MODUS 3D
JOURNAL

# SEMANTTISTEN PISTEPILVIEN TEHOKAS TALLENNUS BIG DATA-TYYPPISILLÄ PYTHON-TYÖKALUILLA

**Sami El-Mahgary[a], Juho-Pekka Virtanen[ab], ja Hannu Hyyppä[ab]**

[a] Aalto-yliopisto, Rakennetun ympäristön laitos, Otakaari 4, 02150 Espoo;
[b] Maanmittauslaitos, Geodeetinrinne 2, 02430 Masala.

Tiivistelmä: Laserkeilauksella kerättyihin pistepilviin voidaan liittää myös semanttista tietoa, joka lisää pistepilven käyttöominaisuuksia merkittävästi. Semanttisen tiedon tallentaminen mahdollistaa mielenkiintoisia tietokantakyselyitä, jotka lisäävät pistepilven käyttömahdollisuuksia huomattavasti. Jos pistepilvi kuvaisi vaikkapa erilaisia rakennuksia kaupungin ydinkeskustassa, niin esimerkkinä yksinkertaisesta semanttisesta kyselystä olisi kysely, jossa haetaan kaikki sellaiset pisteet (ja vain sellaiset), jotka liittyvät annetulla alueella olevaan museoon. Tässä lyhennelmässä tarkastellaan yksinkertaista Python-ohjelmoinnilla toteutettua menetelmää, jolla voidaan helposti tallentaa semanttinen pistepilvi niin, että siihen voidaan kohdistaa erilaisia nopeita semanttisia kyselyitä.

ARTIKKELIN HISTORIA

Vastaanotettu 30.3.2020, hyväksytty 12.6.2020, julkaistu 21.12.2020.

AVAINSANAT

Pistepilvi; Python-ohjelma; semanttinen luokittelu

## 1. JOHDANTO

Useat 3D-kartoituksen menetelmät, kuten fotogrammetria ja laserkeilaus, tuottavat kohteesta kolmiulotteisen pistepilven. Riippuen sekä kohteen laajuudesta että käytetystä tarkkuudesta, yksi pistepilvi voi sisältää satoja tuhansia, miljoonia tai peräti satoja miljoonia pisteitä. Jokaiseen pistepilvitiedoston pisteeseen liittyy sen sijainnin eli (X, Y, Z) koordinaattien lisäksi useimmiten myös erilaista lisätietoa. Tällaista lisätietoa ovat mm. kunkin pisteen väritiedot, intensiteetti sekä pisteen semanttinen luokitus.

Erilaisia semanttisia luokituksia on useita, tässä keskitytään Hackel ja kollegat (2017) luomaan luokitukseen, joka on kahdeksanpohjainen (Taulukko 1). Erityistä Hackel ja kollegat (2017) käyttämässä luokituksessa on se, että jokainen pistepilvitiedosto P jakaantuu kahdeksi tiedostoksi P1 ja P2. Jako tapahtuu niin, että yhteen tiedostoon P1 sisällytetään (X, Y, Z) paikkatiedot sekä kaikki muu lisätieto semanttista luokitusta lukuun ottamatta. Toiseen tiedostoon P2 taas sisällytetään pelkästään kunkin pisteen semanttinen luokitus, joka siis ilmaistaan kullekin (X, Y, Z) pisteelle arvolla 1-8 (kts. Taulukko 1).

Jos pistettä ei ole pystytty luokittelemaan, saa se arvon ’0’, joten käytännössä luokituksessa esiintyvät arvot välillä 0-9.

*Taulukko 1: Hackel ja kollegat (2017) käyttämä kahdeksanpohjainen semanttisen pistepilven luokittelujärjestelmä.*

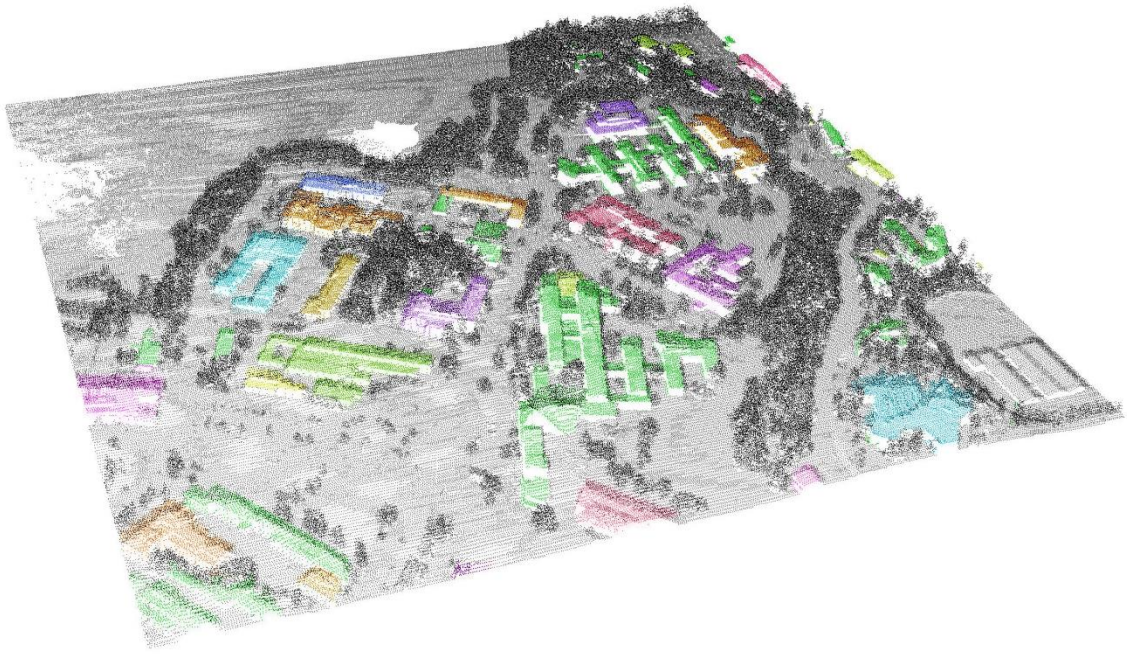
| Luokka-tunnus | Viittaus |
|---|---|
| 1 | Rakennettu maanpinta (esim. Katukäytävät) |
| 2 | Luonnollinen maanpinta (tyypillisesti nurmikko, nurmikentät) |
| 3 | Korkea kasvillisuus (puut sekä suuret pensaat) |
| 4 | Matala kasvillisuus (kukat sekä pienet, alle 2m. Korkeat pensaat) |
| 5 | Päärakennukset (kerrostalot, asemat, kirkot, koulut, jne.) |
| 6 | Pienet rakennukset (pankit, suihkulähteet, jne.) |
| 7 | Erilaiset tunnistamattomat liikkuvat kohteet |
| 8 | Autot sekä yhdistelmäajoneuvot |

## 1.1 LUOKITUS KUN KYSEESSÄ ON VAIN RAKENNUKSIA

Yksinkertainen esimerkki semanttisesta luokituksesta on tietyn alueen rakennusten luokitus. Maanmittauslaitos tuottaa esimerkiksi rakennuskohteista julkisia, automaattisesti luokiteltuja semanttisia pistepilviä, joissa jokainen piste on luokiteltu muutamaan perusluokkaan kuten `matala kasvillisuus’ tai `maanpinta’ (MML, 2020). Tällaisia pistepilviä voidaan jatkokäsitellä esimerkiksi paikkatieto-ohjelmistoilla liittäen niihin muista tietoaineistoista saatavaa dataa, esimerkiksi maastotietokannan rakennuspolygonien rakennustunnukset. Tällöin saadaan pistepilvi, jossa jokaisella rakennuksia kuvaavalla pisteellä on myös pisteen yksittäiseen rakennukseen sitova tunniste, käytännössä yksikäsitteinen kokonaisluku. Laserkeilaimen rakennuksista tuottamaa pistepilveä voidaan varsin suoraviivaisesti rikastuttaa liittämällä kuhunkin rakennukseen tunnusluku sekä myös muuta semanttista tietoa rakennuksesta (esim. rakennusvuosi) kuten eräässä van Oosteromin ohjaamassa lopputyössä (Joosten, 2018) on kuvattu.

Jos käytössämme on yksinkertainen rakennusluokittelu, jossa tyydytään luokittelemaan ainoastaan kohteen eri rakennukset, tällöin kaikille muille alueen kohteille (esim. kasvillisuus, maanpinta, vesistöt, jne.) määrätään luokittelutunnukseksi ’0’. Esimerkkinä yksinkertaisesta rakennusluokittelusta on Kuva 1, joka esittää Espoon Otaniemessä olevaa rakennusrykelmää. Koska pistepilven näytteistystä on harvennettu, viitataan siihen tässä Otaniemen suppeana pistepilvenä, lyhyemmin OTA-Sup. Kyseinen OTA-Sup perustuu Maanmittauslaitoksen tuottamaan pistepilveen Otaniemestä (Maanmittauslaitos, 2019), joka on saatu ilmalaserkeilauksella (ALS), jossa pistetiheys on vähintään 0,5 per neliömetri. Rakennusrykelmä käsittää harvennuksen jälkeen kokonaisuudessaan 84 rakennusta, joista jokaiselle on määrätty oma erillinen värisävynsä Kuvassa 1.

Kuvitteellinen ongelma, joka tässä halutaan ratkaista, on se, miten voidaan nopeasti selvittää, mitkä pisteet kuuluvat tiettyyn rakennukseen. Eli kun tiedossamme on yksikäsitteinen rakennuksen tunnus, kuinka saamme mahdollisimman pienellä viiveellä luettua tietokoneen muistiin kyseisen rakennuksen kaikki pisteet? Tällaista tarkkaan määriteltyä halutunlaisen aineiston poimintaa pistepilvestä kutsumme tässä kyselyksi, lainataksemme tietokantojen yhteydessä käytettyä terminologiaa. Seuraavassa kappaleessa esitetään ratkaisu juuri mainitulle ongelmalle käyttäen Otaniemen rakennusrykelmää OTA-Sup esimerkkinä.



*Kuva 1: Otaniemen rakennusrykelmä: esimerkki pistepilveen kohdistetusta rakennuspohjaisesta semanttisesta luokittelusta. Kohteena on Otaniemien alue Espoossa, Maanmittauslaitoksen kuvaamana (Maanmittauslaitos, 2019). Pistepilven näytteistystä on harvennettu (eng. downsampled) niin, että se käsittää 368 408 pistettä. Erilaisia rakennuksia on kuvassa yhteensä 84 kpl, joista jokainen on esitetty omalla värisävyllään ja joista jokainen pitää sisällään keskimäärin n. 500 pistettä.*

## 2. MENETELMÄ

Tässä kappaleessa selvitetään lyhyesti käytetyn menetelmän perusidea, jolla tallennetaan pistepilvi, joka perustuu edellä mainittuun yksinkertaiseen rakennusluokitteluun. Menetelmä on toteutettu Python-ohjelmoinnilla, mutta aluksi käydään läpi menetelmän perusidea ja sen vaatimukset pistepilven esitystavasta.

### 2.1 OLETUKSIA PISTEPILVEN ESITYSTAVASTA

Kehitetty Python pilkkomis- sekä kyselyalgoritmi olettaa seuraavat seikat annetusta pistepilvestä P:

A. Semanttinen luokittelu perustuu Hackel ja kollegat (2017) luokitteluun.
B. Pistepilvi on jaettu kahteen tiedostoon, P1 sekä P2, jotka kumpikin sisältävät yhtä monta riviä ja joissa rivien järjestys tiedostoissa P1 ja P2 pidetään samana.
C. Tiedosto P1 koostuu riveistä, joista kukin rivi sisältää ainakin kunkin pisteen (X, Y, Z) koordinaatit,
D. Tiedosto P2 koostuu riveistä, joista kukin rivi sisältää ainoastaan yhden luvun väliltä 0-9, jossa nolla ilmaisee sen, että pistettä ei ole voitu luokitella.

## 2.2 HAKEMISTOJEN KÄYTTÖ INDEKSOINTIMENETELMÄNÄ

Idea hakemistojen käyttämiseen indeksoinnissa juontaa juurensa Aalto-yliopistossa tehtyyn tutkimukseen (El-Mahgary, Soisalon-Soininen, Orponen, Rönnholm & Hyyppä, 2019). Ideana on, että kukin semanttinen luokitus tallentuu omaan erilliseen hakemistoonsa. Näin ollen OTA-Sup pistepilvi tallentuisi yhteensä 84 eri hakemistoon, jossa jokainen hakemisto on nimetty sen rakennuksen tunnuksen mukaan, joka tallentuu kyseiseen hakemistoon. Tarkemmin sanottuna jokaisen erillisen rakennuksen kohdalla pistepilvitiedosto P1 tallentuu omaan hakemistoonsa binääritiedostona P'1. Binääritiedosto P'1n tarkoituksena on lähinnä vain nopeuttaa tiedon käsittelyä. Menetelmän välittömänä etuna on, että semanttista pistepilvitiedostoa P2 ei enää tarvita, koska siinä oleva semanttinen tieto on siirtynyt hakemistoihin: jokainen hakemisto sisältää vain yhden tietyn rakennuksen tiedot.

Lisäksi tallennuksessa on huomioitava, että koska kyseessä on pelkästään rakennuksia, ja koska noudatamme Hackel ja kollegat (2017) semanttista luokittelua, kaikki edellä mainitut 84 hakemistoa tallentuvat yhden ja saman semanttisen päähakemiston alle, jonka nimi sisältää semanttisen luokan tunnuksen. Tässä oletamme, että päähakemiston nimi on muotoa `SEM_5', jossa tunnus viisi on siis se semanttinen luokka, joka viittaa rakennuksiin. Äsken mainitun OTA-Sup pistepilven erilliset 84 hakemistoa tallentuvat näin ollen kaikki tämän hakemiston `SEM-5' alle. Koska on mahdollista, että käyttäjä haluaa tehdä sellaisia pistepilvikyselyitä, jotka liittyvät ainoastaan annettuihin paikkakoordinaatteihin eivätkä tiettyyn rakennukseen, tallennamme suoraan hakemiston `SEM-5' alle myös yhden binäärisen pistepilvitiedoston P'1, joka sisältää nyt paikkatiedot kaikista niistä rakennuksista, jotka kuuluvat pistepilveen OTA-Sup.

## 2.3 PYTHON-KIELINEN TOTEUTUS

Python 3.0-ohjelmointikielellä toteutettu algoritmi hyödyntää pandas-kirjastoa (McKinney, 2018) sekä siinä olevaa dataframe tietorakennetta (McKinney, 2020). Dataframe on suurten tietomäärien käsittelyyn tarkoitettu tietorakenne, joka mahdollistaa tietokoneen muistissa tapahtuvan erittäin nopean aineiston lajittelun ja sen poiminnan. Toisin kuin esimerkiksi Pythonin numpy-tietorakenne, pandas sallii kunkin sarakkeen olevan eri tietotyyppiä kuin naapurisarakkeen. Tässä suhteessa pandas muistuttaa rakenteeltaan taulukkolaskennasta tuttua taulukkoa riveineen ja sarakkeineen. Pandasin muistissa olevaa tietoa voidaan myös tallentaa levylle ja siirtää levyltä keskusmuistille erittäin nopeilla lisärutiineilla.

Toteutettu algoritmi on käytännössä Python-ohjelma, joka muuntaa edellä mainitun P1-tiedoston binääriseksi tiedostoksi käsittelyn nopeuttamiseksi. Pandas-kirjasto ei itsessään sisällä mitään ominaisuuksia tiedoston tallentamiseen levylle, mutta siihen voidaan liittää useampia erilaisia

tiedonkäsittelyyn tarkoitettuja kirjastoja, joista valitsimme tehokkaan ja kevyen feather-kirjaston. Kun Python-ohjelmalle annetaan tietyn pistepilven muodostavaa kaksi tiedostoa P1 sekä P2, ohjelma muuntaa tekstimuodossa olevan tiedoston P1 binääriseksi feather-muodossa olevaan tiedostoon P'1. Semanttista tietoa sisältävä tiedosto P2 taas muuttuu tarpeettomaksi, koska kaikki samaan semanttiseen luokkaan kuuluva aineisto tallennetaan saman hakemiston alle. Mainittu Python-ohjelma luo jokaiselle tiedostossa P2 olevalle erilliselle rakennukselle hakemiston, jonka nimi perustuu suoraan kyseisen rakennuksen tunnuslukuun. Jos oletamme, että kaikkien hakemistojen nimet alkavat merkkijonolla `SEM_', niin tällöin esimerkiksi hakemisto `SEM_19155' sisältäisi vain yhden P1-tyyppisen tiedoston, jossa olisivat kaikki ne pisteet, jotka liittyvät rakennukseen, jonka tunnusluku on juuri 19155. Kyseiset hakemistot ovat itse asiassa alihakemistoja, jotka luodaan edellä mainitun hakemiston `SEM-5' alaisuuteen. Jokainen rakennukseen liittyvä alihakemisto sisältää siis yhden binäärisen tiedoston P'1 sekä myös yhden erittäin suppean tekstitiedoston, jossa määritellään pelkästään kyseisen tiedoston P'1 (X,Y,Z) koordinaattien ääriarvot (minimi- sekä maksiarvot) hakujen nopeuttamiseksi. Kyseessä on kuitenkin hierarkkinen rakenne, jossa jokaiseen hakemistoon H tallentuu aina yksi laajempi binäärimuotoinen tiedosto P'1, johon on koottu kaikkien niiden binääritiedostojen P'1 pisteet, jotka sijaitsevat hakemiston H alla. Esimerkiksi kyseinen hakemisto `SEM-5' sisältää paitsi rakennusten alihakemistot tiedostoineen, myös yhden laajemman binäärisen tiedoston P'1, joka sisältää tiedot kaikista niistä P'1-tyyppisistä tiedostoista, jotka sijaitsevat missä tahansa hakemiston `SEM-5' alaisuudessa olevassa alihakemistossa.

## 3. SUORITETUT AJOT JA TULOKSET

Rakennusten osalta suoritimme mainitulla pistepilvellä useita erilaisia kyselyitä, joista tässä esitämme kolme erilaista (K1-K3). Kyselyiden K1-K3 tulokset on taulukoitu (Taulukko 2). Yksinkertaisin kysely K1 käsittää kaikkien rakennuksiin liittyvien pisteiden poiminnan, tällaisia pisteitä kertyi 41 865 kpl ja ne kaikki löytyivät suoraan hakemiston `SEM-5' alta sijaitsevasta binäärimuotoisesta tiedostosta P'1. Sekä kyselyt K2 että K3 perustuvat kysely K1n palauttamaan tulokseen. Niiden tarkoituksena on etsiä tietyltä alueelta kaikki rakennuksiin liittyvät pisteet. Kysely K2 määrittelee X ja Y koordinaatteihin perustuvan suorakulmaisen alueen, joka käsittää ainoastaan kyselyn K1 palauttamat pisteet. Vastaavasti kysely K3 määrittelee X ja Y koordinaatteihin perustuvan ympyränmuotoisen alueen, joka niin ikään käsittää ainoastaan kyselyn K1 palauttamat pisteet. Näistä kolmesta kyselystä kysely K3 palautti selvästi vähiten pisteitä (2 556 pistettä) mutta oli silti joukon hitain kysely (42,27ms) johtuen kyselyn monimutkaisuudesta. Kysely K2 taas oli joukon nopein kysely (12,27ms).

*Taulukko 2: Tulokset kun suppea Otaniemen pistepilvi ajettiin kolmella eri kyselyllä. Kysely K1 palautti rakennuksiin liittyviä pisteitä yhteensä 41 865 kpl. Kyselyt K2 ja K3 perustuivat kyselyn K1n palauttamaan pistejoukkoon rajoittamalla sitä joko suorakulmaiselta alalta (K2) tai ympyrän muotoiselta alueelta (K3). Kysely K3 oli joukon hitain kysely, palauttaen 2 556 pistettä runsaassa 42 millisekunnissa.*

| Luokka-tunnus | Kysely K1, jossa haetaan kaikki rakennuksiin liittyvät pisteet.<br>Kesto (ms) | Kysely K2, jossa haetaan rakennuksiin liittyvät pisteet tietyn suorakulmion muotoiselta alalta.<br>Kesto (ms) | Kysely K3, jossa haetaan pisteet ympyrän muotoiselta alalta.<br>Kesto (ms) |
|---|---|---|---|
| Kyselyn palauttama pistemäärä | 41 865 | 19 722 | 2 556 |
| Kyselyn kesto (ms) | 12,34 | 12,27 | 42,27 |

## 3.1 KOKEET SUUREMMALLA PISTEPILVELLÄ

Edellisen testin tulokset olivat varsin kannustavia, mutta herättivät kysymyksen siitä, miten esitetty menetelmä käyttäytyy, kun kyseessä on selvästi suurempi pistepilvi, joka sisältää vastaavasti paljon suuremman määrän rakennuksia. Tämän selvittämiseksi muodostimme Maanmittauslaitoksen aineistosta (Maanmittauslaitos, 2019) toisen semanttisen pistepilven, joka käsittää paitsi Otaniemen alueen myös sen lähiympäristön. Kutsumme näin saatua pistepilveä laajennetuksi pistepilveksi, lyhyemmin OTA-Laaj. Laajennettu OTA-Laaj pistepilvi käsitti yhteensä 8 971 327 pistettä, jotka esittivät 1 940 eri rakennusta. Kyseiseen OTA-Laaj pistepilveen kohdistettiin viisi kyselyä (K1-K5), jotka itse asiassa perustuivat kaikki yhteen ja saman tyyppiseen monimutkaiseen kyselyyn, jonka perusta kuului näin: kun on annettu tietty (X,Y) koordinaateilla rajattu alue Rx, niin mikä on jokaisen rakennuksen Z-koordinaatin maksimiarvo kyseisellä alueella Rx, kun jätetään pois kaksi tiettyä rakennusta? Kyselyissä K1-K5 siis ainoastaan alueen Rx määritelmä vaihteli niin, että kuhunkin kyselyyn liittyi vaihteleva määrä rakennuksia. Kyselyyn K1 liittyi vähiten rakennuksia (61 kpl) ja rakennusten määrä kasvoi edetessä kohti kyselyä K5, johon liittyi suurin määrä rakennuksia, yhteensä 905 kpl kuten Taulukosta 3 näkyy.

Kyselyt K1-K5 toteutettiin vertailun vuoksi kahdella eri Python-menetelmällä. Ensimmäinen menetelmä perustui Pythonin groupby-funktioon kun taas jälkimmäisessä ei käytetty lainkaan groupby funktiota. Menetelmässä, joka nojautui groupby-funktioon luettiin ensin suoraan hakemiston SEM_5 alta sijaitsevalta binääritiedostolta P'1 kaikki rakennuksiin liittyvät pisteet. Kyseiset pisteet siirtyivät näin levyltä muistiin pandas-dataframe tietorakenteeseen, johon sovellettiin groupby funktiota. Samalla määriteltiin, että annetut kaksi rakennusta ohitetaan sekä laskettiin kullekin rakennukselle sen maksimi Z-koordinaattipiste. Kyseisen Python koodin keskeinen osa, joka hyödyntää groupby-funktiota on sangen suppea ja se on esitetty Kuvassa 2.

```
PTS_Block_grouped = Pts_BlockXY.loc[Pts_BlockXY.groupby('s').z.idxmax()]
```

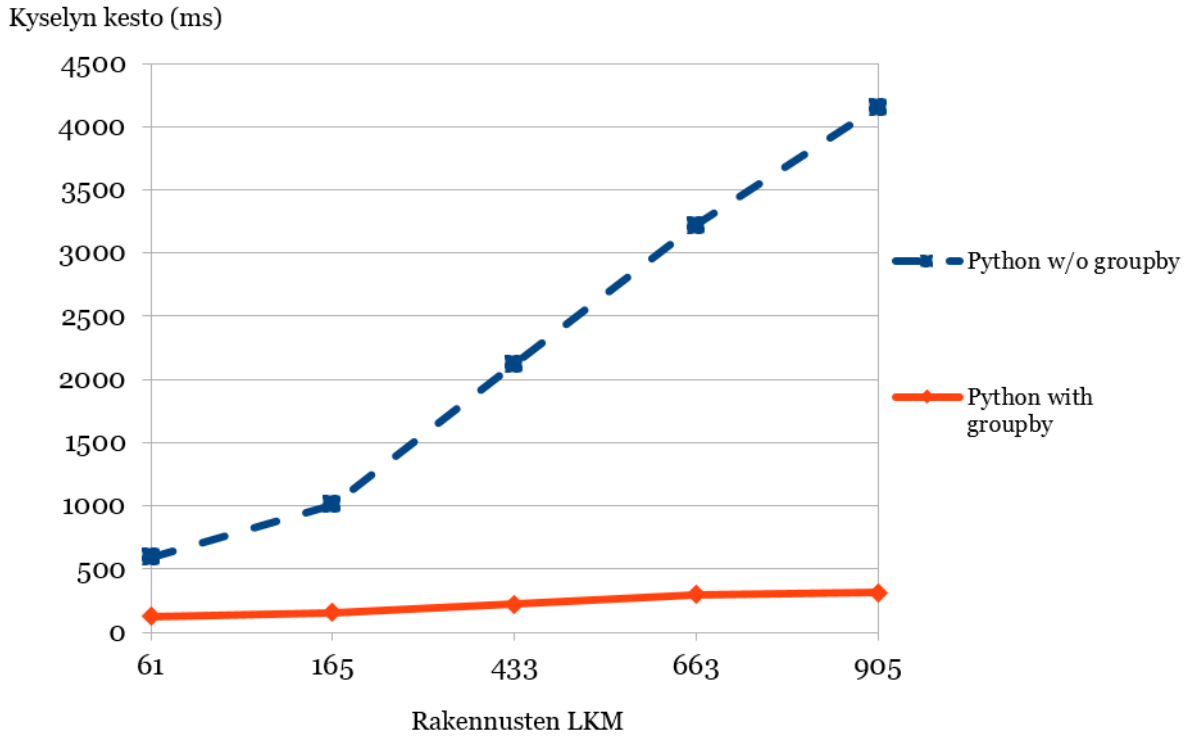
*Kuva 2: Keskeinen osa Python koodista, jossa käytetään groupby funktiota laskemaan kunkin rakennuksen maksimi Z-koordinaatille. Dataframe Pts_BlockXY sisältää tiettyyn kyselyyn (K1-K5) kuuluvat pisteet, jotka ensin ryhmitellään groupby-funktiota hyödyntäen kunkin rakennustunnuksen mukaan ('s'). Kustakin ryhmästä, joka siis vastaa yhtä rakennusta, lasketaan lopuksi Z-koordinaatille maksimiarvo.*

*Taulukko 3: Suoritetut kyselyt laajennetulla OTA-Laaj pistepilvellä. Selvästi nopeampi groupby-funktion käyttö perustuu yhden tiedoston ratkaisuun ja siinä tulokset vaihtelevat välillä 0,125s-0,31s. Kysely ilman groupby-funktiota puolestaan lukee useita erillisiä tiedostoja (yksi per rakennus) ja näin ollen toimii varsin hitaasti rakennusten lukumäärän kasvaessa (yli 4s. kyselyn K5 kohdalla, joka käsittää kaikki 905 rakennusta).*

| Laajennettu OTA-Laaj pistepilvi. Otaniemi sekä lähialue, (8 971 327 pistettä, 1 940 rakennusta) | Kysely K1 kesto (ms) | Kysely K2 kesto (ms) | Kysely K3 kesto (ms) | Kysely K4 kesto (ms) | Kysely K5 kesto (ms) |
|---|---|---|---|---|---|
| Kyselyn palauttama pistemäärä | 61 | 165 | 433 | 663 | 905 |
| Kyselyn kesto ilman groupby-funktiota (ms) | 590 | 1 010 | 2 120 | 3 220 | 4 150 |
| Kyselyn kesto käytettäessä groupby-funktiota (ms) | 125 | 156 | 219 | 297 | 310 |

Jälkimmäisessä menetelmässä taas ei käytetty lainkaan groupby-funktiota. Siinä luettiin ensin jokaisesta alihakemistosta (ohittaen kuitenkin ne kaksi alihakemistoa, jotka liittyivät rakennuksiin, joita ei haluttu kyselyyn) kyseiseen rakennukseen liittyvä binääritiedosto P'1. Samalla lisättiin luetun tiedoston pisteet pandas-dataframe tietorakenteeseen ja laskettiin jokaiselle rakennukselle sen Z-koordinaatin maksimiarvo. Näin jatkettiin, kunnes kaikki rakennukset oli käyty läpi ja saatiin koottua dataframe tietorakenne, joka sisälsi kaikki tarvittavat pisteet.

Tulokset kyseisten kahden eri menetelmän välillä poikkesivat merkittävästi toisistaan kuten Taulukko 3 ilmenee. Kuvasta 3 on selvemmin nähtävissä, miten rakennusten (ja täten alihakemistojen) lukumäärän kasvaessa ilman groupby-funktiota kyselyn kestoaika (katkoviivalla esitetty) lähtee jyrkkään kasvuun. Kysely K1 ilman groupby-funktiota kestää nimittäin n. yhden sekunnin kun läpi käytäviä alihakemistoja on 165 kpl, mutta koko rakennusrykelmähän käsittää 905 hakemistoa ja niiden kaikkien käsittely (kysely K5) vaatii yli 4s (4 150ms). Kun puolestaan käytetään groupby-funktiolla toimivaa kyselyä, kyselyn K5 kestoaika jää selvästi alle 0,5s vaikkakin kysely sisältää siis kaikki 905 rakennusta. Kuten Kuvasta 3 näkyykin, groupby-toteutusta kuvaava käyrä kasvaa hyvin maltillisesti kun hakemistojen lukumäärää kasvatetaan. Groupby-menetelmän nopeus perustuu pitkälti siihen, että rakennukset on koottu yhteen binääritiedostoon, jonka siirtäminen dataframe muistirakenteeseen on hyvin nopeaa.




*Kuva 3: Tulokset Python-ohjelmalla suoritetusta kyselystä ilman ja 'groupby' rakenteen kanssa. Vaakatasossa on niiden rakennusten lukumäärä, jotka palautettiin kyselyssä. Kyselyjen kesto on millisekunteina.*

## 4. YHTEENVETO

Tässä tarkastelussa esitimme yksinkertaisen menetelmän, jolla voidaan pilkkoa ja tallentaa pistepilvi semanttisiin osiin, jotta erilaisten kyselyjen teko pistepilvestä olisi nopeaa ja vaivatonta. Yksinkertaisuuden vuoksi rajoitimme tarkastelun rakennuksiin. Vaikkakin tässä työssä oletetaan, että annettu pistepilvi on jaettu kahteen tiedostoon P1 ja P2, joista jälkimmäinen sisältää semanttisen luokituksen, menetelmä itse asiassa siirtää tiedostossa P2 olevan semanttisen informaation hakemistoihin, jotka on nimetty semanttisen luokituksen mukaisesti. Näin ollen kyselyjä suoritettaessa tarvitsee lukea ainoastaan yhtä tiedostotyyppiä, nimittäin paikkatietoja sisältävää tiedostoa P1, tai tarkemmin sanottuna sen binäärimuotoista tiedostoa P'1. Lisäksi esitetty menetelmä poistaa tarpeen rakentaa erillinen indeksointijärjestelmä. Indeksointi, eli tarvittavien binääritiedostojen P'1 nopea paikallistaminen on nimittäin tässä siirretty käyttöjärjestelmän harteille, jonka tehtäviin kuuluu mm. tiedostojen- ja hakemistojen hallinta.

Paitsi, että tässä esitetty menetelmä on varsin yksinkertainen toteutustavaltaan, niin se mahdollistaa myös varsin nopeat kyselytulokset, erityisesti, jos kyselyssä on määritelty mitkä rakennukset tai minkä tyyppiset rakennukset (esim. hotellit, kaupungintalot, museot, sairaalat) tulee huomioida. Vaikkakin käytetty feather-tiedosto on rakenteeltaan kevyt ja nopea, tulokset osoittivat, että etsittäessä annettuun kyselyyn sopivia pisteitä useiden satojen binääristen P'1 tiedostojen joukosta on turhan hidasta. Tämä havaittiin suoritettaessa monimutkaista kyselyä laajemmasta pistepilvestä OTA-Laaj, jossa ei käytetty groupby-

funktiota vaan kyselyn tulos koottiin monesta eri tiedostosta olevasta pistepilvipalasesta. Tästä syystä menetelmä käyttää hierarkkista rakennetta, jossa tiettyyn hakemistoon tallentuu aina yksi suurempi binäärimuotoinen tiedosto P'1, joka käsittää kaikkien niiden binääritiedostojen P'1 pisteet, jotka sijaitsevat kyseisen hakemiston alla.

Tässä artikkelissa on vain raapaistu pintaa siihen minkälaisiin pistepilvisovelluksiin menetelmää voisi hyödyntää. Menetelmän pitäisi soveltua yhtä lailla sisätilan kuin ulkotilankin mallinnukseen ja sen yksinkertainen hierarkkinen rakenne mahdollistaa erityisesti kaupunkialueen mallinnuksen. Esimerkiksi menetelmällä voisi helposti rakentaa monitasoisen hierarkian tyyliin: (5) rakennukset-> (15) tietty rakennus -> (15A) rakennuksen katto-osa sekä (15B) rakennuksen julkisivut. Yleisesti ottaen tällaisessa hierarkiassa tarvittavien hakemistojen lukumäärä H saadaan selville Kaavalla 1, jossa $N_B$ kuvaa rakennusten lukumäärä ja parametri C puolestaan kertoo sen, moneenko aliluokkaan kukin rakennus jaetaan. Nyt esitetyssä monitasoisessa esimerkissä on kaksi aliluokkaa (15A sekä 15B), ja jos rakennuksia olisi 10 000 kpl, tarvittavien hakemistojen lukumääräksi muodostuisi 30 001. Tällaisen määrän tallentaminen ei ole mikään ongelma nykyisissä käyttöjärjestelmissä, jotka eivät aseta rajoituksia hakemistojen määrälle. Kyselyt, jotka koskettavat suurta määrää rakennuksia on aina mahdollista suorittaa lukemalla yksi suurempi tiedosto (se binääritiedosto P'1, joka sijaitsee välittömästi hakemiston SEM_5 alla). Sovelluksen tarkoituksesta ja tietokoneen muistikapasiteetista riippuen saattaa olla edullisempaa lukea sovelluksen käynnistämisen yhteydessä ainakin osa rakennuksista muistiin, säilyttäen näin niiden dataframe rakenteen koko sovelluksen ajan.

$$H = N_B + C^* N_B + 1 \quad (1)$$

Edellä kuvattuun hierarkiaan voisi toki vielä lisätä rakennuksen käyttötarkoitusta kuvaava luokitus. Tarkoituksenamme on tulevaisuudessa etsiä tälle menetelmälle haastavampia sovelluskohteita erityisesti kaupunkimallinnuksen piiristä. Lisätietoa tässä kuvatusta menetelmästä on saatavissa julkaisussa El-Mahgary, Virtanen & Hyyppä (2020).

## 5. LÄHTEET

El-Mahgary, S., Soisalon-Soininen, E., Orponen, P., Rönnholm, P. & Hyyppä, H. (2019). OVI-3: An incremental, NoSQL visual query system with directory-based indexing. Working paper.

El-Mahgary, S., Virtanen, J.-P. & Hyyppä, H. (2020). A Simple Semantic-Based Data Storage Layout for Querying Point Clouds. *ISPRS International Journal of Geo-Information*, 9(2).

Hackel, T., Savinov, N., Ladicky, L., Wegner, J. D., Schindler, K., & Pollefeys, M. (2017). Semantic3D.net: A new Large-scale Point Cloud Classification Benchmark. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* II-3/W4, 91–98.

Joosten, F. (2018). *Map supported point cloud registration* (pro gradu-tutkielma). University of Twente.

Maanmittauslaitos. (2019). Maanmittauslaitos, Kartat ja paikkatieto. Haettu 3. 12. 2019 osoitteesta https://www.maanmittauslaitos.fi/en/maps-and-spatial-data/expert-users/product-descriptions/laser-scanning-data

McKinney, W. (2018). *Python for Data Analysis* (2. painos). Sebastopol, California, USA: O'Reilly.

McKinney, W. (5. February 2020). Powerful Python Data Analysis Toolkit. Haettu 3. 3 2020 osoitteesta Powerful Python Data Analysis Toolkit: https://pandas.pydata.org/pandas-docs/stable/pandas.pdf

MML. (2020). Maanmittauslaitos: Ympäristön 3D-mallinnus. Haettu 9. 3 2020 osoitteesta https://www.maanmittauslaitos.fi/tutkimus/teematietoa/ympariston-3d-mallinnus

**Kiitokset**: Ilmaisemme kiitollisuutemme Suomen Akatemialle saaduista rahoituksista: projekti no 272195 (Center of Excellence in Laser Scanning Research CoE-LaSR) sekä projekti no 293389.