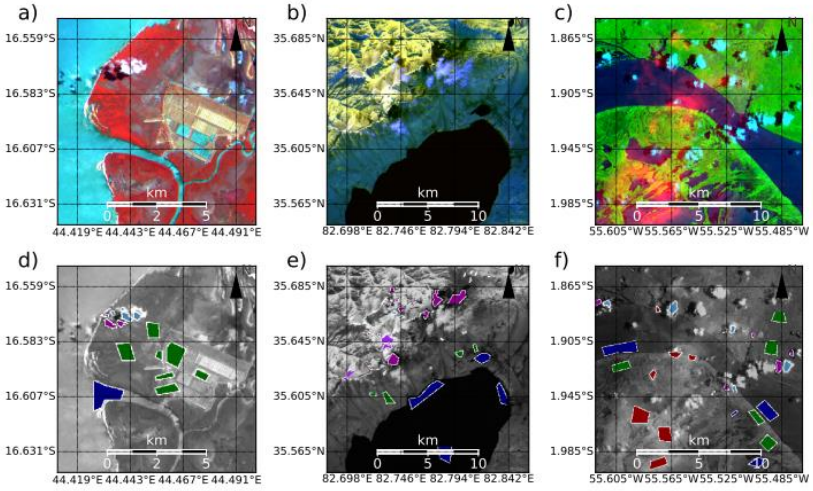
Attēlā ilustrēts lēmuma koks, kas balstoties uz pacienta informāciju 'start' un 'age' klasificē to konkrētā varbūtības klasē.

Ar klasificēšanu saprot, ka konkrētais objekts, balstoties uz tā aprakstošo informāciju, tiek ielikts kaut kādā kategorijā. Konkrēts lēmuma koks ir viens konkrēts lēmuma koka algoritms. Paša lēmuma koka būve ir iespējama dažādi, piemēram, to var sastādīt cilvēks manuāli, balstoties uz pieredzi un zināšanām par konkrēto problēmas reģionu. Eksistē arī algoritmi, kas spēj koku

uzbūvēt automātiski, ja ir pieejami marķēti mācīšanās dati. Šie koku būvēšanas algoritmi ir mašīnmācīšanās algoritmi, kas ir vadītās mašīnmācīšanas apakšnozares algoritmi. Šādus algoritmus apmāca ar datiem, kādi varētu būs praksē jāklasificē. Datiem papildus nepieciešams, lai ir pieejama pareizās klasificēšanas informācija. Daži no populārākajiem koku būvēšanas algoritmiem ir C4.5 algoritms un CART algoritms, kurus var izmantot automātiskai koku būvēšanai [32].

Mākoņu detektēšanas problēmas risināšanai ir nepieciešams sastādīt vai kaut kur izgūt marķētus treniņa kopas datus, kurus izmantot, lai varētu automātiski izveidot lēmuma koku. Jāņem gan vērā, ka lielākajai daļai darbā apskatīto algoritmu būs nepieciešamība pēc šādiem datiem, lai tos spētu apmācīt pirms lietošanas reālo problēmu risināšanai.

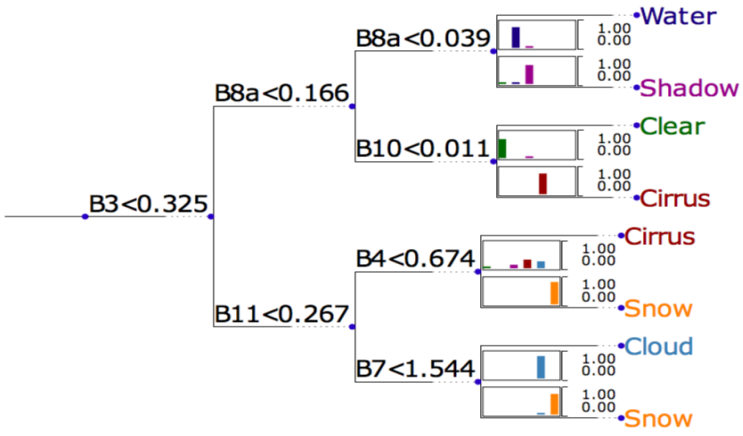
Saistībā ar lēmumu kokiem, tika izpētīts zinātniskais raksts [3], kurā autori paši veidoja savu treniņu datu kopu no Sentinel-2 pieejamām datu granulām, kas iegūtas dažādos Zemes reģionos, lai iegūtu pietiekoši dažāda veida datus, kas reprezentētu labu izlases kopu no visas Zemes teritorijas. Attēlā zemāk ir redzams paraugs kā tika šajā rakstā sagatavoti treniņa kopas dati.



2.2. attēls **Rakstā apskatīto treniņa granulu reģionu klasificēšana. Apskatītajā rakstā tika pētīta ne tikai mākoņu detektēšana, bet arī cita veida reģionu noteikšana, piemēram, sniegs, ūdens, ēnas [3]**

Kā redzams, tad augstāk parādītajā attēlā ir manuāli ar roku ievilkti poligoni, kas identificē specifisku intereses reģionu klases. Visi ievilktā poligona pikseļiem tiek piešķirta konkrēta klase, piemēram, mākoņu klase. Šos datus pēc tam var izmantot, lai apmācītu algoritmus.

Apskatītajā rakstā gala rezultātā ieguva šādu lēmuma koku, kurš ilustrēts attēlā zemāk.



2.3. attēls **Viens no rakstā iegūtajiem lēmumu kokiem, kas spēj klasificēt ūdens, ēnas, spalvmākoņu, mākoņu un sniega reģionus**

Augstāk redzamais lēmuma koks spēja detektēt minētos reģionus ar 87% procentu precizitāti. Lēmuma kokā vienādojumos Bx apzīmē konkrētās granulas x spektru, bet vērtība ir konkrētā pikseļa vērtība šajā spektrā. Kā redzams, tad šajā kokā tika izmantoti spektri B3, B4, B7 B8a, B11. Ja interesētu, tikai daļa no klašu klasificēšanām, tad varētu samazināt izmantojamo spektru skaitu.

Rakstā tika novērots, ka netika iegūti īpaši labāki rezultāti ar sarežģītākiem kokiem, kuru dziļums ir lielāks par 4. Koki kļuva pārlieku sarežģīti, precizitāti minimāli uzlabojās, sāka palielināties risks, ka iegūtais koks cietīs no pārmācīšanās problēmas [34].

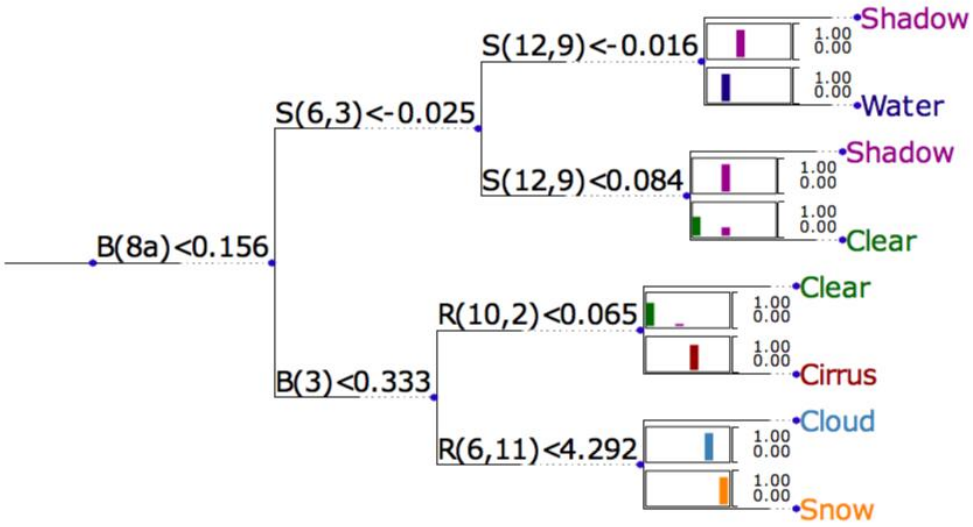
Interesanti, ka rakstā tika apskatīta iespēja ne tikai izmantot pieejamo spektru informāciju, lai klasificētu datus, bet tika apskatīta iespēja izmantot atvasinātus datus, kas iegūti veicot dažāda veida operācijas ar pieejamo spektru informāciju. Attēlā zemāk ir sniegts paraugs ar operācijām, kādas tika izmantotas apskatītajā rakstā.

Table 2. Band math formulas used for the construction of feature spaces. Names and abbreviations are used throughout this paper.

Name	Short Name	Formula
band	B	$f(a) = a$
difference	S	$f(a,b) = a - b$
ratio	R	$f(a,b) = a/b$
depth	D	$f(a,b,c) = \frac{a+b}{c}$
index	I	$f(a,b) = \frac{a-b}{a+b}$
index_{-}^{F}	$I+$	$f(a,b,c,d) = \frac{a-b}{c-d}$
index_{+}^{F}	$I-$	$f(a,b,c,d) = \frac{a+b}{c+d}$

2.4. attēls **Rakstā izmantotās operācijas ar spektru datiem, kas izveido jaunus datus, kurus var izmantot analīzei [3]**

Izmantojot augstāk redzamās operācijas, tika sasniegti nedaudz labāki rezultāti klasificēšanā ar lēmumu kokiem, kas klasificē pēc atvasinātajiem datiem. Attēlā zemāk ir piemērs ar lēmuma koku dziļumā 3, kur tiek izmantoti atvasinātie dati.



2.5. attēls **Rakstā izveidotais klasificēšanas lēmuma koks, kas izmanto atvasinātos datus. Koks sasniedza 89% precizitāti [3]**

2.1.2. Klasiskā Beijesa klasificēšana

Šajā sadaļā tiek apskatīts datizracē pielietots algoritms, kuru sauc par Beijesa klasificēšanu. Algoritms arī ir relatīvi vienkārši saprotams, tas balstās uz statistisku pieeju, lai prognozētu objektu klases.

Beijesa klasifikatoriem ir dabiski, ka tie spēj klasificēt objektus daudzās klasēs, kas citu veidu klasifikatoriem ir ļoti sarežģīti, piemēram, lineārajiem klasifikatoriem [35]. Apskatot Beijesa klasifikatorus, vienkāršāk ir izprast naivo Beijesa klasificēšanu.

Naivajai Beijesa klasificēšanai arī ir nepieciešama treniņu datu kopa, lai varētu aprēķināt statistisku informāciju, kuru izmantos, lai vēlāk varētu veikt prognozes par jaunu objektu piederību konkrētai klasei. Treniņu datu kopai ir nepieciešami marķēti dati (kurai klasei objekts pieder), lai varētu aprēķināt nepieciešamās varbūtības. Zemāk ir pāris piemēru ar varbūtībām, kuras ir iespējams aprēķināt pie nosacījuma, ka ir pieejama marķēta treniņu kopa.

$P(C_k) = \frac{n_k}{n}$ - varbūtība, ka objekts pieder klasei C_k (te n_k ir šīs klases objektu skaits treniņa kopā);

[35]

$P(X_i = a \mid C_k) = \frac{n'}{n_k}$ - varbūtība, ka ja objekts pieder klasei C_k, tad i-ais atribūts = a (te n' ir attiecīgo objektu skaits treniņa kopas klasē C_k); [35]

Naivā Beijesa algoritmā tiek pieņemts, ka katras klases ietvaros objektu parametri pieņem vērtības neatkarīgi viens no otra. Tas stipri vienkāršo nepieciešamo varbūtību aprēķināšanai, kur objekts tiek klasificēts balstoties uz visām tā parametru vērtībām, kas parādīts formulā zemāk.

$$P(X_1 = x_1, \ldots, X_p = x_p \mid C_k) = \prod_{i=1}^{p} P(X_i = x_i \mid C_k)$$

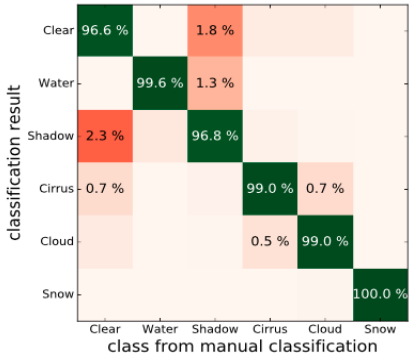
2.6. attēls **Naivā Beijesa klasifikatora funkcija. Tā kā tiek pieņemts, ka parametru vērtības ir neatkarīgas, tad kopējo varbūtību var aprēķināt izrēķinot katra parametra piederību klasei, šīs varbūtības pēc tam reizinot kopā [35]**

Lai varētu aprēķināt augstāk esošo varbūtību, jāizmanto Beijesa teorēma, kas redzama zemāk.

$$P(C_k \mid X_1 = x_1, \ldots, X_p = x_p) = \frac{P(C_k) P(X_1 = x_1, \ldots, X_p = x_p \mid C_k)}{P(X_1 = x_1, \ldots, X_p = x_p)}$$

2.7. attēls **Beijesa teorēma [35]**

Darba apskatītajā rakstā [3] tika izmantota nevis naivā Beijesa klasificēšana, bet klasiskā, kas ir sarežģītāka. Šajā klasificēšanā netiek pieņemts, ka parametru pieņemtās vērtības ir neatkarīgas. Līdzīgi kā lēmumu koku gadījumā, arī šai metodei netika atrasts viens optimālais klasificēšanas algoritms (varbūtību kopa, bet gan vairākas), kas aprēķinātas balstoties uz dažādu spektru datiem. Rakstā iegūtie rezultāti sasniedza ievērojami labākus rezultātus salīdzinot ar lēmuma kokiem (virs 90%). Rakstā algoritms, kas sasniedza labākos rezultātus bija, kas rēķināja varbūtības pēc šādām vērtībām: $B03 \times S(B9, B1) \times I(B10, B2) \times B12 \times I(B2, B8A)$. Algoritms sasniedza 98% korektu klasifikāciju pār tādām pašām klasēm, kas apskatīta lēmumu koku sadaļā. Attēlā zemāk ir redzama matrica, kur ilustrēti korekti klasificēto klašu varbūtības un klases, kuras konkrētā klase tika visvairāk sajaukta.



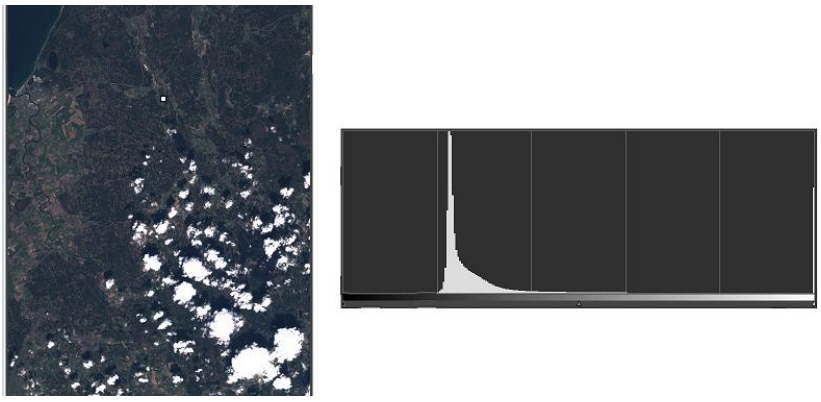
2.8. attēls **Labākā algoritma korekti klasificēto klašu varbūtības un visvairāk nekorekti klasificēto klašu attiecība [3]**

2.2. Kombinētu attēlu vizuālā izskata vienādošanas algoritmi

Šajā darba daļā tiek apskatīti literatūrā pieejami algoritmi, kurus var pielietot, lai atrisinātu problēmu, ka jāapvieno vienā attēlā daļas no vairākiem citiem satelīta attēliem, kur katrs attēls var vizuāli atšķirties viens no otra dažādu faktoru dēļ. Ietekmējošie faktori varētu būt, piemēram, izmantoto sensoru atšķirības, ārējās vides variācija kā piemēram apgaismojums, laikapstākļi, dienas laiks, kad dati tika uzņemti u.c.

2.2.1. Attēla histogrammas jēdziens

Katram attēlam var izrēķināt savu histogrammu. Attēla histogramma reprezentē statistisko informāciju par to cik daudz ir unikālu krāsu attēlā un cik katra konkrētā krāsa ir bieži sastopama attēlā. Attēlojot attēlu histogrammu vizuāli, ir pieņemts uz X ass attēlot unikālo krāsu skaitu, bet uz Y ass attēlot skaitu cik katra krāsa ir sastopama attēlā. Attēlā zemāk ir redzams piemērs satelīta attēlam un tā krāsu histogrammai [60].

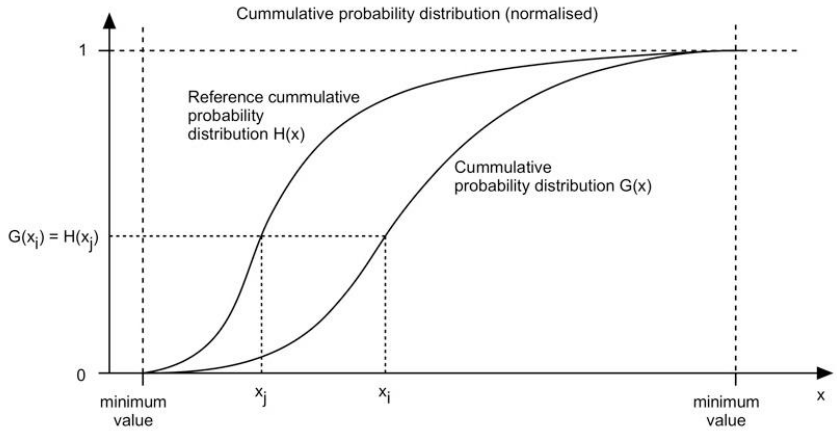


2.9. attēls **Attēla histogrammas piemērs**

Augstāk redzamajā histogrammā ir redzams, ka šajā attēlā visi dati galvenokārt atrodas vienā krāsu reģionā, kas tuvāks melnajai krāsai. Otrs datu reģions ir izteikts pīķis pie baltās krāsas, kas viegli redzams ir mākoņi.

2.2.2. Attēla histogrammas saskaņošanas algoritms

Eksistē daudzi dažādi algoritmi [61, 62], kas balstīti uz attēla histogrammu analīzi, lai pārveidotu attēla datus. Vienkāršākā attēla histogrammas saskaņošanas algoritma ideja ir tāda, ka ievaddatos tiek saņemti divi attēli. Viens attēls kalpos par paraugu, bet otrs attēls tiks izmainīts tā, lai tā krāsas atbilstu pēc iespējas tuvāk parauga attēlam. Algoritms izrēķina abiem attēliem histogrammas. Pēc histogrammu iegūšanas, tām tiek izrēķinātas kumulatīvā sadalījuma funkcijas, kuras pēc tam izmanto, lai pārrēķinātu apstrādājamā attēla katra pikseļa vērtību uz tuvāko pikseļa vērtību parauga attēlā. Attēlā zemāk ir ilustrēta kumulatīvā sadalījuma izmantošana, lai pārveidotu attēla pikseļu vērtības.



2.10. attēls **Kumulatīvā sadalījuma funkcijas diviem attēliem [62]**

Augstāk redzamajā attēlā tiek parādīts kā tiek nomainītas pikseļu vērtības izmantojot aprēķinātās histogrammu kumulatīvās sadalījuma funkcijas. Algoritms apstrādā katru attēla pikseli, atrod pikselim atbilstošo kumulatīvo sadalījuma funkcijas vērtību. Šo vērtību izmanto, lai atrastu references attēla kumulatīvās sadalījuma funkcijas atbilstošo pikseļa vērtību, kurai sakrīt attiecīgā sadalījuma vērtība. Attēlā zemāk ir redzams piemērs aprakstītā algoritma ievaddatiem un rezultātam.

2.11. attēls **Histogrammu saskaņošanas algoritma ievaddatu, rezultāta piemērs [62]**

Augstāk redzamajā attēlā kreisais attēls bija algoritma references attēls, bet vidējais attēls tika mainīts. Labais attēls ir iegūtais algoritma rezultāts. Kā redzams, tad šāda veida algoritmi derēs darba praktiskajā daļā, jo ļauj iegūt līdzīga izskata attēlus.

3. EKSISTĒJOŠI RISINĀJUMI, KAS APSTRĀDĀ SENTINEL-2 DATUS

Šajā nodaļā tiek apskatīti eksistējoši risinājumi, kas apstrādā Sentinel-2 datus, lai iegūtu datus vai servisus, kurus, lietotājs var izmantot, sev nepieciešamiem nolūkiem. Tiek apskatīti līdzīgi risinājumi autora izstrādātajai sistēmai.

3.1.EOxCloudless

Šis risinājums ir Austrijas kompānijas EOX IT Services veidots serviss, kas piedāvā lietotājam iegādāties sagatavotus satelīta datus pasaules teritorijā. Tiek piedāvāti no Sentinel-2 satelīta datu produktiem ģenerēti dati, kas ir bez mākoņiem, attēlu izskats ir konsekvents. Attēlā zemāk ir redzams 2019. gada datu paraugs Latvijas teritorijā no šī risinājuma.

3.1. attēls **Piedāvāto datu apskates iespējas paraugs tīmekļa pārlūkā, kur satelīta dati tiek atrādīti ar WMTS servisa palīdzību [64]**

Piedāvājumā ir ģenerēti dati no 2016., 2018., un 2019. gada Sentinel-2 datiem. Ir iespējams iegūt datus kā ģeoreferencētus attēlus GeoTIFF un JPEG75 formātos. Maksa par 2019. gada satelīta datiem visā pasaulē ir 5000€ , par 2018 gada datiem 3000€, bet 2016. gada dati izmaksās 1500€. Ir iespējams arī nopirkt mazāku reģionu datu apakškopu, kas izmaksā no 400€.

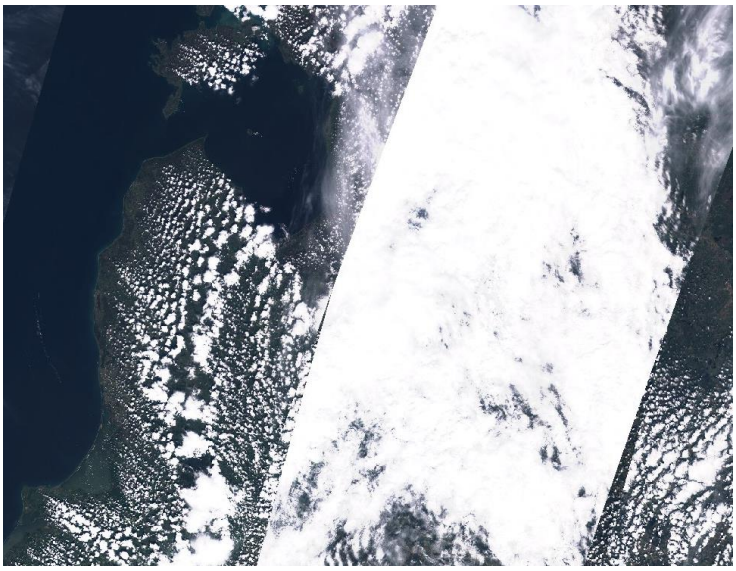
Interesanti, ka 2019. gada dati ir ģenerēti no Sentinel-2 L2A datu produktiem, bet 2018. gada dati no L1C datu produktiem. Šī darba ietvaros realizētais risinājums izmantoja L1C datu produktus.

Bez ģenerētu datu ieguves, tiek piedāvāti arī ģeotelpiski attēlu servisi (WMS/WMTS), kurus var izmantot tīmekļa un darbvirsmas programmatūrās. Ir iespējams izmantot šo servisus bez maksas nekomerciālos nolūkos. Komerciāliem nolūkiem tas arī ir maksas pakalpojums.

3.2.SENTINELHub

Vēl viens risinājums, kurš piedāvā uz Sentinel-2 datiem balstītus risinājumus ir SENTINELHub. Risinājumu veidojusi kompānija Sinergise, kas ir Slovēnijas uzņēmums. Uzņēmumam ir atsevišķs meitas uzņēmums SENTINEL HUB, kas bāzēts Austrijā.

Risinājums piedāvā apskatīt aktuālus Sentinel-2 un citus satelītu programmu uzņemtos datus pārlūkprogrammā ar interaktīvas kartes palīdzību. Apskatāmie dati nav nekā apstrādāti, tie satur mākoņus un citus nevēlamus objektus. Risinājums ļauj vizuāli apskatīt dažādu spektru kombināciju attēlus no pieejamajiem 13 Sentinel-2 datu spektriem. Ir iespējams izdrukāt apskatītā reģiona attēlu. Šo datu apskatīšana ir bez maksas. Attēlā zemāk ir redzams attēla paraugs no interaktīvās kartes, kur redzama Latvijas teritorija RGB spektru attēlā.



3.2. attēls **SENTINELHub brīvpieejas tīmekļa interaktīvās kartes piemērs [66]**

Risinājumā ir ļoti ērti pārslēgties pa dienām un redzēt atšķirīgos Sentinel-2 satelīta uzņemtos attēlus.

Par mēneša abonēšanas maksu ir iespējams iegūt piekļuvi pie sagatavotiem OGC WMS servisiem, kas veidoti publicējot nemodificētus Sentinel-2 datus. Ir papildus pieejami citi OGC servisi kā piemēram WFS, lai iegūtu papildus informāciju par konkrētiem datiem. [67]

3.3.Sentinel-2 satelītattēlu pamatkartes mozaīkas serviss

Latvijā reģistrēta uzņēmuma SIA "Baltic Satellite Service" piedāvājumu klāstā ir pieejams satelītattēlu pamatkartes WMTS serviss, kas ģenerēts no Sentinel-2 satelīta attēliem. Piedāvāto

servisu lietotājs var pats integrēt savos tīmekļa vai darbvirsmas risinājumos. Piedāvātais serviss ir maksas pakalpojums. Piedāvātā servisa dati ir pieejami pār Baltijas teritoriju.

Tiek piedāvāta arī arhīva funkcionalitāte, kur lietotājs var izmantot vecākus datus, lai varētu veikt izmaņu noteikšanu un analīzi konkrētā laika posmā. Katram satelīta attēla fragmentam, kas tiek padots ar ģeotelpisko datu servisa palīdzību, ir pieejami papildus meta dati, kas ļauj izsekot Sentinel-2 datu avotu no kā tie tika ģenerēti. [65]

4. DARBĀ REALIZĒTAIS RISINĀJUMA APRAKSTS

Šajā nodaļā tiek sniegts detāls autora realizētās sistēmas apraksts, kas spēj izveidot aktuālāko satelīta datu noklājumu pār interesējošo reģionu.

Sistēmas sākotnējie ievaddati ir lietotāja interesējošais reģions, kurā nepieciešams ieklāt aktuālos satelīta datus. Šīs nodaļas nākošajā sadaļā tiek detālāk aprakstīta lietotāja ievaddatu ievadīšana.

Pēc lietotāja intereses reģiona ievadīšanas, sistēmai ir jāspēj analizēt to, balstoties uz tā ģeogrāfiskajām koordinātām, ir jāspēj izgūt satelīta datu fragmentus no datu piegādātāja servisiem. Detālāks skaidrojums par šīs problēmas risināšanu tiek apskatīts nodaļas otrajā sadaļā.

Kad interesējošais reģions ir aizpildīts ar aktuālākajiem satelīta datiem, sistēmai ir jāspēj tajos automātiski detektēt mākoņu klātos reģionus, šie reģioni ir jāpiefiksē, tie kalpos par ievaddatiem nākamajam sistēmas procesam. Detāla mākoņu reģionu noteikšana tiek aprakstīta šīs nodaļas trešajā sadaļā.

Kad sistēma ir noteikusi mākoņiem klātos reģionus, šiem reģioniem tiek meklēti citi atbilstošie satelīta datu avoti no datu piegādātāja arhīva, kas uzņemti senākā laikā. Tas tiek darīts tad, ja detektēto mākoņu reģionu skaits ir pietiekoši liels. Atkal tiek pielietota mākoņu detektēšana līdz brīdim, kad tiek atrasts kombinētais attēlu rezultāts ar pietiekamu skaidrās debess noklājumu. Detālāks procesa apraksts tiek sniegts nodaļas ceturtajā sadaļā.

Darba piektajā sadaļā tiek apskatīts sistēmas posms, kad datu reģionam ir atrasti visi nepieciešamie dati, tiek aprakstīts algoritms kā no esošajiem datiem tiek veidoti kombinētie attēlu fragmenti.

Darba sestajā sadaļā tiek aprakstīta situācija, kad lietotāja intereses reģionam jau ir iepriekš atrasts kombinētais attēls, tiek aprakstīts algoritms, lai optimāli atjaunotu šī attēla datus.

Septītajā sadaļā tiek aprakstīts pielietotais attēlu apstrādes algoritms, lai realizētu vienmērīgu vizuālo izskatu kopējam satelīta datu noklātajam reģionam.

Astotajā sadaļā tiek aprakstītas izmantotās pieejas, lai ģenerētos datus būtu tālāk iespējams izmantot citā programmatūrā, piemēram, publicējot datus ar WMS servisa palīdzību vai iesaucot datus ĢIS programmatūrā, piemēram, QGIS.

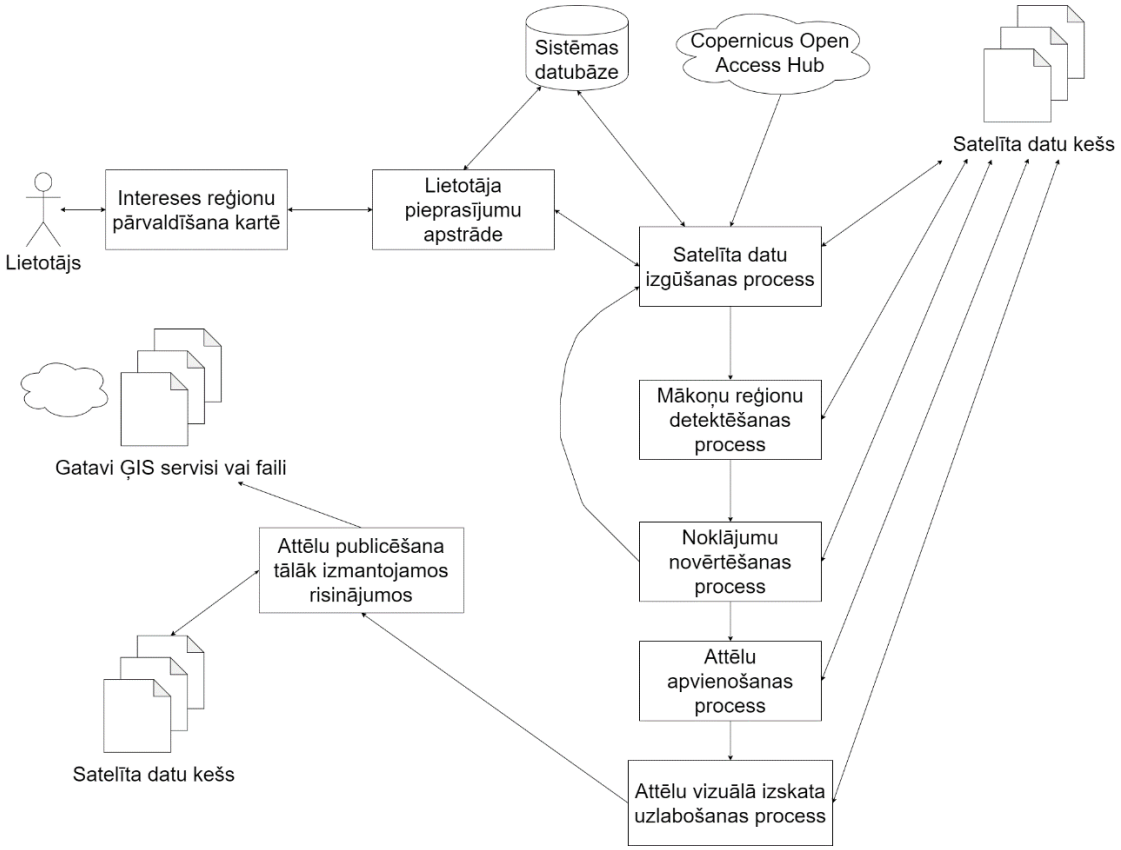
4.1. Kopējās sistēmas vispārīgs apraksts

Šajā sadaļā tiek sniegts vispārīgs ieskats par realizētās sistēmas kopējo struktūru, tās komponentēm un kā tās komunicē savā starpā, lai sasniegtu sistēmas gala rezultātu.

Darbā realizētā sistēma veidota priekš lietotāja, kas veic sistēmas ģenerēto satelīta attēlu tālāku publicēšanu citās ĢIS sistēmās. Izstrādātā sistēma veidota ar domu, ka tās lietotājam interesē konkrēti reģioni pasaules teritorijā, kur nepieciešams redzēt aktuālos satelīta datus, piemēram, tas varētu būt uzņēmums, kurš darbojas Latvijas teritorijā, konkrētā jomā, piemēram, mežsaimniecībā. Šādam gadījumam varētu interesēt, piemēram, konkrētas mežu teritorijas Latvijā. Sistēma veidota tā, lai lietotājam būtu vienkārši redzēt intereses reģionus kartē, būtu iespējams tiem veidot satelīta datus. Lietotājs no sistēmas saņemtu ģeoreferencētus satelīta attēlus, kurus tālāk ir iespējams izmantot citās sistēmās. Sistēma arī spētu nodrošināt ģenerēto ĢIS servisu izveidi un atjaunošanu izmantojot sistēmas veidotos attēlus par servisu ievaddatiem.

Lai sistēmas lietotājam varētu nodrošināt ērtu intereses reģionu izveidi un apskati, sistēma tika projektēta tā, ka datubāzē tiek glabāti intereses reģioni, to atjaunošanas informācija un izmantotie Copernicus satelītu attēli, lai uzbūvētu intereses reģionu. Šī informācija tiek izmantota lietotāja izmantotajā interaktīvajā kartē, kas ļauj lietotājam viegli redzēt definētos reģionus, uzzināt kad pēdējo reizi reģions ir bijis atjaunots, startēt vai apturēt reģiona satelīta attēlu izveidi. Nākotnē būtu iespējams redzēt arī informāciju, par gala datu integrāciju citās sistēmās, būtu iespēja šos datus automātiski atjaunot. Realizējot šo sistēmu praksē, iespējams, ka gala lietotājam nebūtu nepieciešama grafiskā saskarne, pietiktu ar automātisku procesu, kas periodiski tiktu laists, lai veiktu satelīta datu atjaunošanu un šo datu atjaunošanu integrētajās ĢIS sistēmās.

Sistēmas projektējumā bez datubāzes tika iekļauts arī lokālais failu sistēmas kešs, kas atbildīgs par lejupielādēto datu glabāšanu un atkārtotu izmantošanu, lai taupītu tīkla un rēķināšanas resursus. Satelīta attēlu apstrādes procesi tika nodalīti atsevišķos moduļos, lai varētu problēmu vieglāk risināt un uzturēt programmu vēlāk. Bez satelīta datu keša, nepieciešams arī glabāt iepriekš apstrādātos sistēmas gala rezultātus, lai tos nākotnē būtu iespējams izmantot datu atjaunošanas procesam. Izveidotās sistēmas, tās procesu diagramma ir redzama attēlā zemāk.



4.1. attēls **Autora realizētās sistēmas galveno komponenšu, procesu, to savstarpējo saistību diagramma**

Kā redzams attēlā, tad lietotājs ar sistēmu komunicē caur tīmekļa karti, kartes komponente tiek izpildīta uz tīmekļa pārlūka programmatūras.

No šīs komponentes pieprasījumi tālāk iet uz lietotāja pieprasījumu apstrādes komponenti, kas ir starpposms starp lietotāju un pārējiem sistēmas procesiem. Šī komponente startē vai aptur satelīta attēlu apstrādes procesus, kas tiek laisti paralēlos izpildes pavedienos, kur katrs lietotāja intereses reģions tiek apstrādāts paralēli.

Satelīta attēlu apstrādes process sākas ar satelīta datu izgūšanas procesu. Šis process ir atbildīgs par satelīta attēlu noklājuma veidošanu, kas aptvertu lietotāja ievadīto intereses reģionu. Process veic datu lejupielādi no Copernicus Open Access Hub izmantojot HTTP protokolus.

Pēc noklājuma lejupielādes, apstrādi sāk nākamais process, kas ir mākoņu detektēšanas process. Šis process apstrādā katru lejupielādēto satelīta attēlu, kur attēlam tiek detektēti dažādu klašu pikseļu reģioni, piemēram, mākoņu, ūdens, ēnu, skaidras debess reģioni. Dati tiek saglabāti uz diska un veikta nākamā procesa izpilde.

Nākošais sistēmas process veic lejupielādēto datu noklājuma novērtēšanu. Process pārbauda vai apvienojot dažāda vecuma lejupielādētos datus, no kuriem ir izņemti nevēlamo mākoņu reģionu dati, ir pietiekoši augsts datu noklājums. Ja novērtējums ir pārāk zems, tad tiek veikta vecāku datu noklājuma veidošana. Pretējā gadījumā tiek startēts nākamais process.

Attēlu apvienošanas process veido gala noklājuma attēlus, kur viena reģiona attēli veidoti kombinējot dažādos laika posmos iegūtos satelīta datus, kuros ir izņemti mākoņu reģioni. Attēli apvienoti tā, ka katrs pikselis satur pašu jaunāko informāciju, kas pieejama.

Attēla vizuālā izskata uzlabošanas process pielieto attēlu apstrādes algoritmus gatavajiem satelītu attēliem, lai vienādotu to vizuālo izskatu, jo attēlu daļas var izskatīties izteikti atšķirīgi, kas radies no dažādos laikos uzņemto attēlu apvienošanas.

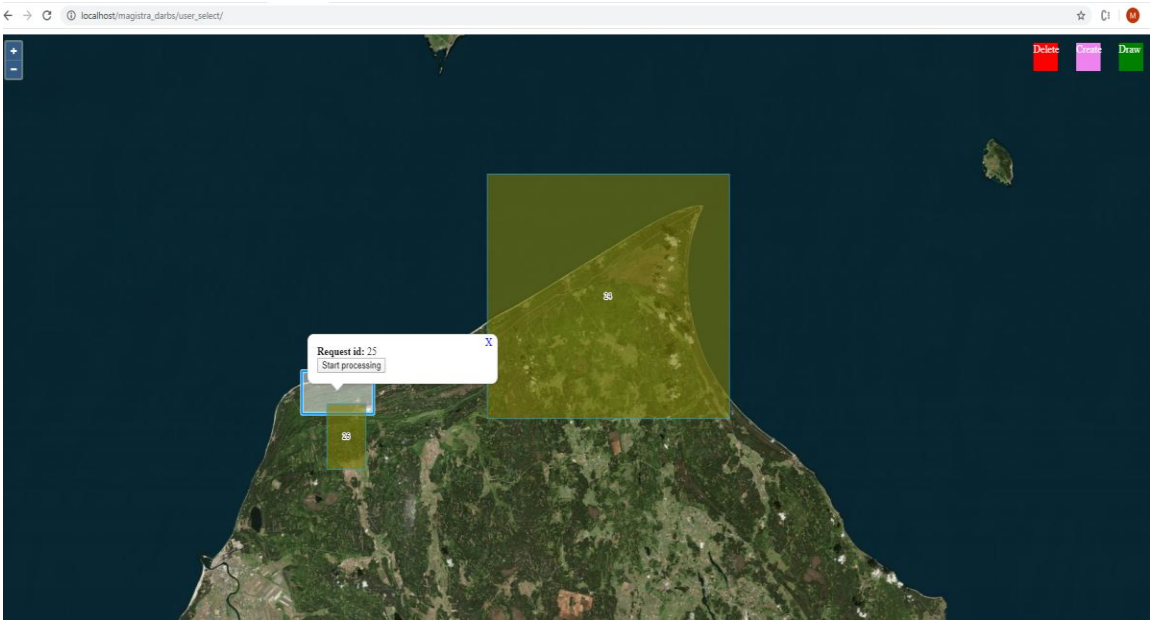
Visbeidzot ir sistēmas process, kas ļauj iegūtos datus tālāk integrēt citās ĢIS sistēmās, piemēram QGIS [36] vai Geoserver [40].

Katrā nākamajā sadaļā tiek sīkāk aprakstīts katrs no konkrētajiem sistēmas posmiem, kas tika šeit īsumā ieskicēts.

Autora realizētās sistēmas pirmkods ir publiski pieejams vietnes http://bitbucket.org repozitorijā [35].

4.2. Sistēmas lietotāja ievaddati

Šajā sadaļā tiek apskatīti realizētās sistēmas lietotāju ievaddati. Sistēma ļauj tās lietotājam ievadīt ģeogrāfisku reģionu ar tīmekļa lietotnes palīdzību. Lietotnē ir realizēta interaktīvā karte, kas kalpo kā grafiskā saskarne ievaddatu ievadīšanai. Šī sistēmas daļa arī ļauj lietotājam mijiedarboties ar pārējiem sistēmas procesiem. Realizētais interaktīvās kartes piemērs ir redzams attēlā zemāk.



4.2. attēls **Tīmekļa lietotnes grafiskā saskarne, kas ļauj ievadīt un pārvaldīt intereses objektus**

Lietotājam interaktīvā karte ļauj pārvietoties horizontālā un vertikālā virzienā pa pasaules teritoriju izmantojot peles palīdzību. Pietuvināt un attālināt interesējošo reģionu ar peles rullīša palīdzību. Intereses reģionu ir iespējams uzzīmēt kartē arī ar peles palīdzību, jābūt izvēlētai reģiona atlikšanas opcijai, tad ir iespējams atlikt reģionu klikšķinot ar kreiso peles taustiņu un kustinot peli, šādi tiek iegūts taisnstūrveida poligona reģions.

Kad lietotnē ir atlikts intereses reģions, tad lietotne to saglabā WKT [7], teksta formātā, kas ir plaši izmantots ģeogrāfisku datu ģeometrijas formāts. Izveidoto ģeometriju tīmekļa lietotne saglabā sistēmas PostgreSQL SQL datubāzē [8].

Kartē ir arī iespējams apskatīt jau iepriekš izveidotus lietotāja reģionus, uz katra no reģioniem ir iespējams uzklikšķināt, to darot tiek iegūta papildus opcija startēt vai apturēt satelīta datu meklēšanas procesu konkrētajam reģionam. Identificējot konkrēto reģionu ir iespējams arī uzzināt vai sistēma pašlaik rēķina konkrētajam reģionam datus vai nē.

Lai atvieglotu darba izstrādi, tika realizētas arī palīgfunkcijas kā piemēram lietotāja intereses reģionu dzēšana, lietotāja iepriekš skatītā kartes reģiona saglabāšana, lai vēlāk atverot saskarni tā atrastos iepriekš skatītajā reģionā.

Aprakstīto interaktīvo karti autors realizēja ar moderno tīmekļa tehnoloģiju palīdzību: TypeScript [38], HTML5 un CSS3. Kartes funkcionalitātei tika izmantota OpenLayers [19] bibliotēka, kas ļauj veidot interaktīvas tīmekļa kartes.

4.3. Lietotāja pieprasījumu apstrāde

Šajā sadaļā tiek apskatīta sistēmas otrā procesa darbība, kas veic visu lietotāja ievaddatu apstrādi. Interaktīvā karte komunicē ar šo procesu izmantojot HTTP protokola POST un GET metožu palīdzību. Tiek apstrādāti šādi lietotāja pieprasījumi:

- Intereses reģiona izveide, dzēšana;
- Intereses reģiona rēķināšanas statusa iegūšana;
- Intereses reģiona rēķināšanas startēšana, apturēšana.

Process apstrādāto ievaddatu rezultātus saglabā sistēmas PostgreSQL datubāzē, bet, ja tiek startēta vai apturēta intereses reģiona rēķināšana, tad process palaiž vai aptur atsevišķā pavedienā tālākos sistēmas procesus.

Bez lietotāja pieprasījumu apstrādes, process uztur visus pārējos paralēlo procesus atmiņā, ja tīmekļa serviss tiek apturēts, tad šīs process veic paralēlo procesu apturēšanu pirms tas tiek pats apturēts.

Līdzīgi kā pārējās sistēmas komponentes, neskaitot lietotāja ievaddatu komponenti, šis process tika realizēta Python programmēšanas valodā. Šai komponentei tika izmantota Flask [39] Python bibliotēka, kas ļāva startēt šo procesu kā tīmekļa serveri, lai komponente spētu komunicēt ar lietotāja ievaddatu tīmekļa lietotnes procesu.

4.4. Intereses reģiona aizpildīšana ar satelīta datu reģioniem

Šajā sadaļā tiek apskatīta sistēmas trešā procesa darbība. Šajā procesā par ievaddatiem kalpo PostgreSQL datubāzes tabulas ieraksti, kuri reprezentē satelīta reģiona izveides pieprasījumu. Papildus datubāzē tiek glabāti arī iepriekš jau lejupielādēto satelītu attēlu aprakstošā informācija konkrētajam intereses reģionam. Šie dati tiek arī analizēti, kad tiek veikta lietotāja pieprasījuma apstrāde.

Šo procesu ir iespējams startēt un apturēt no interaktīvās kartes, kas tika aprakstīta lietotāja ievaddatu sadaļā.

Process sākot darbu vispirms vēršas pie Copernicus Open Access Hub turpmāk API HUB izmantojot HTTP GET pieprasījumu, lai iegūtu sarakstu ar jaunākajām pieejamām Sentinel-2 [5] satelīta Level-1C granulām, kas satur lietotāja intereses reģionu vai tā fragmentu. Detāls apraksts par Sentinel-2 datu produktiem tiek sniegts 2. nodaļā.

API HUB nodrošina dažādus meklēšanas kritērijus [41], lai varētu meklēt sev vēlamos satelīta attēlus programmātiski. Šī procesa funkcionalitātes realizēšanai tika izmantoti šādi meklēšanas kritēriji:

- ingestiondate – datums un laiks, kad dati ir iegūti,
- producttype – satelīta datu produkta veids, šī darba ietvaros tā bija konstanta vērtība "S2MSI1C", kas ir Sentinel-2 Level-1C satelītu datu granulas,
- platformname – satelītu programmas veids, arī konstanta vērtība "Sentinel-2",
- footprint:"intersects(<geographic type>)" – ģeogrāfiskā reģiona ievades parametrs, kas nosaka, ka atgriezīs tikai tās datu granulas, kas satur daļu no šī reģiona vai pilnīgi visu.

Veicot granulu meklēšanu, ģeogrāfiskā reģiona laukā tiek norādīta lietotāja ievadītā intereses ģeometrija WKT formātā. Iegūtos meklēšanas rezultātus kārto pēc iegūšanas datuma dilstošā secībā. Papildus šim kārtošanas principam, tiek pielietota sekundārā kārtošana pēc datu granulas platības laukuma, kurš ir kopīgs ar interesējošo ģeometriju, šādi iegūst optimālu apstrādājamo objektu secību.

Meklējot jaunākās pieejamās granulas, datuma laukā tiek norādīts periods četras dienas no pašreizējā laika līdz šim brīdim, ja šajā datumā neizdodas pilnībā noklāt vēlamo reģionu, tad tiek iteratīvi meklēts ik pa 4 dienu vecākiem laika periodiem līdz izdodas atrast pilnībā noklājošu rezultātu.

Situācijā, kad tiek meklētas vecākas datu granulas, tad algoritms datuma periodu iegūst balstoties no vecākās apstrādātās datu granulas, kuras dati glabājas datubāzē, no šī datuma tiek atņemtas 4 dienas, lai iegūtu laika periodu. Šis periods tiek izmantots. Lai sāktu veidot interesējošā reģiona noklājumu, ja ar pirmo iterāciju tas neizdodas, tad periods tiek iteratīvi bīdīts ik pa četrām dienām senākā laika posmā, analogi kā tas notiek jaunākā noklājuma meklēšanas gadījumā.

Meklēšanai tika izmantota 4 dienu konstante, jo tas ir vidējais laika periods, kurā kļūst pieejami jauni satelīta dati. Meklēšanas un arī citu metadatu pieprasījumu rezultāti no API HUB tiek atgriezti XML datu formātā, iegūtie dati tiek noparsēti uz Python klašu objektiem, kurus tālāk realizētās programmas komponentes izmanto tālākai apstrādei.

Pēc katras iteratīvās meklēšanas no API HUB tiek iegūts saraksts ar satelīta attēlu granulām, lai pateiktu kuru granulu kombinācija no saraksta veido pilnu noklājumu pār intereses reģionu, tika realizēts papildus algoritms. Algoritms pie inicializācijas datubāzē atjauno speciālu

ģeometrijas lauku intereses reģiona ierakstam uz sākotnējo vērtību, kas sakrīt ar intereses reģiona ģeometriju. Šo lauku sauksim par gcd. Pēc tam algoritms ņem no atrasto granulu saraksta pa vienai granulai un atņem tās ģeogrāfisko ģeometriju no ģeometrijas lauka gcd. Ģeometriju atņemšanas operācijām tika izmantota PostgreSQL datubāzes ģeotelpisko funkciju moduļa PostGIS [42] pieejamā funkcija ST_Difference [43]. Algoritms konkrēto datu granulu pievieno pie pārklājošo granulu saraksta tikai tad, ja pēc veiktās atņemšanas operācijas tiek iegūta jauna netukša ģeometrija. Algoritms pārtrauc darbu, ja laukā gcd ir tukša ģeometrija vai ir iziets cauri visam apstrādājamo granulu sarakstam. Ģeometriju atšķirību detektēšanai un tukšu ģeometriju noteikšanai tika izmantotas šādas PostGIS funkcijas: ST_Equals, ST_IsEmpty [44, 45]. Attēlā zemāk ir redzams ilustratīvs piemērs kā algoritms iteratīvi atņem kopējos ģeometriskos reģionus no sākotnējā intereses reģiona ģeometrijas.

4.3. attēls **Noklājoša reģiona algoritma ilustrācija, kur no kreisās uz labo tiek iteratīvi atņemtas granulu ģeometrijas no intereses reģiona**

Ja augstāk aprakstītais algoritms no atrasto satelīta attēlu saraksta neatrod pilnu pārklājumu, tad algoritms meklē vēl papildus vecākas datu granulas, bet šajā gadījumā tiek meklēts nevis pēc sākotnējā intereses reģiona, bet pēc iztrūkstošā pārklājuma ģeometrijas.

Ja algoritmam izdodas atrast pārklājošo reģionu, tad tas padod rezultējošo sarakstu uz procesa noslēdzošo fāzi, kur atrastās datu granulas tiek lejupielādētas uz lokālās iekārtas cietā diska. Lejupielādes process ir veidots tā, ka vispirms tiek pārbaudīts vai konkrētā datu granula jau nav bijusi iepriekš lejupielādēta, tas ir svarīgi, jo granulu izmēri mērāmi vairākos simtos megabaitu, no aptuveni 300MB, ja reģions bijis mākoņains vai datu granula satur tikai nelielu daļu ar datiem, līdz 900MB, ja reģions bijis relatīvi maz mākoņains, saturēja visu ģeogrāfisko reģiona datus. Šis kešošanas mehānisms pārbauda vai uz cietā diska atrodas mape konkrētā vietā ar nosaukumu, kas atbilst granulas identifikatoram, ja tas neizpildās, tad tiek papildus pārbaudīts vai uz cietā diska neatrodas kompresēts datu fails (šis fails tiek izgūts no API HUB), kas satur granulas

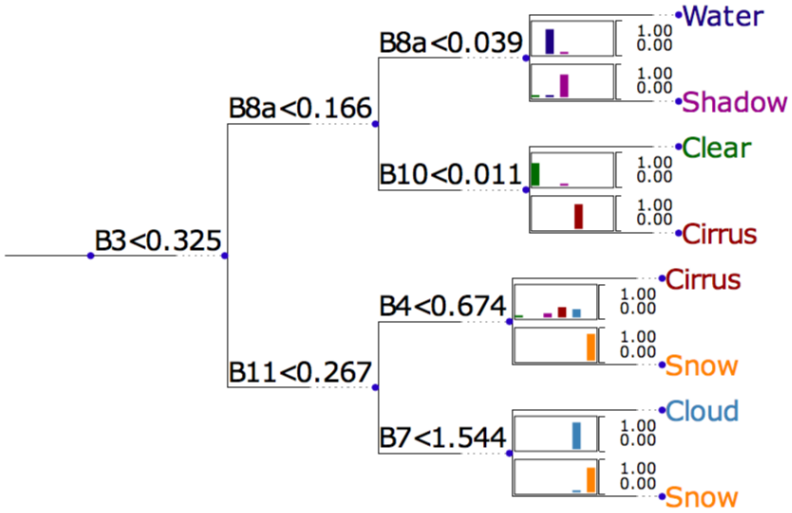
identifikatoru. Tikai tad, ja abas situācijas neizpildās, tiek veikts lejupielādes pieprasījums uz API HUB.

Pēc vajadzīgo granulu lejupielādes, process ir pabeidzis savu darbu, tālāk darbu var turpināt mākoņu detektēšanas process, kas aprakstīts tālāk.

4.5. Mākoņu reģionu detektēšana un piefiksēšana

Šajā nodaļas daļā tiek aprakstīts sistēmas process, kas par ievaddatiem saņem sarakstu ar datu granulām, kas iegūtas ar iepriekšējā procesa palīdzību. Process iteratīvi apstrādā katru granulu, kur katrai tiek pielietots darbā izvēlētais mākoņu detektēšanas algoritms.

Praksē tika realizēts lēmumu koka mākoņu detektēšanas algoritms, kas tika detāli apskatīts darba otrajā nodaļā. Šis algoritms tika realizēts, jo ir viegli saprotams un sniedz pietiekami labu precizitāti. Tika realizēts lēmuma koks ar dziļumu trīs, kas aprakstītajā literatūrā [3] sasniedz 87% lielu klasificēšanas precizitāti. Lēmuma koka attēls ir redzams zemāk



4.4. attēls **Darbā realizētais lēmumu koka detektēšanas algoritms**

Algoritms tika realizēts Python programmēšanas valodā, kur attēlu dati tika glabāti un apstrādāti ar NumPy [46] Python bibliotēkas datu struktūras un algoritmu palīdzību. NumPy tiek pielietots daudzos zinātniskos lietojumos, piemēram, mašīnmācīšanās algoritmu realizācijā, statistikā u.c. NumPy datu struktūra ļauj eleganti strādāt arī ar attēliem, tādēļ šī bibliotēka un arī pati Python valoda tika izvēlēta šī darba ietvaros.

NumPy datu struktūra ļauj strādāt nevis ar attēla individuālajiem pikseļiem, bet uzreiz ar visu attēlu kopumā, piemērs redzams zemāk esošajā Python koda rindā.

- clear[clear > 0] = 255

Šeit mainīgais clear ir NumPy objekts, kas ir divdimensionāla matrica, kas iegūta nolasot attēla failu. Ar šo konkrēto vienu operāciju tiek nomainītas visas matricas vērtības uz 255, kurām tagadējā vērtība ir lielāka par 0. Uz šīs NumPy metodes tika balstīta lēmuma koka implementācija.

Realizētais algoritms tika inicializēts ar pamata binära koka struktūru, kas atbilst augstāk redzamajam attēlam. Koka katra virsotne glabā informāciju par:

- vērtību, kura jāizmanto, lai pārvietotos uz bērnu šķautnēm,
- granulas attēla identifikatoru,
- atsauci uz kreiso bērnu,
- atsauci uz labo bērnu,
- NumPy matricu, kurā glabās rezultātu, kas nepieciešams, ja pārvietosies uz kreiso bērnu,
- NumPy matricu, kurā glabās rezultātu, kas nepieciešams, ja pārvietosies uz labo bērnu.

Pēc inicializācijas, tiek izpildīta koka DFS apstaigāšana, kur apstaigājot katru virsotni, algoritms izrēķina tai abas NumPy matricu vērtības.

Lai varētu izrēķināt matricu vērtības, nepieciešams nolasīt no diska konkrētu granulas spektra attēlu. Attēla nolasīšana nav triviāls process, jo ir trīs apgrūtinoši faktori saistībā ar granulu spektru datiem, kurus jārisina, lai tos varētu izmantot algoritmos.

1. Konkrēto spektru dati tiek mērīti reālos skaitļos, kuri netiek glabāti pa tiešo attēla spektra failā;
2. Attēlos var būt reģioni, kuros nav datu, tie ir speciāli jāapstrādā atsevišķi;
3. Savstarpējo spektru attēli savā starpā atšķiras izšķirtspējās, kā rezultātā paši attēli ir trīs dažādos izmēros.

Kā redzams, ir nepieciešami papildus apstrādes soļi pie attēlu nolasīšanas pirms tos var pielietot algoritmā. Pirmo augstāk minēto punktu var relatīvi vienkārši atrisināt, pie datu nolasīšanas, ir jāveic vērtību konvertācija uz reāliem skaitļiem, kur katra pikseļa nolasītā vērtība ir jāizdala ar QUANTIFICATION_VALUE [47] vērtību, kas tiek glabāta Level-1C granulas metadatu failā ar nosaukumu MTD_MSIL1C.xml. Otrs minētais punkts ir jāņem vērā apstrādājot datus gan šajā

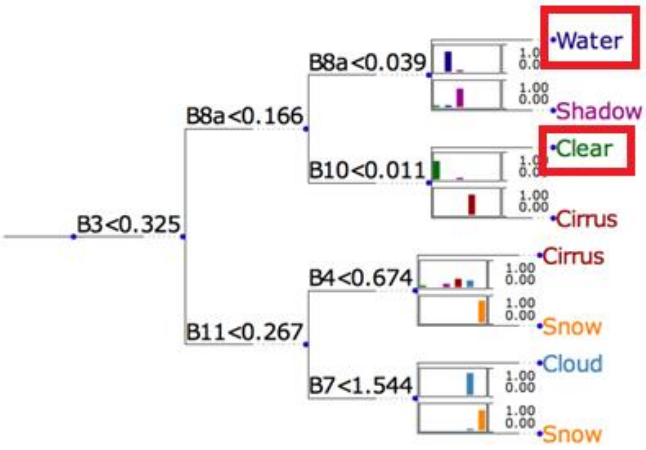
sistēmas posmā, gan arī citos. Spektru attēlu failos vērtība "0" ir rezervēta reģioniem, kuros nav dati. Trešo problēmu risina ar pāreju uz vienotu izšķirtspēju, lai varētu analizēt dažāda izmēra spektra datus vienā algoritmā. Šajā darbā par vienotu izšķirtspēju izvēlējos 20 metrus, jeb 5490x5490px lielus attēlus. Tas nozīmē, ka veicot detektēšanu, trīs detālākos spektrus, kuru izmērs ir 10980x10980px, samazina uz divreiz mazāku attēlu, bet vēl 3 spektrus, kuru izmērs ir tikai 1830x1830px, palielina 3 reizes lielākus.

Ņemot vērā augstāk minēto informāciju, veicot spektra attēla nolasīšanu, tika veiktas papildus darbības, kā rezultātā tika iegūta NumPy divdimensionāla matrica, kuras izmērs ir 5490x5490px, katra vērtība satur reālu skaitli un nulle reprezentē pikseļus, kuros nav datu. Šo matricu var izmantot, lai izrēķinātu abas NumPy matricas, kas tiek glabātas konkrētajā koka virsotnē. Zemāk redzamās 3 koda rindiņas izrēķina abas NumPy matricas, kurās glabā vērtības 1 un 0.

- node.maskLeft[band < node.value] = 1
- band[band == self.NO_DATA] = granule.NO_DATA
- node.maskRight[band > node.value] = 1

Izrēķinātajā matricā 1 reprezentē to, ka konkrētais pikselis atbilst nosacījumam, bet 0, ka neatbilst. Otrā rindiņa ir nepieciešama, lai apstrādātu korekti tukšos datu reģionus, lai tie tiktu ignorēti (paliktu par 0 abās matricās, tāpēc sākotnēji tā bija uzstādīta uz ļoti lielu vērtību, lai korekti izpildītos pirmā koda rindiņa, bet pēc tam uzlikta atpakaļ uz 0, lai arī netiktu ņemta vērā trešajā rindiņā).

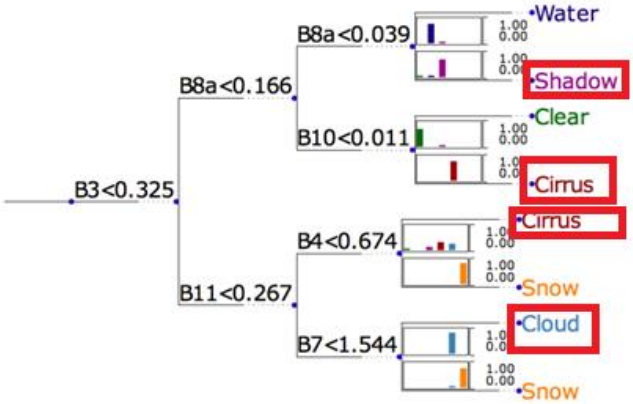
Pēc koka apstaigāšanas ir izrēķināta visa nepieciešamā informācija, lai varētu iegūt attēla masku, kas ļauj izgūt interesējošos attēla reģionus. Attēla masku iegūst veicot visu matricu reizināšanu, ejot cauri kokam no virsotnes līdz konkrētai lapai, kas reprezentē konkrēto objektu klasi attēlā. Šeit ar matricu reizināšanu saprot nevis matemātisko operāciju, bet ar operāciju, kur konkrētās matricas elements ar koordinātām x,y tiek reizināts tikai ar elementu ar koordinātām x,y no otras matricas. Šādi sareizinot konkrētās matricas iegūst jaunu matricu, kurā būs šūnas ar vērtību 1 tikai tām šūnām, kurām izpildījās visi nosacījumi. Darbā izmantotā skaidrā debess maska tika iegūta saskaitot izrēķināto ūdens reģionu matricu ar skaidrās debess matricu, kas iezīmēti attēlā zemāk.



4.5. attēls **Darbā izmantotās matricas, lai iegūtu skaidras debess reģionus**

Pie skaidrās debess matricas tika pieskaitīta ūdens matrica, jo bez tās, ūdens reģioni netika iekļauti detektētajos rezultātos, kas bija nevēlams rezultāts.

Otra darbā interesējošā maska bija mākoņu reģionu maska, to var iegūt saskaitot zemāk ievilktās matricas.



4.6. attēls **Darbā izmantotās matricas, lai iegūtu mākoņu reģionu masku**

Kā redzams, tad mākoņu maskas aprēķināšanai ir nepieciešams saskaitīt vairāk masku kā tas ir skaidras debess gadījumā. Pēc praksē novērotajiem rezultātiem, šķiet, ka šai maskai var kā alternatīvu izmantot skaidrās debess maskas inverso matricu. Tas gan varētu būt tikai pielietojams, ja apstrādājamā attēlā nav sniegotu reģionu.

Kad tiek aprēķināta granulas skaidro reģionu maska, tā tiek saglabāta uz cietā diska kā attēls, lai to varētu pēc tam tālāk izmantot vēlākiem aprēķiniem. Bez maskas faila tiek saglabāts arī skaidrās debess attēls, kas iegūts reizinot aprēķināto masku ar granulas RGB attēlu. Papildus

šiem diviem jaunajiem failiem tiek saglabāti arī konkrētā intereses reģiona skaidrās debess attēls un maskas fails, intereses reģiona faili.

Realizētais mākoņu detektēšanas algoritms tika veidots arī ar kešošanas mehānismu analogi granulu datu lejupielādei, pirms tiek veikta mākoņu detektēšana, tiek pārbaudīts vai jau iepriekš konkrētajai granulai nav aprēķināti skaidrās debess reģioni un maskas fails. Ja tas jau ir izdarīts, tad netiek veikta mākoņu detektēšana vēlreiz, tiek izveidoti tikai konkrētā intereses reģiona attēli no eksistējošās maskas faila un skaidrās debess faila.

Intereses reģiona izgūšana no granulas faila tiek veikta, lai paātrinātu tālākos sistēmas aprēķinus. Lai varētu korekti izgūt nepieciešamos intereses reģionus no granulām, tika izstrādāts algoritms, kas spēj izgūt no granulu attēliem taisnstūrveida fragmentus, kas atbilst intereses reģiona ģeogrāfisko koordināšu reģionam pēc kura tika pieprasīts attēla fragments.

4.6.Noklājuma novērtēšana

Šajā sadaļā tiek apskatīts sistēmas process, kas apstrādā iepriekšējā procesa rezultātu - detektētos mākoņu reģionus. Procesa uzdevums ir pateikt vai ir nepieciešams turpināt meklēt vecākus granulu attēlus, lai iegūtais apvienoto attēlu kopums būtu pieņemami aizpildīts ar skaidras debess datiem.

Izstrādātajā sistēmā noklājumu novērtējumam tiek izmantotas aprēķinātās intereses reģionu skaidrās debess maskas, kuras tika izrēķinātas mākoņu detektēšanas posmā. Algoritms nolasa no datubāzes visas apstrādātās granulas, kas satur lietotāja intereses reģionu. Pēc tam no visām granulām tiek atlasītas tikai pašas jaunākās, kuras veido pilnu reģiona noklājumu. No šīm granulām tiks veidots gala noklājuma novērtējums. Kad ir iegūts šis saraksts, tad katra granula tiek iteratīvi apstrādāta. Granulai x atrod visas pārējās granulas y, kuras dala kopēju reģionu ar granulas x intereses reģionu. Granulas y tiek arī sakārtotas pēc to vecuma dilstošā secībā. Kad ir iegūts vajadzīgais šķērsojošo granulu saraksts, tad algoritms sāk veidot kopējo granulas x noklājuma masku, kas veidots kombinējot granulas x intereses reģiona skaidrā debess masku ar visu granulu y intereses reģionu maskām. Algoritms pirmajā solā apvieno granulas x masku ar pirmās granulas no granulu saraksta y masku. Lai maskas varētu apvienot, no katras ir jāizgūst kopējais attēla reģions, kas ir starp granulas x un y maskām. To ir iespējams iegūt, vispirms atrodot kopīgo ģeogrāfisko reģionu.

Mākoņu detektēšanas un masku saglabāšanas posmā ir realizēta loģika, ka katram intereses reģionam tiek saglabāts arī speciāls piesaistes teksta fails, kas glabā ģeogrāfisko informāciju par konkrēto intereses reģionu. Šī faila detālāka informācija tiek sniegta nodaļas astotajā sadaļā. Šo failu izmanto, lai programma korekti noteiktu intereses reģiona ģeogrāfiskās koordinātas. Kad ir zināmas abu reģionu ģeogrāfiskās koordinātas, tad tiek aprēķināts kopējā reģiona ģeogrāfiskās koordinātas izmantojot PostgreSQL PostGIS funkciju ST_Intersection. Kad ģeogrāfiski kopīgais reģions ir zināms, tad atliek izrēķināt šim reģionam atbilstošās pikseļu koordinātas abos masku failos, to var izdarīt relatīvi vienkārši, ja ir zināma ģeogrāfiskā izšķirtspēja attēlam vienā pikselī, attēla platus, augstums un attēla stūru koordinātas. Visa šī informācija ir pieejama vai nu pēc augstāk minētajiem aprēķiniem vai izgūta no apstrādāto granulu metadatu failiem.

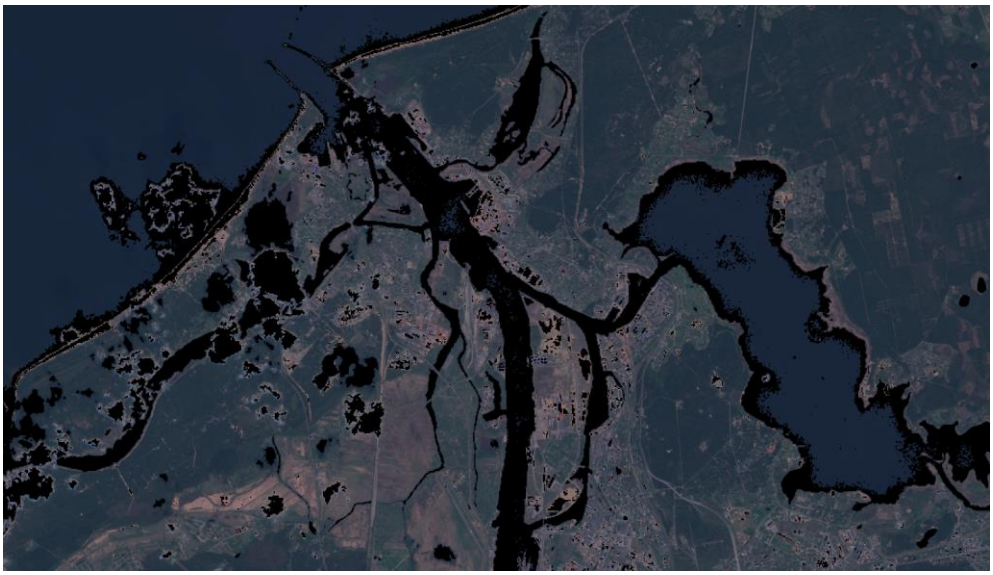
Kad ir atrasti abu masku kopīgie reģioni, tad jāveic masku saskaitīšana, lai iegūtu kombinēto masku, kura reprezentē apvienotā attēla informāciju. Saskaitīšanu veic ar NumPy matricu saskaitīšanu, pēc saskaitīšanas jāveic papildus operācija, kur jāuzstāda visi pikseļi uz vērtību 1, kuru vērtības ir lielākas par viens. Tas tā var gadīties, ja abos masku reģionos kādos no pikseļiem ar vienādām koordinātām bija skaidras debess pikseļi. Saskaitītos kopīgās maskas reģionus pēc tam jāliek atpakaļ granulas x maskā, modificēto masku jāatgriež atpakaļ tālākiem aprēķiniem. Kad ir iegūta modificētā maska, tad var veikt pārbaudi vai iegūtajā maskā ir pietiekoši liels skaits pikseļu, lai varētu akceptēt esošos datus par pietiekamu.

Veicot datu skaitīšanu ir svarīgi ņemt vērā to, ka daļa no pikseļiem var saturēt tukšos datu reģionus. Ja tiek rēķināts datu noklājums neapvienotai granulas maskai, tad noklājumu var izrēķināt saskaitot maskā pikseļus ar vērtību 1, šo skaitu pēc tam jāizdala ar maskas kopējo pikseļu skaitu, no kura atņemts bez datu pikseļu skaits. Līdzīgi aprēķini jāveic, ja rēķina kombinētajai maskai, bet te jāsamazina bez datu reģiona skaits atņemot no tā to pikseļu skaitu, kas atbilst datu reģioniem no pārējām kombinētajām maskām, kurās šajā bez datu reģionā bija dati.

Ja izrēķinātais skaits ir pietiekošs, tad konkrēto intereses reģiona fragmentu var uzskatīt par apstrādātu, nav vairs nepieciešams pievienot citu granulu attēlus šim reģionam. Var turpināt veikt novērtēšanu pārējiem intereses reģionu fragmentiem. Ja arī pārējiem intereses reģioniem ir apmierinošs datu daudzums, tad process var veiksmīgi pabeigt darbu un ļaut turpināties nākamajam procesam, kas atbildīgs par gala attēlu sagatavošanu. Ja intereses reģiona noklājums nav pietiekams, tad var pārtraukt novērtēšanu un sākt meklēt vecāku granulu noklājumu. Jāpiebilst, ka šeit var veikt papildus analīzi, jo teorētiski var būt situācija, ka konkrētais intereses reģiona

fragments ir maza izmēra, ja tam nedaudz pietrūkst datu, lai apmierinātu minimālo datu skaitu, varbūt, ka uz kopējā fona tas arī ir apmierinoši, ja pārējo interese reģionu datu noklājums ir pietiekoši augsts, tad visa reģiona kopējais datu skaits būtu apmierinošs.

Šī darba izstrādē par apmierinošu noklājumu tika izvēlēts 93% liels attēlu fragmentu noklājums. Šāda procentuāla vērtība tika izvēlēta pirmkārt, lai paātrinātu izstrādes un testēšanas laiku, jo gan mākoņu detektēšana, gan granulu failu lejupielāde aizņem laiku, kas mērāms pāris minūtēs. Otrkārt, šāda veida pieeja bija nepieciešama arī dēļ izvēlētā mākoņu detektēšanas algoritma. Algoritmam bija problēmas ar specifisku reģionu nekorektu klasifikāciju, kā rezultātā daļa no objektu klasēm tika bieži klasificēta kā nevēlama, piemēram, ūdens reģioni, kā rezultātā samazinājās noklājuma datu piepildīšanas laiks. Attēlā zemāk ir piemērs, kur klasificēšanas algoritms ir nekorekti detektējis lielu daļu ūdens reģionu par ne skaidras debess reģioniem.



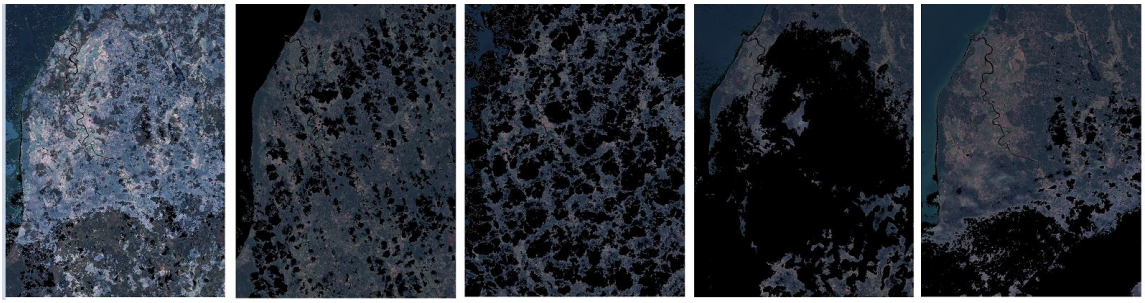
4.7. attēls **Izmantotā klasificēšanas algoritma nevēlamie blakus efekti, kur krastu zonas un ūdens reģioni tiek klasificēti nekorekti**

Attēlā redzamais attēls ir iegūts no datu granulas izgūstot tikai skaidrās debess pikseļus un ūdens pikseļus. Kā redzams, tad liela daļa no ūdens reģioniem, piemēram, gandrīz visa Daugava, jūras krastu reģioni un Ķīšezera krasti ir nekorekti klasificēti.

4.7. Attēlu apvienošana

Šis ir viens no noslēdzošajiem realizētās sistēmas soļiem. Šis process tiek startēts tad, kad noklājuma veiktā novērtēšana ir notikusi veiksmīgi, uz lokālā sistēmas diska saglabātās un apstrādātās datu granulas satur pietiekošu intereses reģiona skaidrās debess pikseļu kopu.

Šis process strādā pēc līdzīga algoritma kā noklājuma novērtēšanas process, kas tika detāli aprakstīts iepriekšējā sadaļā. Lielākā atšķirība ir tas, ka tiek iteratīvi veidots gala RGB attēls paralēli skaidrās debess matricu apvienošanai. Otra atšķirība ir tas, ka pēc katras attēla papildināšanas ar citu attēlu, tiek saglabāta uz diska pagaidu maska, kas atbilsts skaidrās debess pikseļu masku summai pašreizējam apvienoto attēlu rezultātam. Tas tiek darīts, lai atvieglotu algoritma sarežģītību, šādi nav jāatkārto visu iepriekš saskaitīto attēlu masku nolasīšana un analīze, kad tiek pievienots katrs nākamais attēls pie apstrādājamā attēla. Šādi apstrādājot katru attēlu fragmentu pāri x, y, vienmēr ir jāanalizē tikai divas mākoņu maskas. Attēlā zemāk ir ilustrēts piemērs apvienotajam gala attēlam un attēliem, kas tika izmantoti to iegūstot.



4.8. attēls **Realizētā apvienošanas algoritma gala rezultāts un attēli, kuri tika izmantoti apvienošanas procesā**

Augstāk redzamajā attēlā kreisais attēls ir procesa rezultāts, bet pārējie četri attēli tika izmantoti, lai attēlu uzģenerētu.

4.8. Uzģenerētā attēla periodiska atjaunošana

Šajā sadaļā tiek apskatīta situācija, kā būtu jārīkojas, ja sistēma jau ir atradusi intereses reģionam atbilstošo aktuālāko bez mākoņu satelīta attēlu. Lai šo attēlu uzģenerētu, sistēmai vajadzēja lejupielādēt pašus jaunākos pieejamās datu granulas, tad no tām tika izņemti mākoņu reģioni, reģioni tika aizpildīti ar vecāku granulu datiem līdz attēla kopējais noklājums sasniedz noteiktu apjomu. Augstāk aprakstītajā algoritmā tika izgūti no attēliem bez mākoņu reģioni, tie kombinēti kopā. Lai datus atjaunotu, pietiek atrast pašreiz aktuālāko satelīta datu noklājumu.

Noklājumā ir nepieciešams detektēt nevis skaidrās debess pikseļus, bet mākoņu un citu nevēlamu reģionu pikseļus, piemēram, ēnu, bez datu reģionus. Maskā šo reģionu koordinātās atliek vērtību 1, bet pārējos reģionos ir vērtība 0. Šo pikseļu masku var pēc tam izmantot, lai kombinētu aktuālo noklājumu ar iepriekš izrēķināto attēlu. Kombinēšana ar iepriekš ģenerēto attēlu tiek veikta pēc algoritma, ka visi jaunākie intereses reģioni tiek pārkopēti uz iepriekš ģenerēto attēlu, ja izrēķinātajā nevēlamo reģionu maskā pikseļu koordinātās ir vērtība 0. To var viegli realizēt izmantojot NumPy pieejamās operācijas, pikseļu pārneses koda rinda ir ilustrēta zemāk.

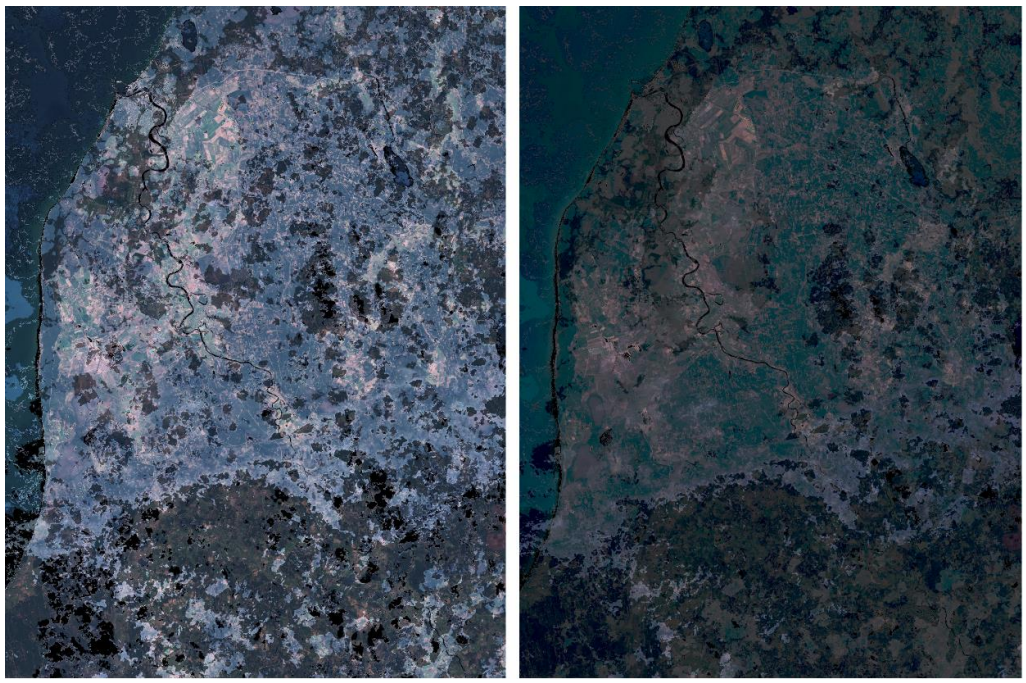
- combinedImg[cloudMask == 0] = latestCoverageImg

Augstāk redzamajā koda rindā cloudMask ir NumPy matrica, kas satur nevēlamo reģionu pikseļus, combinedImg ir NumPy matrica, ka satur iepriekš izrēķināto intereses reģionu vai tā daļu, bet latestCoverageImg ir NumPy matrica, kas satur jaunāku satelīta intereses reģiona informāciju, kurš sakrīt ar combinedImg ģeogrāfisko reģionu. Pēc veiktās operācijas iegūto attēlu var saglabāt kā gala attēlu. Kā redzams, tad attēla atjaunošanas process vairs nav tik apjomīgs, tā kā Sentinel-2 satelīta dati tiek papildināti vidēji 4 dienās vienā reģionā, tad arī būtu nepieciešams veikt atjaunošanu šādā laika periodā vienu reizi.

4.9.Satelīta attēlu vizuālā izskata vienādošana

Šajā sadaļā tiek apskatīts pielietotais algoritms, kas veic sagatavoto granulu vizuālā izskata korekciju, lai gala produkts izskatītos pēc iespējas konsekventāk visā intereses reģiona teritorijā. Šis process ir nepieciešams, jo sistēmas veidotie granulu attēli ir kombinācijas no dažādām granulām, kas uzņemtas, iespējams, ļoti atšķirīgos laika periodos. Katrā granulā var būt atšķirīgi apgaismojumi, var būt citi mainīgi faktori, kas ietekmē attēla gala izskatu.

Realizētajā sistēmā tika izmantots histogrammu saskaņošanas algoritms, kas tika apskatīts darba otrajā daļā. Algoritms tika pielietots attēlu apvienošanas procesa laikā. Precīzāk tas tika veikts tieši pirms apvienotais kopīgais attēlu reģions tiek ievietots atpakaļ attēlā, kuram pašlaik tiek papildināti dati no citiem satelītu attēliem. Algoritmam par references attēlu tiek padots attēls, kuru papildina, bet izmaināmais attēls tiek padots apvienotais kopējā reģiona attēls. Attēlā zemāk ir piemērs gala uzģenerētajam attēlam, kuru kombinējot netika izmantota krāsu korekcija, tam blakus ir tas pats attēls, bet algoritms šī attēla ģenerēšanā izmantoja histogrammu saskaņošanas algoritmu.



4.9. attēls **Gala apvienotā attēla salīdzinājums, ja apvienošanas procesā netiek pielietots histogrammu saskaņošanas algoritms un tiek pielietots**

Augstāk redzamajā attēlā kreisais attēls ir apvienotais algoritms bez histogrammu saskaņošanas algoritma pielietošanas, bet labais attēls ir. Kā redzams, tad labais attēls ir ar ievērojami labāku kopējo izskatu kā kreisais.

Jāpiemin, ka pielietojot minēto algoritmu rodas papildus problēmas, kuras nepieciešams risināt. Daļa no reģioniem var zaudēt informāciju, kas ir svarīga gala lietotājam, piemēram, zaļas krāsas reģions var kļūt pelēks. Autora prāt viena no pirmajām lietām, kuru vajadzētu uzlabot ir tas, ka histogrammu analīzē nedrīkst ņemt vērā tukšos reģionus, kas bija kādā no attēliem. Skaidrās debess attēlu fragmentos tukšie dati tiek kodēti kā melna krāsa, ja attēlā ir ļoti liels datu iztrūkums un tas tiek izmantots par references attēlu, tad otrs attēls var kļūt stipri tumšāks, ja tajā ir lielāks daudzums datu. Otra problēma ir references attēla izvēle, lai algoritms pēc iespējas mazāk kropļotu attēlus, būtu labāk izvēlēties references attēlus tā, ka lielākas platības datu saturošos attēlus izmanto par references attēlu, bet mazāko attēlu pielāgo tam. Tādējādi lielāko datu apjomu atstāj kā ir, maina pēc iespējas mazāk.

Augstāk aprakstītie uzlabojumi netika realizēti šī darba ietvaros, tie ir vieni no potenciālajiem uzlabojumu darbiem, ja darbs tiks turpināts nākotnē.

4.10. Attēlu publicēšana ĢIS

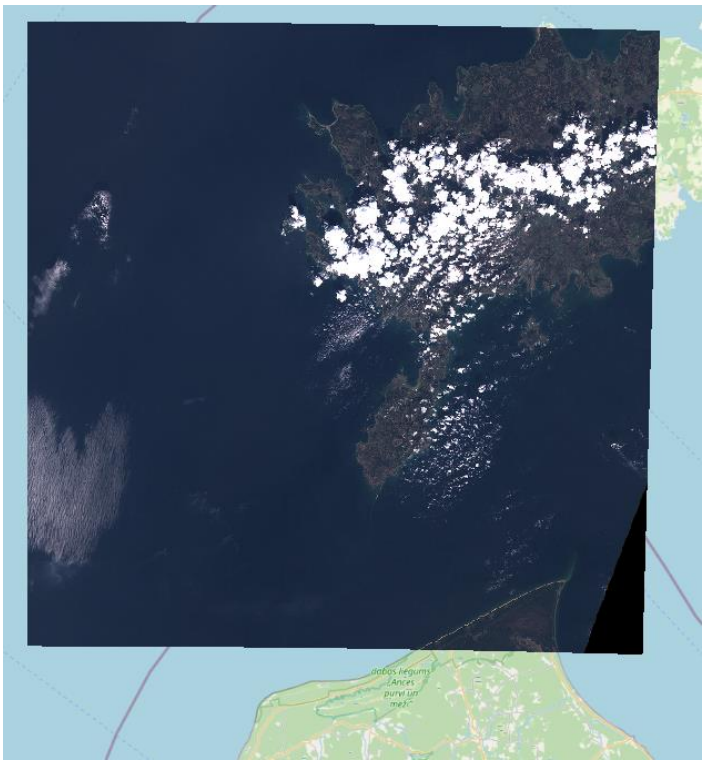
Šajā sadaļā tiek apskatīti sistēmas gala procesi, kas nepieciešami, lai lietotājs varētu ar sistēmu ģenerētos attēlus tālāk izmantot citās ģeotelpiskās informācijas sistēmās.

Sentinel-2 granulu attēli ir saglabāti JPEG2000 formātā, kur katram attēla failam ir pievienota galvene, kas satur ģeotelpisko informāciju, kur attēlam jāatrodas ģeogrāfiskajā telpā. Attēlā zemāk ir redzams piemērs šai galvenes informācijai.

```
<gml:featureMember>
  <gml:FeatureCollection>
    <gml:featureMember>
      <gml:RectifiedGridCoverage dimension="2" gml:id="RGC0001">
        <gml:rectifiedGridDomain>
          <gml:RectifiedGrid dimension="2">
            <gml:limits>
              <gml:GridEnvelope>
                <gml:low>1 1</gml:low>
                <gml:high>10980 10980</gml:high>
              </gml:GridEnvelope>
            </gml:limits>
            <gml:axisName>x</gml:axisName>
            <gml:axisName>y</gml:axisName>
            <gml:origin>
              <gml:Point gml:id="P0001" srsName="urn:ogc:def:crs:EPSG::32634">
                <gml:pos>499985 6500035</gml:pos>
              </gml:Point>
            </gml:origin>
            <gml:offsetVector srsName="urn:ogc:def:crs:EPSG::32634">10 0</gml:offsetVector>
            <gml:offsetVector srsName="urn:ogc:def:crs:EPSG::32634">0 -10</gml:offsetVector>
          </gml:RectifiedGrid>
        </gml:rectifiedGridDomain>
        <gml:rangeSet>
          <gml:File>
            <gml:rangeParameters>
              <gml:QuantityExtent uom="urn:ogc:def:crs:EPSG::32634">inapplicable inapplicable</gml:QuantityExtent>
            </gml:rangeParameters>
            <gml:fileName>gmljp2://codestream/0</gml:fileName>
            <gml:fileStructure>Record Interleaved</gml:fileStructure>
          </gml:File>
        </gml:rangeSet>
        <gml:coverageFunction>
          <gml:GridFunction>
            <gml:sequenceRule order="+x-y">Linear</gml:sequenceRule>
            <gml:startPoint>1 10980</gml:startPoint>
          </gml:GridFunction>
        </gml:coverageFunction>
      </gml:RectifiedGridCoverage>
    </gml:featureMember>
  </gml:FeatureCollection>
</gml:featureMember>
```

4.10. attēls **Ģeogrāfiskā informācijas piemērs, kas pievienots granulas B02 JPEG2000 attēlam**

Kā redzams attēlā, tad ir norādīta gan koordinātu sistēma, kurai attēls atbilst, šajā gadījumā tā ir EPSG:32634 [48] koordinātu sistēma, apakšējā kreisā stūra x koordināta un attēla augšējā labā stūra y koordināta norādītajā koordinātu sistēmā. Ir pieejama arī izšķirtspējas informācija x un y virzienā, kas šajā gadījumā ir 10 metri. Ar šo norādīto informāciju pietiek ĢIS sistēmām, lai failu varētu korekti atrādīt ģeogrāfiskajā pozīcijā. Attēlā zemāk ir piemērs, kur granulas sākotnējais attēls ir atvērts ar QGIS programmatūras palīdzību.



4.11. attēls **Paraugs no QGIS programmatūras, kurā ir atvērts RGB granulas attēls, kas tiek rādīts virs OpenStreetMap [49] kartes**

Attēlā redzams, ka granulas attēls korekti ir iestājies pareizajā ģeogrāfiskajā vietā. Šādi ir vienkāršākais veids pārbaudīt vai attēla saturošā ģeogrāfiskā informācija ir pareizi ievadīta.

Izveidotā autora sistēma veidota tā, ka tās ģenerētie attēlu faili tiek glabāti nevis JPEG2000 formātā, bet PNG attēlu formātā. Tas tika darīts, lai atvieglotu attēlu apskati un netērētu izstrādes laiku, lai meklētu risinājumus, kas spētu saglabāt failus citos attēlu formātos. Lai ģenerētos failus varētu tālāk izmantot citās ĢIS sistēmās, bija nepieciešams papildus izveidot funkcionalitāti sistēmā, kas spēj ģenerēt ģeogrāfiskās informācijas piesaistes failus attēliem, kas saturētu līdzīgu ģeogrāfisko informāciju kā tas ir sākotnējiem granulu attēliem. Tika realizēts, ka katram attēla failam, kuru ģenerē sistēma tiek izveidoti divi papildus piesaistes faili, kur viens satur informāciju par attēla koordinātām un izšķirtspēju, bet otrs fails satur informāciju par koordinātu sistēmu. Pirmais faila veids saucas World file [37], kas satur koordinātu informāciju, bet otrs ir PRJ [51] fails, kas satur projekcijas informāciju. Abi failu standarti ir Esri kompānijas [50] ieviesti failu formāti, kas kļuvuši plaši izmantoti ĢIS industrijā. Ar sistēmu ģenerētie attēli, kuri publicējami citās ĢIS programmās satur pašu attēlu failu, piemēram, “24_finished.png” un divus minētos ģeogrāfiskās informācijas piesaistes failus, piemēram, “24_finished.pgw” un “24_finished.prj”.

Otrs variants, kas tika apskatīts darba ietvaros ir kā ģenerētos attēlu failus ir iespējams ērti apskatīt tīmekļa bāzētos ĢIS risinājumos, piemēram, tādā risinājumā, kā lietotāja ievaddatu ievades un pārvaldības interaktīvā karte, kura tika realizēta šī darba ietvaros, tā ir detālāk aprakstīta ceturtajā nodaļā.

Lai varētu failus integrēt tīmekļa risinājumos, ir jāatrisina vispirms tas, lai ģenerētajiem failiem būtu ģeogrāfiskā piesaistes informācija, tas tika aprakstīts sadaļas sākumā. Otra problēma ir failu relatīvi lielais izmērs, kas apgrūtina darbu ar tiem. Tā kā sistēmas gala rezultāti ir vairāku Sentinel-2 datu granulu kopa vai šo granulu failu daļas, kur katrs attēla fragments var aizņemt līdz 100x100 km reģionu, viena attēla fragmenta RGB faila izmērs var pārsniegt 100MB. Šāda problēma ir bieži sastopama ģeogrāfisko informācijas sistēmu (ĢIS) pasaulē [14], kur bieži saskarās ar to, ka ir nepieciešams lietotājam vizuāli parādīt datus, kas ir ļoti apjomīgi. Viens no veidiem ir izmantot tīmekļa karšu standartu (WMS) [13], kas ļauj izveidot no apjomīgajiem datiem daudzus maza izmēra attēlus, kurus pēc tam ir viegli atrādīt lietotājam ar tīmekļa lietotnes palīdzību. Mūsdienās, lai WMS servisu paātrinātu, ir standarta prakse tā datus kešot attēlos uz diska. WMS keša līmeņi sastāv no daudziem vienāda izmēra attēliem, jeb karšu flīzēm [15]. Tīmeklim izplatītākais WMS karšu flīžu izmērs ir 256x256 pikseļi. Neoficiāls standarts ir izmantot WMS servisu, kura dati ir Google Web Mercator projekcijā, kurai ir standarta izšķirtspējas, projekcija un keša flīžu izmēri [16, 17, 57]. Attēlā zemāk ir parādīts piemērs, kur kartē ir vizuāli parādīti flīžu gabaliņi, kas veido kopējo karti, kas redzama lietotāja ekrānā.

4.12. attēls ***Interaktīvās kartes piemērs, kur speciāli tiek vizualizētas kartes flīzes, kas veido kopējo kartes attēlu [17]***

Ir pieejami vairāki risinājumi, ar kuru palīdzību ir iespējams veidot WMS servisu no attēliem, šī darba ietvaros tika izmantots Geoserver [40] atvērtā koda risinājums, kas ļauj publicēt dažāda veida ģeogrāfisko informāciju par OGC [52] standarta servisiem, kurus tālāk ir iespējams

izmantot citās programmatūrās, piemēram, tīmekļa risinājumos. WMS serviss ir viens no senākajiem OGC standartiem, kas joprojām tiek plaši izmantots praksē.

Geoserver risinājumā ir iespējams izveidot datu glabātuves, jeb Stores [53], ar to palīdzību tiek definēts datu avots Geoserver vidē, kuru tālāk var izmantot, lai publicētu gala servisus. Šī darba ietvaros atbilstošais datu glabātuves tips ir rastra ImageMosaic store[54], kas ļauj publicēt vairākus ģeoreferencētus attēlus kā vienu kopīgu WMS servisu.

Publicējot Sentinel-2 attēlus ir jāņem vērā, ka katrs attēls var būt savā koordinātu sistēmā, ja intereses reģions bija pietiekoši plašs, piemēram, valsts izmērā. Sentinel-2 attēli tiek referencēti UTM/WGS84 [56] projekcijā, kur pasaule ir sadalīta 60 zonās, kur katra zona ir savā projekcijā. Latvijas teritoriju šķērso pāris šīs zonas, piemēram, 34N un 35N zonas. Lai varētu atrādīt vienotā WMS servisā datu avotus no dažādām ģeogrāfiskām projekcijām, ir nepieciešams datus pārprojicēt uz vienotu ģeogrāfisko projekciju. Ir vairāki iespējami varianti kā to varētu izdarīt. Viens no vienkāršākajiem variantiem, ja tiek izmantots Geoserver risinājums, ir publicēšanas procesā nokonfigurēt ImageMosaic, lai tas veiktu pārprojicēšanu automātiski uz vienotu projekcijas sistēmu. Konfigurāciju var veikt izveidojot speciālu papildus failus, kas saucas "indexer.properties". Failu ir jāizveido mapē, kurā atrodas visi publicējamie attēlu fragmenti, faila satura piemērs ir pieejams Geoserver dokumentācijā [55].

Svarīgākais ir jāizvēlas vienotā projekcijas sistēma uz kuru visi attēli tiks pārprojicēti. Pārprojicēšanas process strādā tā, ka sākotnējo attēlu izmaina tā, ka katra pikseļa koordinātas pārvieto no vecās pozīcijas uz jauno tā, lai tā korekti atbilstu jaunās projekcijas koordināšu sistēmai. Praksē pārprojicēta attēla kvalitāte būs parasti zemāka kā attēlam, kurš sākotnēji veidots konkrētai ģeogrāfiskai koordināšu sistēmai. Pārprojicēšana ievērojami pasliktina attēla kvalitāti, ja tiek pārprojicēts uz būtiski atšķirīgu koordināšu sistēmu no sākotnējās. Mūsu gadījumā optimāls risinājums ir par kopējo projekcijas sistēmu izvēlēties EPSG:3857 jeb Web Mercator projekciju [16, 17, 57]. Šī projekcija ir kļuvusi par standartu dažādās lietotnēs, piemēram, tīmekļa lietotnēs, mobilajās lietotnēs u.c. Šajā projekcijā ir Google Maps kartes, Bing Maps kartes, OpenStreetMap u.c. plaši izmantotas kartes. Bez plašā lietojuma industrijā, šī projekcija ir pietiekoši tuva UTM/WGS84 projekcijām, kā rezultātā pārprojicēšana īpaši nepasliktinās attēlu kvalitāti. Attēlā zemāk ir redzams piemērs, kur QGIS lietotnē ir pieslēgts WMS serviss, kas izveidots ar augstāk aprakstīto ImageMosaic metodi.

4.13. attēls **QGIS pieslēgts WMS slānis, kas veidots no sistēmas ģenerētiem granulu attēliem, kur granulu dati ir no dažādām UTM/WGS84 projekcijām**

Kā redzams attēlā, tad Kolkas reģionā esošais attēls ir no 34N zonas, bet Rīgas reģiona fragmenti ir 35N zonā. WMS serviss ir sakonfigurēts, lai dati tiktu pārprojicēti uz EPSG:3857 projekciju. Šo pašu WMS servisu ir iespējams arī iesaukt izstrādātajā tīmekļa lietotnē, kur lietotājs pārvalda intereses objektus. Iesauktā WMS slānis šajā lietotnē ir redzams attēlā zemāk.

4.14. attēls **Uzģenerētais WMS serviss, kas pieslēgts pie sistēmas intereses reģionu kartes tīmekļa pārlūkā**

Lai gan šī darba ietvaros netika saprogrammēta automātiska WMS servisu izveide, bet izveidoti WMS servisi Geoserver vidē manuāli ar roku, tomēr visi nepieciešamie izpētes darbi ir izdarīti, lai to nākotnē būtu iespējams realizēt automātiski. Geoserver nodrošina programmātisku pieeju pie tā funkcijām izmantojot tā nodrošināto REST API. Ir iespējams izveidot jaunu Store elementu tajā no savas programmatūras ar HTTP pieprasījumu palīdzību uz Geoserver, kas ļautu automātisku WMS servisu izveidi.

Bez automātiskās pārprojicēšanas, kuru veic Geoserver risinājums, ir iespējams atšķirīgo projekciju datus pārprojicēt uz vienotu projekciju pirms datu publicēšanas citos risinājumos. Ir pieejami gatavi rīki, kas spēj pārprojicēt eksistējošos rastra failus no sākotnējās projekcijas uz citu. Viens no atvērtajiem risinājumiem, kuru izmanto liela daļa citu ĢIS risinājumu ir GDAL [58], šī bibliotēka paredzēta, lai varētu viegli pāriet no viena ģeogrāfiskā datu formāta citā. Bibliotēka nodrošina arī rastra datu pārprojicēšanu izmantojot gdalwarp [59] funkciju. Jāpiemin, ka šī darba ietvaros netika realizēta attēlu apvienošana granulām, kurām ir atšķirīgas projekcijas, bet reizē tās savā starpā saturēja kopīgu intereses reģionu fragmentu. Tas netika realizēts, jo būtiski apgrūtinātu izstrādi un palielinātu risinājuma sarežģītību. To varētu realizēt kā nākotnes attīstības darbu, pie šādas realizācijas būtu nepieciešams granulu datus apstrādes laikā pārprojicēt uz vienotu sistēmu, lai varētu kopīgos reģionus izņemt no attiecīgā attēla un ielikt otrā.

REZULTĀTI

Tika praksē realizēta sistēma, kas spēj no Sentinel-2 satelītu attēliem veidot kombinētus satelīta attēlus, kuros nav mākoņu reģioni.

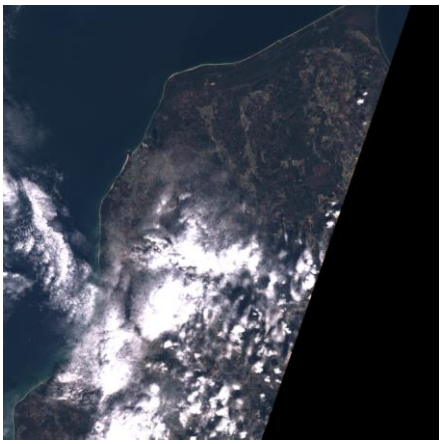
Tika noprogrammēta tīmekļa interaktīvā karte, kas ļauj lietotājam uzzīmēt kartē intereses reģionus, startēt, apturēt satelīta attēlu izveides procesu, uzzināt izveides procesa statusu un dzēst intereses reģionus. Lietotāja intereses reģioni, granulu informācija tika glabāta autora izveidotā PostgreSQL datubāzē, kurā tika izmantota PostGIS paplašinājums, kas ļāva glabāt un manipulēt ģeotelpisko atribūtu informāciju.

Attēlu kombinēšanas procesu pārvaldīšanai tika noprogrammēts Python skripts, kas darbojās kā tīmekļa serveris, kurš apstrādāja lietotāja pieprasījumus, kas tika veikti no interaktīvās kartes un startēja un apturēja satelītu apstrādes procesus konkrētajiem intereses reģioniem. Tīmekļa serveris darbojās kā paralēlu procesu pārvaldītājs, kur katrs intereses reģiona izveides process tika startēts atsevišķā Python pavedienā. Serveris veica procesu apturēšanu pēc lietotāja veikta pieprasījuma vai servera apturēšanas gadījumā.

Autors izveidoja algoritmu, kas spēj atrast jaunākās datu granulas, kas satur lietotāja ievadīto intereses reģionu. Tika noprogrammēta automātiska granulu datu lejupielāde uz lokālo ierīces atmiņu, tika izveidots lejupielādēto datu apstrādes mehānisms, kas ļauj no lejupielādēto datu kopuma izveidot Python objektus, kas ļāva tālāk veidot apstrādes algoritmu, kas izmanto šos datus. Tika papildus izveidots keša mehānisms, kas optimizēja sistēmas darbību, lai netiktu lejupielādēti atkārtoti dati, kas jau iepriekš tikuši ielādēti.

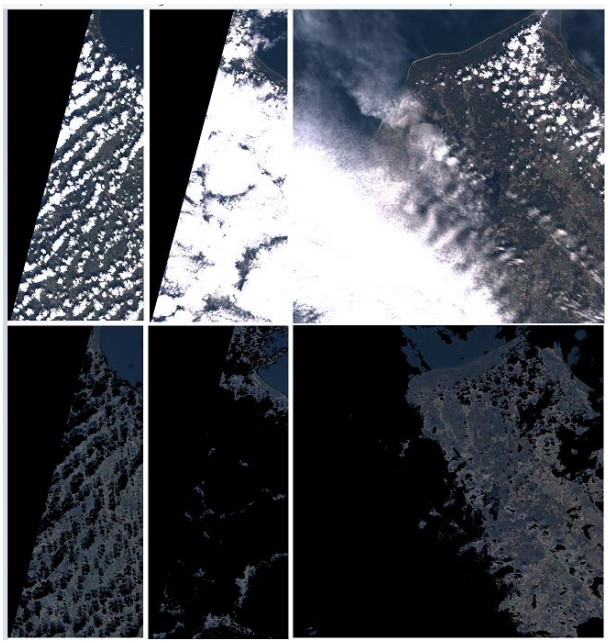
Mākoņu un citu intereses reģionu detektēšanai tika realizēts lēmumu koka algoritms, kurš tika apskatīts darba teorētiskās daļas laikā. Algoritms tika realizēts Python programmēšanas valodā, kur algoritmam tika izmantota NumPy bibliotēka par pamatu, lai realizētu matricu masku aprēķināšanu, kuras tālāk var izmantot, lai izņemtu no satelīta attēliem sev interesējošos datu reģionus vai veiktu datu noklājuma aprēķinus. Realizētajam algoritmam vidēji bija jāapstrādā attēlu dati, kuros vienā attēlā ir 100x100km liels ģeogrāfiskais reģions. Realizētajam lēmumu koka algoritmam bija jāapstrādā 7 attēla spektri (no 13), lai varētu aprēķināt vajadzīgās datu maskas. Apstrādājot gandrīz pilnībā noklātu 100x100km attēlu tas ir vidēji 250MB datu, kuri jānolasa no cietā diska un jāveic aprēķini. Uz šāda izmēra apstrādājamiem datiem, algoritms aprēķināja vajadzīgās maskas vidēji 45 sekundēs, kur visa sistēma tika palaista uz relatīvi standarta portatīvā datora, kuram bija 16GB operatīvā atmiņa, Intel i5 procesors un SSD cietais disks. Papildus laiku

vēl pēc tam prasīja pēcapstrādes procesi, kur aprēķinātās maskas, skaidrās debess attēli tika saglabāti uz cietā diska. Zemāk ir redzams attēla piemērs, kura apstrādes laiks vidēji atbilst aprakstītajam.



5.1. attēls **Viens no darba laikā izmantotajiem Sentinel-2 attēliem**

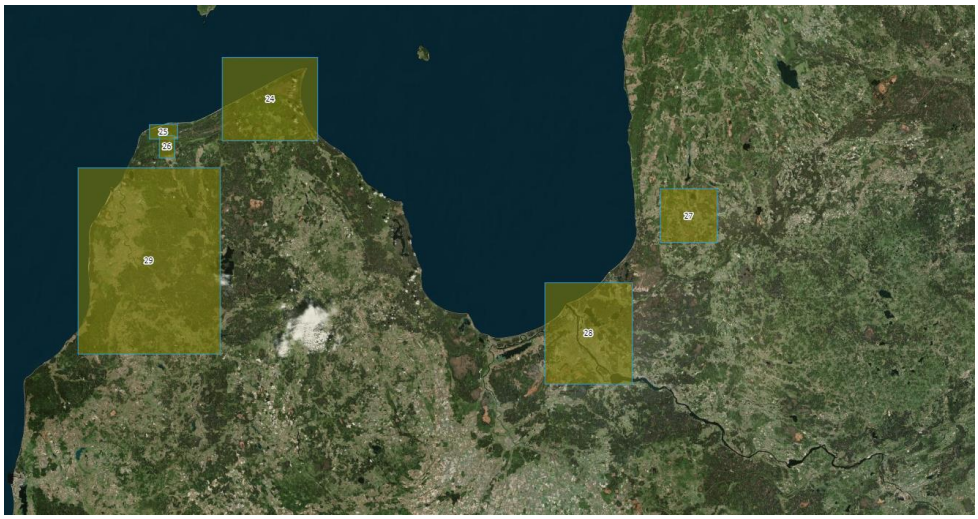
Realizētais lēmuma koks strādāja relatīvi labi salīdzinot ar to, ka tā realizācija nebija pārāk sarežģīta. Algoritms lielākoties spēja korekti detektēt dažāda veida mākoņus. Algoritmam bija problēmas ar ūdens reģioniem un ūdens tilpņu krasta reģioniem, tie bieži tika klasificēti kā ne skaidras debess pikseļi. Mākoņu detektēšanā lielāko problēmu algoritmam sagādāja mākoņu maliņu pikseļi, kas atrodas starp skaidrās debess reģionu un mākoni. Bieži nelieli reģioni palika skaidrās debess attēlā, iespējams, ka šo problēmu varētu risināt mākslīgi palielinot mākoņu detektētos laukumus, šādi iegūtu teorētiski labāku rezultātu, bet palielinātu rezultāta meklēšanas laiku. Attēlā zemāk ir redzami pāris analizēto attēlu piemēri un tajos detektētie skaidrās debess pikseļi.



5.2. attēls **Realizētā lēmuma koka algoritma apstrādāto datu piemēri un skaidrās debess rezultāti**

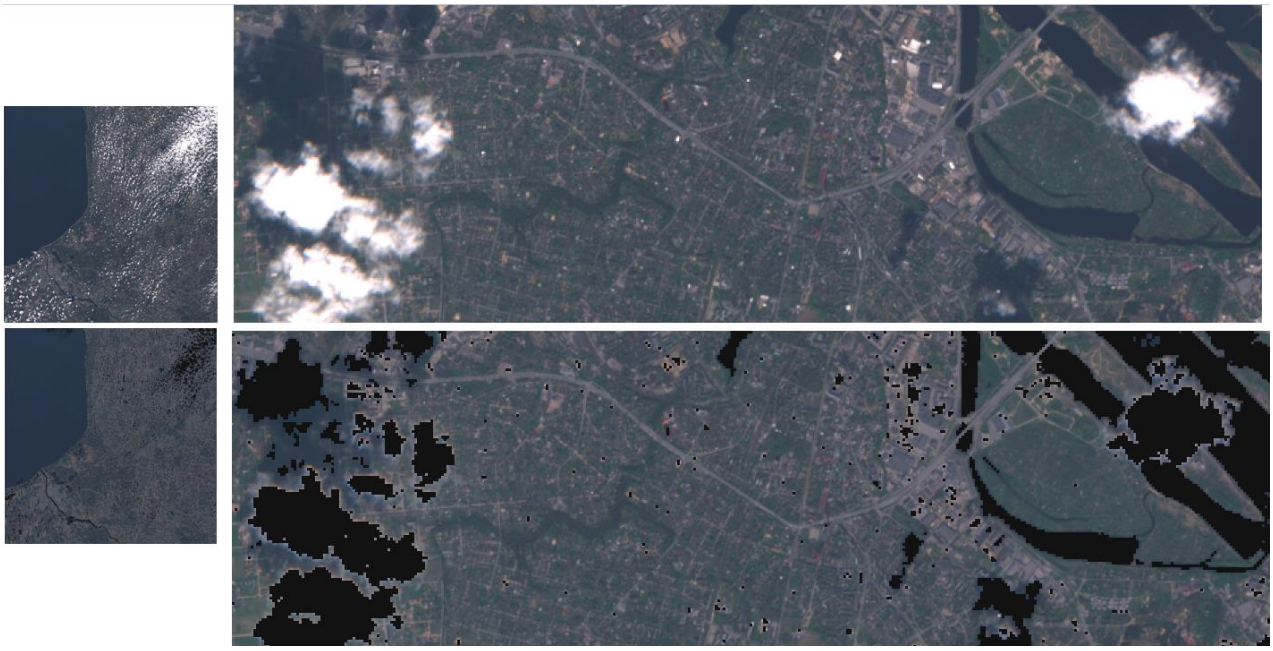
Augstāk redzamajā attēlā augšējie attēli ir sākotnējie Sentinel-2 attēli, bet apakšā ir algoritma izrēķinātie attēli bez mākoņiem.

Realizētais mākoņu detektēšanas algoritms tika testēts izstrādes laikā uz vairākiem Sentinel-2 satelītu attēliem. Apstrādātie attēli, galvenokārt, bija uzņemti 2020. gada aprīļa beigās līdz maija vidum, kad notika praktiskās daļas realizēšana. Tika izvēlēti vairāki ģeogrāfiskie reģioni, lai pārbaudītu algoritma darbību uz dažāda veida objektu klasēm, kas sastopami intereses reģionos. Lielākā daļa no testu reģionu kopas tika izvēlēta Latvijas teritorijā, jo, ja darbu reāli izmantotu praksē, tad tas, visticamāk, būtu izmantots šajā teritorijā. Attēlā zemāk ir piemērs ar Latvijā testēto intereses reģionu kopu.



5.3. attēls **Intereses reģionu testu kopa Latvijas teritorijā**

Tika izvēlēts intereses reģions, kas ir virs Rīgas teritorijas, lai pārbaudītu algoritma darbību uz blīvi apdzīvota pilsētas fona. Papildus reģionā ir arī ūdens tilpnes, liels daudzums ceļu, ēku. Attēlā zemāk ir redzams fragments no šī reģionu.

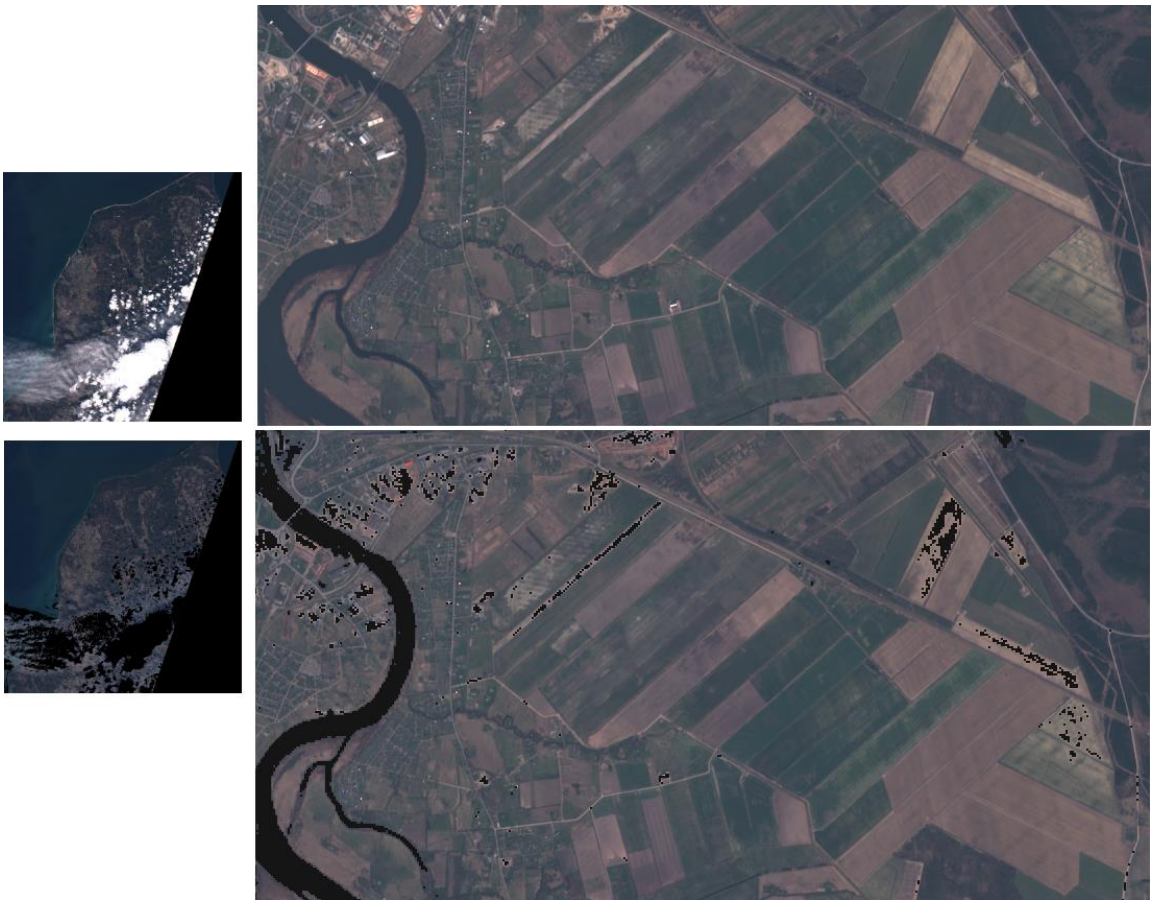


5.4. attēls **Mākoņu detektēšanas rezultāts salīdzinot ar sākotnējo attēlu, Rīga**

Attēlā augšējie attēli ir sākotnējie, bet apakšējie ir pēc mākoņu izņemšanas. Kreisajā pusē esošie ir visas datu granulas attēli, bet labajā pusē ir pietuvināts fragments Rīgā. Kā redzams, tad visā datu granulā izskatās, ka lielākā daļa mākoņu ir izņemts. Pietuvinātajā fragmentā algoritmam bija problēmas ar cilvēku veidotām struktūrām, kuras atstarojas ar baltu krāsu, izteikti redzams,

piemēram, gaišas krāsas ēku gadījumos. Redzams arī, ka algoritmam bija problēmas ar ūdens tilpnēm.

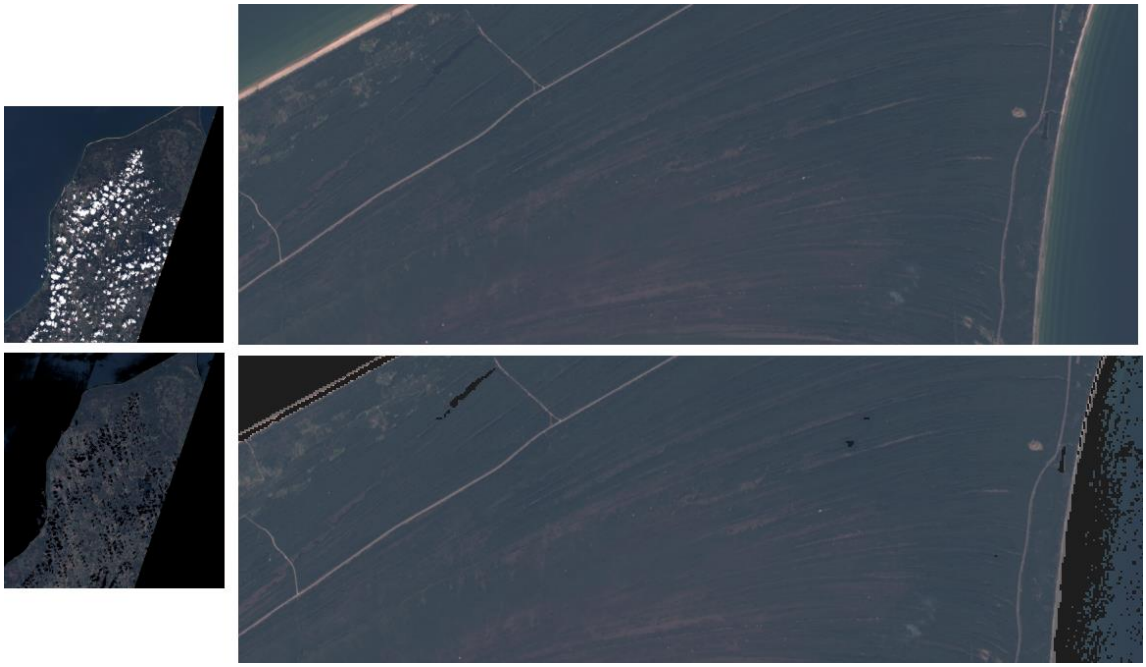
Pārējie intereses reģioni atradās, galvenokārt, virs mežiem, pļavām, lauksaimniecības laukiem, jūras, pilsētām Kurzemes reģionā. Attēlā zemāk redzams paraugs fragmentam no reģiona, kas atrodas virs Ventspils.



5.5. attēls **Mākoņu detektēšanas rezultāts salīdzinot ar sākotnējo attēlu, Ventspils**

Attēlā augšējie attēli ir sākotnējie, bet apakšējie ir pēc mākoņu izņemšanas. Kreisajā pusē esošie ir visas datu granulas attēli, bet labajā pusē ir pietuvināts fragments pie Ventspils. Pietuvinātajā fragmentā redzams, ka algoritms relatīvi nedaudz kļūdījās lauksaimniecības lauku teritorijā. Izteikti redzams, ka pilsētas reģionā ir krietni vairāk nekorekti izņemti fragmenti. Redzams, ka vietām arī ceļu struktūras ir nekorekti izņemtas.

Jūras krastu zonas, mežu teritorijas paraugs redzams attēlā zemāk.



5.6. attēls **Mākoņu detektēšanas rezultāts salīdzinot ar sākotnējo attēlu, Kolka**

Attēlos redzams, ka algoritms pieļāva maz kļūdu mežu klātajos reģionos. Algoritmam vairāk sagādā problēmas jūras krastu reģioni, baltās smiltis.

Tika realizēts algoritms, kas ļāva novērtēt skaidrās debess pikseļu daudzumu intereses reģionos, kas atrodas satelīta attēlos, kuriem tika veikta mākoņu detektēšana. Algoritms veica novērtēšanu apstrādājot intereses reģionus pēc uzņemšanas laika dilstošā secībā, datu daudzuma rēķināšanai tika izmantotas izrēķinātās skaidrās debess matricas, kas iteratīvi tika apvienotas no vairākām attēlu maskām.

Līdzīgs algoritms bija nepieciešams arī gala attēla izveidošanai, šim algoritmam papildus bija nepieciešams izgūt interesējošos reģionus no RGB satelīta attēliem izmantojot aprēķinātās maskas. Darba 4.8. attēlā ir iespējams apskatīt apvienošanas algoritma rezultāta piemēru.

Autors piedāvāja arī risinājumu kā veikt jau uzģenerēta apvienotā satelīta attēla periodisku apvienošanu. Algoritmam pietiek iegūt pašreiz aktuālāko satelīta attēlu noklājumu, tajā detektēt visus nevēlamos reģionus, piemēram, mākoņi, ēnas utml. Pēc tam jāizmanto iegūtā maska, lai pārnestu pārējos datus uz iepriekš apstrādātā attēlu.

Praksē tika pielietoti attēlu uzlabošanas algoritmi, lai apvienotais satelītu attēls izskatītos viendabīgāks. Tika izmantots histogrammu saskaņošanas algoritms satelīta attēlu apvienošanas procesā, lai panāktu konsekventāku gala attēla izskatu. Autors aprakstīja potenciālās problēmas un iespējamos risinājumus, kurus varētu pielietot, lai risinātu problēmas, kas rodas, kad tiek pielietots

minētais algoritms. Attēlā 4.9. ir iespējams apskatīt iegūto rezultātu, kad tika pielietota krāsu saskaņošana.

Tika detāli izpētīts kā ģenerētos attēlus pēc tam tālāk ir iespējams izmantot un publicēt citās ĢIS sistēmās. Tika praktiski integrēti ģenerētie faili tīmekļa lietotnes interaktīvajā kartē izmantojot WMS servisu, kas tika izveidots integrējot ģenerētos failus Geoserver risinājumā. Sistēmas veidotajiem failiem tika izveidoti speciāli ģeogrāfiskās informācijas piesaistes faili, lai failus varētu atvērt gan ar darbvirsmas programmatūru QGIS, gan lai varētu tos integrēt Geoserver risinājumā.

Darba trešajā daļā tika sniegts ieskats par eksistējošiem risinājumiem, kas piedāvā lietotājam apstrādātus Sentinel-2 datu produktus vai servisus, kas ir līdzīgi darbā realizētajai sistēma. Apskatītais SENTINELHub risinājums nodrošina gala lietotājam gatavus OGC servisus, piemēram, WMS, kurus var uzreiz lietot savos ĢIS risinājumos. Servisu datu avoti ir Sentinel-2 dati, kuri nav nekā speciāli apstrādāti satur, piemēram, mākoņus. Komerciāliem nolūkiem risinājums ir par maksu. Šī darba ietvaros tika sasniegts līdzīgs rezultāts izmantojot Geoserver atvērtā koda risinājumu, lai publicētu Sentinel-2 ģeoreferencētus attēlus kā WMS servisu. Netika līdz galam noprogrammēta servisa automātiska atjaunošana, bet process ir skaidrs un pietiekoši viegli realizējams nākotnē.

Apskatītie EOxCloudless un SIA "Baltic Satellite Service" risinājumi bija advancētāki salīdzinot ar SENTINELHub risinājumu. Abi risinājumi ne tikai piedāvā gatavus datu servisus, bet arī to, ka servisos un datos ir aizvietoti mākoņu reģioni ar vecākiem satelīta datiem, kuros mākoņu nebija. Šī darba ietvaros realizētā sistēma ieguva tuvus rezultātus šo risinājumu piedāvātajiem servisiem. Tika sasniegta kopēju attēlu izveide, kas nesatur mākoņu reģionus. Darbā netika veikta lielu intereses reģionu ģenerēšana kā abos pieminētajos risinājumos, kas autora prāt vēl prasītu papildus izstrādi dažādu problēmu risināšanai, piemēram, vairāku ģeogrāfiski nodalītu reģionu vizuāla izskata saskaņošana. Apskatītās sistēmas piedāvā arī papildus aprakstošo informāciju par ģenerētajiem attēliem, piemēram, ir iespējams pateikt no kādiem sākotnējiem datiem attēli ir ģenerēti. Šī darba ietvaros netika realizēta šāda iespēja, bet autora prāt tas arī nebūtu pārāk sarežģīti realizējams, jo izveidotā risinājuma datubāzē tiek glabāta sākotnējo datu informācija, kas tika izmantota rezultāta ieguvē, atliek tikai šos datus papildināt un pievienot lietotājam lietojamā servisā.

SECINĀJUMI

Pēc darba izstrādes ir skaidrs, ka ir iespējams bez ļoti lieliem cilvēka un skaitļošanas resursiem realizēt praktisku sistēmu, kura spēj iegūt aktuālus satelītu datus, kuros nav tādi nevēlami objekti kā mākoņi. Datus var iegūt un lietot bez maksas gan zinātniskiem, gan komerciāliem mērķiem.

Mākoņu detektēšana satelīta datos ir realizējama ar samērā vienkāršu algoritmu palīdzību, to sniegtie rezultāti ir pietiekoši apmierinoši. Labos algoritmu rezultātus noteikti arī ietekmē tas, ka pašreizējo satelītu programmu analizējamo datu kopa ir ļoti laba, tajā ir speciāli izvēlēti un iekļauti datu spektri, kas ļauj kvalitatīvi veikt šāda veida apstrādi.

Apstrādāt Sentinel-2 satelīta datu kopu nav pietiekoši triviāls uzdevums, pie apstrādes ir jāņem vērā tādi apgrūtinoši faktori kā attēlu dažādās ģeogrāfiskās koordinātu sistēmas, kas būtiski apgrūtina attēlu apstrādi, ja vēlas analizēt attēlus uz koordināšu sistēmu robežām, kur dati var pārklāties, bet koordinātu sistēmas var atšķirties. Lai korekti varētu apstrādāt šādas situācijas ir jāveic attēlu transformāciju uz vienotu koordināšu sistēmu, kas ir netriviāla darbība, iespējams izmainītu attēlu augstumu un platumu, ieviestu papildus datu reģionus, kuros nav datu, kas viss būtu jāņem vērā tālākā apstrādē. Darbu papildus apgrūtina arī attēlu spektru datu izšķirtspēju atšķirības, nepieciešamība veikt datu palielināšanu vai samazināšanu, lai varētu veikt kopēju analīzi. Eksistē datu reģioni attēlos, kuros nav dati, to speciāla apstrāde ir nepieciešama daudzos apstrādes posmos.

Apvienotajiem attēliem ir diezgan liela vizuālā atšķirība, kas rodas no dažādiem ārējiem vides faktoriem. Ir iespējams pielietot attēlu apstrādē pieejamus algoritmus, kas samazina vizuālo atšķirību, bet var ieviest jaunas problēmas, kā piemēram attēla krāsas pazaudēšana specifiskos reģionos, kuras nepieciešams risināt.

Ģenerētos datus pēc tam ir iespējams relatīvi vienkārši publicēt un iesaukt citos ĢIS risinājumos. Šos ĢIS risinājumus varētu uzskatīt par gala lietotāju lietotnēm, kuros varētu integrēt izstrādātās sistēmas ģenerētos attēlus.

IZMANTOTĀ LITERATŪRA UN AVOTI

1. Jacob Høxbroe, Jeppesen Rune Hylsberg, Jacobsen Fadil Inceoglu, Thomas Skjødeberg Toftegaard "A cloud detection algorithm for satellite imagery based on deep learning" *Remote Sensing of Environment*, Volume 229, August 2019, Pages 247-259 [Tiešsaiste – 13.01.2020]. Pieejams tīmeklī: https://reader.elsevier.com/reader/sd/pii/S0034425719301294?token=FCB7B88CC1224F95BFD29FD9C8D223685291898C0CC22995CB71E6FAC2A5E9B1A9E1797E9D290E33433E503EB94885F1
2. Kārlis Podnieks "Modeļi datizracē: parametriskie modeļi, estimatori." [Tiešsaiste – 12.05.2020]. Pieejams tīmeklī: http://podnieks.id.lv/slides/mining/metodes5%20EST.pdf
3. André Hollstein, Karl Segl, Luis Guanter, Maximilian Brell, Marta Enesco "Ready-to-Use Methods for the Detection of Clouds, Cirrus, Snow, Shadow, Water and Clear Sky Pixels in Sentinel-2 MSI Images" *Remote Sens.*, 8, 666, 18 August 2016 [Tiešsaiste – 14.01.2020]. Pieejams tīmeklī: https://www.mdpi.com/2072-4292/8/8/666
4. LANDSAT 8 satelītu programma [Tiešsaiste – 14.01.2020]. Pieejams tīmeklī: https://www.usgs.gov/land-resources/nli/landsat/landsat-8?qt-science_support_page_related_con=0#qt-science_support_page_related_con
5. Sentinel-2 [Tiešsaiste – 14.01.2020]. Pieejams tīmeklī: https://sentinel.esa.int/web/sentinel/missions/sentinel-2
6. Atmosphere and shading compensation of satellite images [Tiešsaiste – 14.01.2020]. Pieejams tīmeklī: http://www.imagico.de/pov/earth_atmosphere.php
7. Well-known text representation of geometry [Tiešsaiste – 16.01.2020]. Pieejams tīmeklī: https://en.wikipedia.org/wiki/Well-known_text_representation_of_geometry
8. SQL [Tiešsaiste – 16.01.2020]. Pieejams tīmeklī: https://en.wikipedia.org/wiki/SQL
9. cron [Tiešsaiste – 16.01.2020]. Pieejams tīmeklī: https://en.wikipedia.org/wiki/Cron
10. Sentinel-2 datu atjaunošanas biežums [Tiešsaiste – 17.01.2020]. Pieejams tīmeklī: https://sentinel.esa.int/web/sentinel/user-guides/sentinel-2-msi/revisit-coverage

11. Sentinel-2
[Tiešsaiste – 18.01.2020]. Pieejams tīmeklī:
https://en.wikipedia.org/wiki/Sentinel-2
12. Global cloud cover map
[Tiešsaiste – 18.01.2020]. Pieejams tīmeklī:
https://www.climatecentral.org/news/13-years-clouds-nasa-map-18985
13. Web Map Service Implementation Specification
[Tiešsaiste – 18.01.2020]. Pieejams tīmeklī:
http://portal.opengeospatial.org/files/?artifact_id=1081&version=1&format=pdf
14. GIS
[Tiešsaiste – 18.01.2020]. Pieejams tīmeklī:
https://en.wikipedia.org/wiki/Geographic_information_system
15. Tiles
[Tiešsaiste – 18.01.2020]. Pieejams tīmeklī:
https://wiki.openstreetmap.org/wiki/Tiles
16. Web Mercator projection
[Tiešsaiste – 18.01.2020]. Pieejams tīmeklī:
https://en.wikipedia.org/wiki/Web_Mercator_projection
17. Web Mercator projection
[Tiešsaiste – 18.01.2020]. Pieejams tīmeklī:
https://www.maptiler.com/google-maps-coordinates-tile-bounds-projection/
18. Mapproxy
[Tiešsaiste – 18.01.2020]. Pieejams tīmeklī:
https://mapproxy.org/
19. OpenLayers
[Tiešsaiste – 18.01.2020]. Pieejams tīmeklī:
https://openlayers.org/
20. Copernicus programme
[Tiešsaiste – 18.01.2020]. Pieejams tīmeklī:
https://en.wikipedia.org/wiki/Copernicus_Programme
21. Copernicus programme brochure
[Tiešsaiste – 18.01.2020]. Pieejams tīmeklī:
https://www.copernicus.eu/sites/default/files/documents/Copernicus_brochure_EN_web_Oct2017.pdf
22. SAR
[Tiešsaiste – 18.01.2020]. Pieejams tīmeklī:
https://lv.wikipedia.org/wiki/Sintez%C4%93t%C4%81s_apert%C5%ABras_radars
23. MSI overview
[Tiešsaiste – 18.01.2020]. Pieejams tīmeklī:
https://earth.esa.int/web/sentinel/technical-guides/sentinel-2-msi/msi-instrument

24. Sentinel-3
[Tiešsaiste – 19.01.2020]. Pieejams tīmeklī:
https://en.wikipedia.org/wiki/Sentinel-3
25. Spalvmākoņi
[Tiešsaiste – 19.01.2020]. Pieejams tīmeklī:
https://lv.wikipedia.org/wiki/Spalvm%C4%81ko%C5%86i
26. Cirrus clouds
[Tiešsaiste – 19.01.2020]. Pieejams tīmeklī:
http://ww2010.atmos.uiuc.edu/(Gh)/guides/mtr/cld/cldtyp/hgh/crs.rxml
27. Sentinel-2 cirrus cloud detection band
[Tiešsaiste – 19.01.2020]. Pieejams tīmeklī:
https://labo.obs-mip.fr/multitemp/4109/
28. Polar orbit
[Tiešsaiste – 19.01.2020]. Pieejams tīmeklī:
https://en.wikipedia.org/wiki/Polar_orbit
29. Sentinel-2 overview
[Tiešsaiste – 19.01.2020]. Pieejams tīmeklī:
https://sentinel.esa.int/web/sentinel/missions/sentinel-2/overview
30. Sentinel-2 data format
[Tiešsaiste – 19.01.2020]. Pieejams tīmeklī:
https://sentinel.esa.int/web/sentinel/user-guides/sentinel-2-msi/data-formats
31. Copernicus Open Access Hub
[Tiešsaiste – 19.01.2020]. Pieejams tīmeklī:
https://scihub.copernicus.eu/
32. Lēmuma koki
[Tiešsaiste – 19.01.2020]. Pieejams tīmeklī:
http://podnieks.id.lv/slides/mining/metodes1%20DT.pdf
33. Decision tree learning
[Tiešsaiste – 19.01.2020]. Pieejams tīmeklī:
https://en.wikipedia.org/wiki/Decision_tree_learning
34. Overfitting
[Tiešsaiste – 20.01.2020]. Pieejams tīmeklī:
https://www.investopedia.com/terms/o/overfitting.asp
35. Autora realizētais sistēmas pirmkods
[Tiešsaiste – 06.05.2020]. Pieejams tīmeklī:
https://bitbucket.org/mzervens/magistra_darbs/src/master/
36. QGIS programmatūra
[Tiešsaiste – 06.05.2020]. Pieejams tīmeklī:
https://qgis.org/en/site/

37. World file
[Tiešsaiste – 06.05.2020]. Pieejams tīmeklī: https://en.wikipedia.org/wiki/World_file
38. TypeScript
[Tiešsaiste – 06.05.2020]. Pieejams tīmeklī: https://www.typescriptlang.org/
39. Flask
[Tiešsaiste – 06.05.2020]. Pieejams tīmeklī: https://flask.palletsprojects.com/en/1.1.x/
40. Geoserver
[Tiešsaiste – 06.05.2020]. Pieejams tīmeklī: http://geoserver.org/
41. API HUB Full Text Search
[Tiešsaiste – 06.05.2020]. Pieejams tīmeklī: https://scihub.copernicus.eu/userguide/FullTextSearch
42. PostGIS
[Tiešsaiste – 06.05.2020]. Pieejams tīmeklī: https://postgis.net/
43. ST_Difference
[Tiešsaiste – 06.05.2020]. Pieejams tīmeklī: https://postgis.net/docs/ST_Difference.html
44. ST_Equals
[Tiešsaiste – 06.05.2020]. Pieejams tīmeklī: https://postgis.net/docs/ST_Equals.html
45. ST_IsEmpty
[Tiešsaiste – 06.05.2020]. Pieejams tīmeklī: https://postgis.net/docs/ST_IsEmpty.html
46. NumPy
[Tiešsaiste – 06.05.2020]. Pieejams tīmeklī: https://numpy.org/
47. Sentinel-2 Products Specification Document
[Tiešsaiste – 06.05.2020]. Pieejams tīmeklī: https://sentinel.esa.int/documents/247904/685211/Sentinel-2-Products-Specification-Document
48. EPSG:32634 WGS 84 / UTM zone 34N
[Tiešsaiste – 07.05.2020]. Pieejams tīmeklī: https://epsg.io/32634
49. OpenStreetMap
[Tiešsaiste – 07.05.2020]. Pieejams tīmeklī: https://www.openstreetmap.org/

50. Esri
[Tiešsaiste – 07.05.2020]. Pieejams tīmeklī:
https://en.wikipedia.org/wiki/Esri

51. ESRI PRJ File
[Tiešsaiste – 07.05.2020]. Pieejams tīmeklī:
https://vsp.pnnl.gov/help/Vsample/ESRI_PRJ_File.htm
52. OGC
[Tiešsaiste – 07.05.2020]. Pieejams tīmeklī:
https://www.ogc.org/
53. Geoserver Stores
[Tiešsaiste – 07.05.2020]. Pieejams tīmeklī:
https://docs.geoserver.org/latest/en/user/data/webadmin/stores.html
54. Geoserver ImageMosaic
[Tiešsaiste – 07.05.2020]. Pieejams tīmeklī:
https://docs.geoserver.org/stable/en/user/data/raster/imagemosaic/
55. Multi-resolution imagery with reprojection
[Tiešsaiste – 07.05.2020]. Pieejams tīmeklī:
https://docs.geoserver.org/stable/en/user/data/raster/imagemosaic/tutorial.html#multi-crs-mosaic
56. Universal Transverse Mercator coordinate system
[Tiešsaiste – 07.05.2020]. Pieejams tīmeklī:
https://en.wikipedia.org/wiki/Universal_Transverse_Mercator_coordinate_system
57. EPSG:3857
[Tiešsaiste – 07.05.2020]. Pieejams tīmeklī:
https://epsg.io/3857
58. GDAL
[Tiešsaiste – 07.05.2020]. Pieejams tīmeklī:
https://gdal.org/
59. gdalwarp
[Tiešsaiste – 07.05.2020]. Pieejams tīmeklī:
https://gdal.org/programs/gdalwarp.html
60. Histograms - 1 : Find, Plot, Analyze !!!
[Tiešsaiste – 08.05.2020]. Pieejams tīmeklī:
https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_histograms/py_histogram_begins/py_histogram_begins.html
61. Mark Grundland, Neil A. Dodgson “Color histogram specification by histogram warping”, *Proc. SPIE 5667, Color Imaging X: Processing, Hardcopy, and Applications*, (17 January 2005):

[Tiešsaiste – 08.05.2020]. Pieejams tīmeklī:
http://www.eyemaginary.com/Rendering/ColorHistogramSpecification.pdf
62. Histogram Matching
[Tiešsaiste – 08.05.2020]. Pieejams tīmeklī:
http://paulbourke.net/miscellaneous/equalisation/
63. Maģistra darba izstrādes un aizstāvēšanas metodiskie norādījumi
[Tiešsaiste – 10.05.2020]. Pieejams tīmeklī:
https://estudijas.lu.lv/mod/resource/view.php?id=88291
64. Sentinel-2 cloudless
[Tiešsaiste – 14.05.2020]. Pieejams tīmeklī:
https://s2maps.eu/
65. Sentinel-2 satelītattēlu pamatkartes mozaīkas serviss
[Tiešsaiste – 14.05.2020]. Pieejams tīmeklī:
https://www.baltsat.lv/jaunumi/params/post/1539800/sentinel-2-satelitattelu-pamatkartes-mozaikas-serviss
66. SENTINELHub playground
[Tiešsaiste – 14.05.2020]. Pieejams tīmeklī:
https://apps.sentinel-hub.com/sentinel-playground
67. SENTINELHub Capabilities
[Tiešsaiste – 14.05.2020]. Pieejams tīmeklī:
https://www.sentinel-hub.com/develop/capabilities

Maģistra darbs “Mākoņu noņemšana no satelīta attēliem”

Ar savu parakstu apliecinu, ka pētījums veikts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors: Matīss Zērvēns _____
(Autora paraksts un datums)

Ar savu parakstu apliecinu, ka esmu lasījis augstāk minēto maģistra darbu un atzīstu to par **p i e m ē r o t u / n e p i e m ē r o t u** (nevajadzīgo svītrot) aizstāvēšanai Latvijas Universitātes datorzinātņu maģistrantūrā.

Darba vadītājs: docents Dr. dat. Kārlis Freivalds _____
(Vadītāja paraksts un datums)

Darbs iesniegts maģistratūras sekretariātā_____.
(Iesniegšanas datums)

Ar šo es apliecinu, ka darba elektroniskā versija ir augšupielādēta LU informatīvajā sistēmā.

Studiju metodiķe: Ella Arša _____.
(Metodiķes paraksts)

Recenzents: profesors Dr. dat. Juris Vīksna
(Akad.amats, zin.grāds, vārds, uzvārds)

Darbs aizstāvēts maģistra gala pārbaudījuma komisijas sēdē

_____prot. Nr. _____
(Darba aizstāvēšanas datums)

Komisijas sekretārs: _____
(Sekretāra paraksts)