TAMPEREEN
AMMATTIKORKEAKOULU

TERÄSRAKENTEISTEN SILTOJEN PARAMETRINEN MALLINNUS

Mikko Toola

Opinnäytetyö
Elokuu 2017
Rakennustekniikka
Talonrakennustekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Rakennustekniikan koulutusohjelma
Talonrakennustekniikka

TOOLA, MIKKO:
Teräsrakenteisten siltojen parametrinen mallinnus

Opinnäytetyö 34 sivua, joista liitteitä 0 sivua
Elokuu 2017

Opinnäytetyön toimeksiantajana toimi WSP Finland Oy, joka on erikoistunut muun muassa vaativiin siltaprojekteihin Suomessa ja sen ulkopuolella. Työn ohjaajana oli toimeksiantajan siltasuunnittelija, diplomi-insinööri, Olli Perälä. Tavoitteena oli tutkia parametrisen mallintamisen mahdollisuuksia, selvittää siitä aiheutuvia ongelmia ja kehittää visuaalista skriptausta osana mallinnusta.

Tässä insinöörityössä toteutettiin Kruunuvuorensillan pylonin teräskotelo ja teräsosien parametrinen malli Tekla Structures -ohjelmaan, mikä on osa Kruunusillat-hanketta. Työ oli käytännönläheinen ja se sisältää tietoa ohjelmista sekä algoritmiavusteisesta mallintamisesta. Työssä esiteltiin myös lyhyesti ja yksinkertaisesti Rhinoceros 3D -ohjelman lisäosa Grasshopper, jolla parametrinen malli toteutettiin. Työtä tehdessä opittiin paljon uutta mallintamisesta ja etenkin Grasshopper -lisäosasta.

Grasshopper -lisäosan ja Tekla Structures -ohjelman yhteiskäytöstä selvitettiin paljon huomioitavia asioita. Parametrinen malli toteutettiin onnistuneesti ja todettiin visuaalisen skriptauksen avulla tuotettu malli tarpeelliseksi kyseisessä kohteessa, koska lopullista mitoitusta ei ollut tehty. Siltojen haastavan geometrian ja lähtötietojen muuttumisen vuoksi mallintamista voisi toteuttaa tulevaisuudessa täysin visuaalisella skriptauksella.

Asiasanat: algoritmi, algoritmiavusteinen suunnittelu, parametri, parametrinen mallintaminen, Grasshopper

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Construction Engineering
Building Construction

TOOLA, MIKKO:
Parametric Modelling of Steel Structure Bridges

Bachelor's thesis 34 pages, appendices 0 pages
August 2017

The thesis was commissioned by WSP Finland Ltd., which is specialized among other things in demanding bridge projects in Finland and beyond. The work was supervised by Olli Perälä, M.Sc. Bridge Engineer. The objective was to explore the possibilities of parametric modelling, to identify possible problems and to develop visual scripting as part of modelling.

In the framework of this thesis a parametric model of Kruunuvuorensilta's pylon steel cage and steel parts was created to Tekla Structures, as part of Kruunusillat project. The thesis was practice-oriented and it includes basic information about programs and algorithm-aided modelling. Rhinoceros 3D program's add-on Grasshopper, used for creating the parametric model, is described briefly and in a simplified way. The thesis offered the opportunity to learn more about modelling and especially about using the Grasshopper plugin.

Several issues arose, that are to be considered while using Grasshopper add-on in relation with Tekla Structures program. The parametric model was implemented successfully and the model produced by visual scripting was found necessary for this case, because a final design had not been made. Due to bridges' challenging geometry and possible changes of underlying data, the modelling of bridges could be done entirely using visual scripting in the future.

Key words: algorithm, algorithm-aided design, parameter, parametric modelling, Grasshopper

SISÄLLYS

LYHENTEET JA TERMIT

Tekla	Tekla Structures -tietomallinnus ohjelmisto
Rhino	Rhinoceros 3D -mallinnus ohjelmisto
Grasshopper	Visuaalisen skriptauksen lisäosa Rhinoceros 3D:hen
Ohjelmakomponentti	Komponentti, joka suorittaa laskuja Grasshopper -lisäosassa
Tekla komponentti	Komponentti, joka mallintaa objekteja Teklaan Grasshopperista
Algoritmi	Suorittaa tietyn tehtäväsarjan
Parametri	Algoritmissa yksittäinen muuttuja
Visuaalinen skriptaus	Visuaalisten ohjelmakomponenttien avulla luotu algoritmi
Lanka	Komponenttien yhdistämistä tehtäväsarjaksi
Parametrinen malli	Riippuvuussuhteiden rakentamista algoritmia ohjaavien parametrien välille suunnittelumallissa
Input	Ohjelmakomponenttiin tai algoritmiin syötettävä data
Output	Ohjelmakomponentista saatava data

1 JOHDANTO

Tietomallintaminen on kehittynyt huomattavasti rakentamisessa ja rakennesuunnittelussa 2000-luvun alusta alkaen. Suunnittelu ja mallintaminen tehdään lähes kokonaan tietokoneavusteisesti. Nykyään siltojen suunnittelussa käytetään yhä enemmän tietomalleja suunnittelun apuna. Suurissa siltakohteissa on monen eri alan asiantuntijoita mukana ja suunnittelun eri vaiheissa voidaan tehdä vielä suuria muutoksia, jotka vaikuttavat mallintamiseen. Muutoksiin voidaan varautua tekemällä parametrisia malleja. Tähän tarkoitukseen soveltuu visuaaliseen skriptaukseen perustuva lisäosa Grasshopper.

Grasshopper on Rhinoceros 3D -ohjelman lisäosa, jolla voidaan mallintaa Tekla Structures -ohjelmaan objekteja. Lisäosan avulla objekteja on myös mahdollista mallintaa parametrisesti. Muutoksiin pystytään varautumaan etukäteen jo suunnittelun alkuvaiheessa hyödyntämällä parametrista mallinnusta. Tarkoituksena on havainnollistaa Grasshopperin käyttöä, soveltuvuutta ja mahdollisuuksia rakennesuunnittelijan mallintamisen työkaluna.

Kokeneiden siltasuunnittelijoiden kanssa käymieni keskustelujen mukaan siltojen mallintamisen kehityksen suunta on yhä enemmän parametrisessa mallinnuksessa ja sillä on potentiaalia jäädä osaksi mallintamista. Opinnäytetyössä tehdään sillan pylonin teräsrakenteiden parametrinen malli ja esitellään algoritmiavusteista mallintamista. Työssä keskitytään mallintamiseen visuaalista skriptausta hyväksi käyttäen Grasshopperin avulla. Työ toteutettiin sekä kirjallisuustutkimuksena, että tapaustutkimuksena. Lopussa selvitetään esimerkkitapauksessa havaittuja ongelmia ja mahdollisia hyötyjä.

2 3D-mallintaminen

2.1 Tekla Structures

Tekla Structures -ohjelmisto on Trimble Solutions Oy:n omistama rakennesuunnitteluohjelmisto, jolla voidaan tehdä tarkkoja, luotettavia ja yksityiskohtaisia tietomalleja. Tarkkoja tietomalleja tarvitaan onnistuneeseen rakentamisen toteutukseen ja ylläpitoon. Tekla Structures toimii kaikkien materiaalien kanssa ja sillä voidaan suunnitella monimutkaisia rakenteita, esimerkiksi geometrialtaan haastavia siltoja. (Trimble Solutions Corporation n.d.)

Tekla Structures voidaan yhdistää tärkeimpiin tuotannon- tai resurssisuunnittelujärjestelmiin ja koneiden ohjausjärjestelmiin, joita käyttävät mm. teräsrakenteiden ja raudoitusten valmistajat. Tämä vähentää manuaalisia töitä ja virheitä, sillä tiedot voidaan siirtää automaattisesti Tekla-mallista tuotannon käyttämiin järjestelmiin. (Trimble Solutions Corporation n.d.)

Tekla Structures -tietomalli voi parantaa rakennesuunnittelun tietojen laatua ja tarkkuutta. Dokumentointiin ja suunnitteluratkaisuihin jää enemmän aikaan, koska Tekla Structures tuottaa piirustukset ja raportit suoraan mallista. Myös määräluettelot on helppo tuottaa ja vaihtoehtoisten ratkaisujen analysointi on nopeaa. Kun Tekla-malli on kunnossa myös piirustukset pitävät paikkansa. (Trimble Solutions Corporation n.d.)

Havainnollistaminen on huomattavasti helpompaa kolmiulotteisena kuin tasopiirustuksena. Siltarakenteissa on valtava määrä raudoitusta, joten kolmiulotteisena tehdyissä rakenteissa on helppo huomata, jos jokin objekti leikkaa toisen objektin kanssa. Tämän ansiosta suunnittelijan virheet saattavat välttyä jo suunnitteluvaiheessa.

2.2 Rhinoceros 3D

Rhinoceros 3D, tutummin Rhino, on Robert McNeel & Associatesin kehittämä monipuolinen ja tehokas 3D NURBS -mallinnusohjelma. Sillä voidaan mallintaa tarkasti minkä

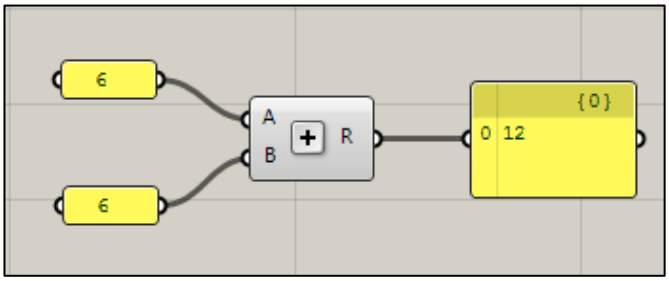
tahansa kappaleen muodon. Rhino soveltuu erinomaisesti käytettäväksi muiden suunnitteluohjelmien kanssa laajan tiedostotuen ansiosta ja sen vahvuus on monipuoliset mallinnusominaisuudet. Ohjelma on helppo oppia, vaikka ei olisikaan aiempaa 3D-mallinnuskokemusta. Rhinoceroksen toimintoja on myös mahdollistaa laajentaa erilaisilla asennettavilla lisäosilla. (Moonsoft 2017.)

Tässä opinnäytetyössä ei keskitytä Rhinon käyttöön vaan siihen asennettavaan Grasshopper -lisäosaan, jota ei voida käyttää ilman Rhinocerosta.

2.3 Grasshopper

Grasshopper on David Ruttenin luoma algoritmiseen mallintamiseen perustuva ilmainen lisäosa Rhinoceros 3D -ohjelmaan. Poiketen muista skriptaukseen perustuvista ohjelmista, kuten esimerkiksi RhinoScript, perinteistä tekstimuotoista skriptausta ei tarvitse osata Grasshopperissa, sillä se perustuu visuaaliseen skriptaukseen. Ensimmäinen vakaa versio ohjelmasta julkaistiin vuonna 2014. (Grasshopper 2017.) Visuaalinen skriptaus ei vaadi aloittelijalta kovin suurta aloituskynnystä.

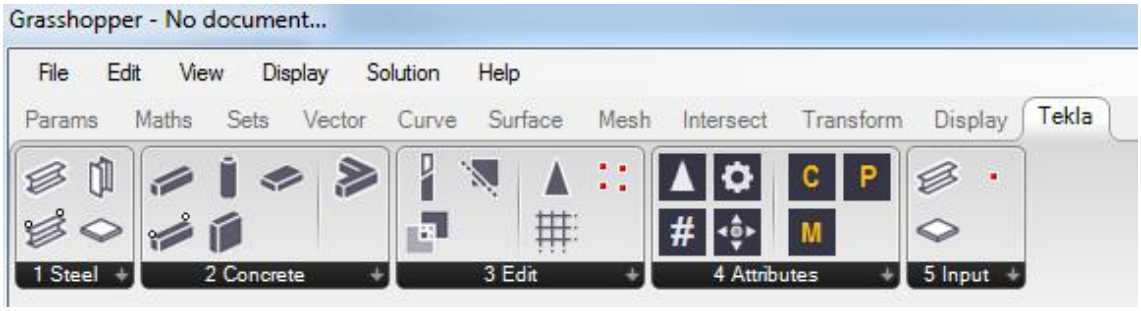
Grasshopperissa luodaan algoritmi visuaalisesti linkittämällä erilaisia ohjelmasta löytyviä komponentteja toisiinsa. Näin ollen tekstimuotoista ohjelmakoodia ei tarvitse kirjoittaa, sillä ohjelmakomponentit sisältävät itsessään kaiken koodin. Ohjelmakomponentteja on monia erilaisia ja jokaisella niistä on tietty toiminto. Komponentteja yhdistelemällä saadaan luotua erilaisia matemaattisia toimintoja ja geometriaa melko yksinkertaisesti. (Tanska & Österlund 2017, 30). Yksinkertaisimmillaan ohjelmakomponentti voi olla esimerkiksi summakomponentti (kuva 1). Komponentti laskee summalaskun, kun sille annetaan ensimmäinen luku ja yhteenlaskettava luku. Lopputuloksena saadaan summa, jonka komponentti laskee itse ilman perinteisesti käsin suoritettavaa skriptausta.



KUVA 1. Summakomponentti Grasshopperissa

2.4 Grasshopper-Tekla live link

Grasshopper-Tekla live link mahdollistaa algoritmisen mallintamisen suoraan Grasshopperista reaaliajassa Tekla Structuresiin. Teklaa voidaan ohjata Grasshopperista käsin. Tämä automaattinen linkki ohjelmistojen välille on Trimble Solutions Oy:n kehittämä ja on vielä beta-vaiheessa tätä kirjoittaessa. Linkki julkaistiin vuonna 2016 ja toimiakseen se tarvitsee Rhinon, Grasshopperin ja Tekla Structuresin asennettuna samalle koneelle. (Grasshopper-Tekla Live Link 2017.)

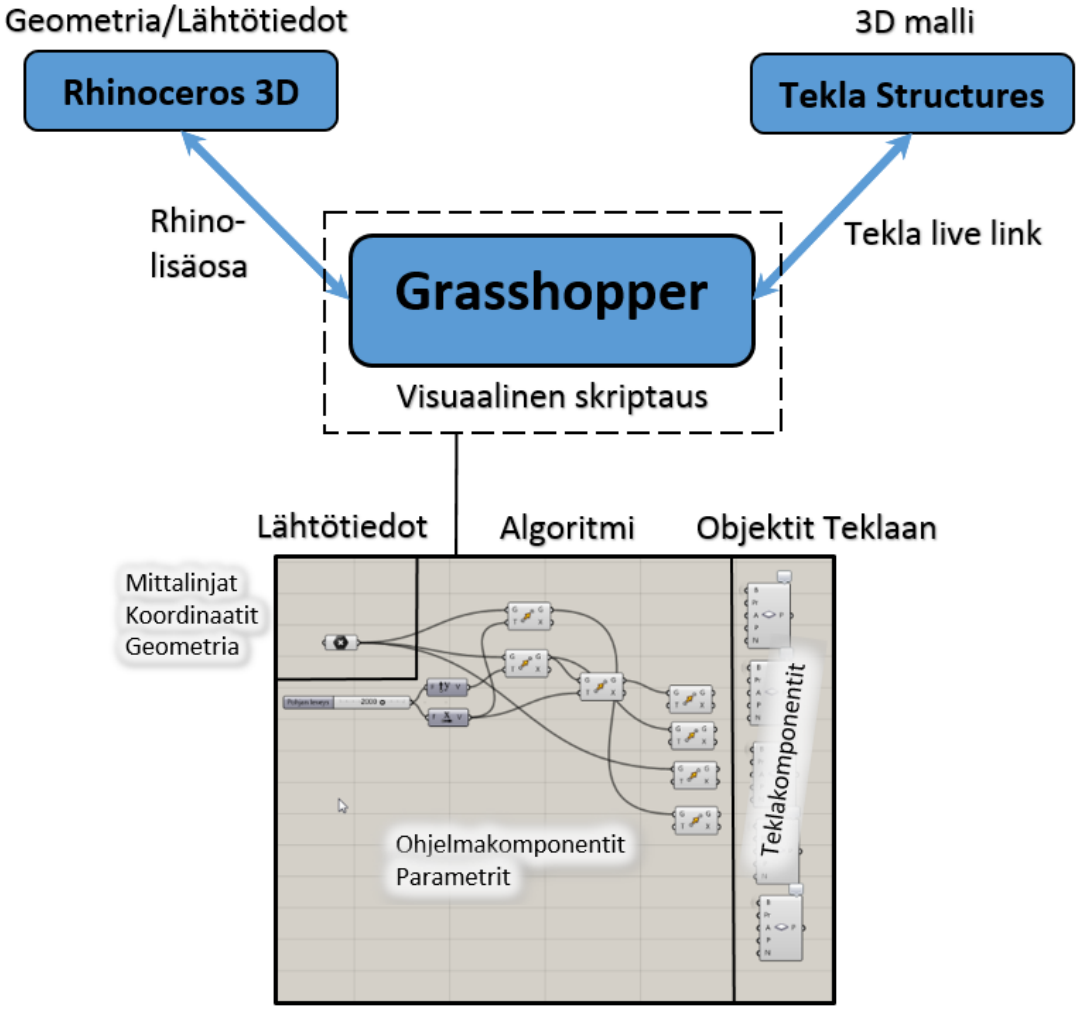


KUVA 2. Grasshopper-Tekla live link -lisäosa asennettuna

Kun Grasshopper-Tekla live link on asennettu, ilmestyy Grasshopperin komponentteihin Tekla-niminen välilehti (kuva 2). Välilehdeltä löytyvät komponentit ovat kaikki suorassa yhteydessä Tekla Structuresiin ja näillä luodaan haluttuja objekteja Tekla-malliin. Tässä opinnäytetyössä käytetään tätä hyväksi, kun tehdään parametrinen malli esimerkkitapauksessa.

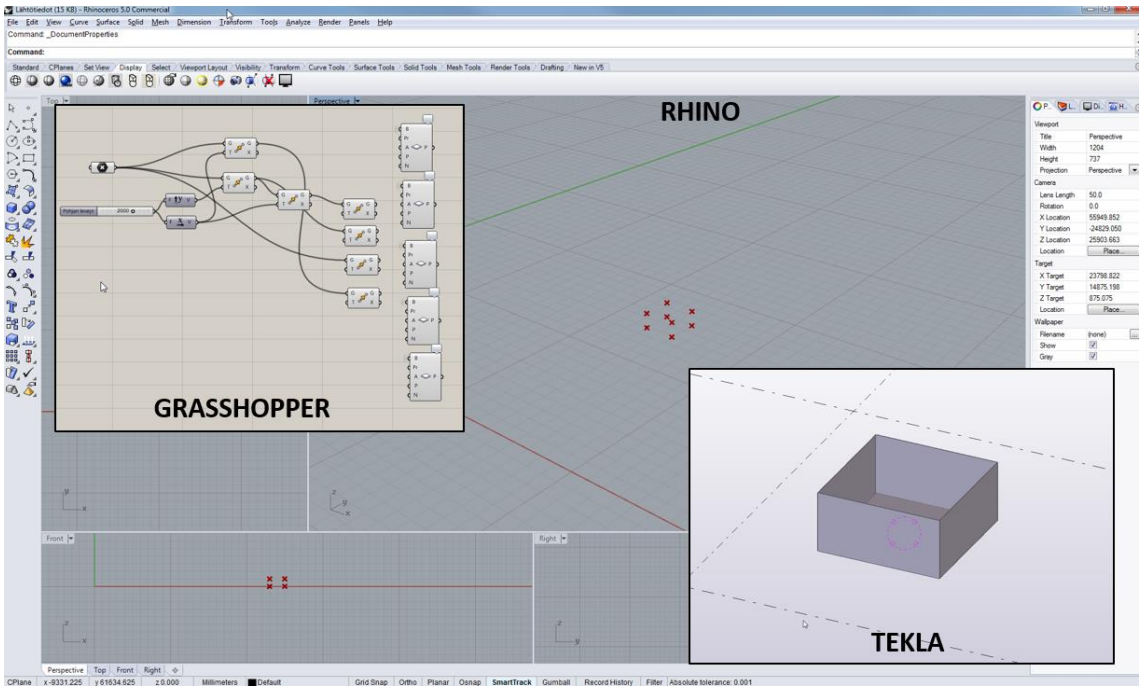
2.5 Ohjelmien yhteys toisiinsa

Edellä mainitut ohjelmat ovat kaikki yhteydessä keskenään (kaavio 1). Kun Grasshopperissa tehdään visuaalista skriptausta, tiedot välittyvät Teklaan ja Rhinoon. Teklaan ilmestyy objekti vasta sitten, kun algoritmin loppuun tai tiettyyn osaan on yhdistetty Tekla komponentti Grasshopperissa. Rhinoon päivittyy kaikki Grasshopperin visuaalisella skriptauksella tehdyt vaiheet ja siellä näkyy mm. laskettua geometriaa.



KAAVIO 1. Ohjelmien yhteys

Kuvassa 3 on havainnollistettu, miltä algoritmi näyttää Grasshopperissa, Grasshopperin luomat pisteet Rhinoon sekä näkymä Teklassa, kun Tekla komponentit on mallintanut objektin Teklaan.



KUVA 3. Ohjelmistojen näkymät

3 Algoritminen mallintaminen

3.1 Algoritmi

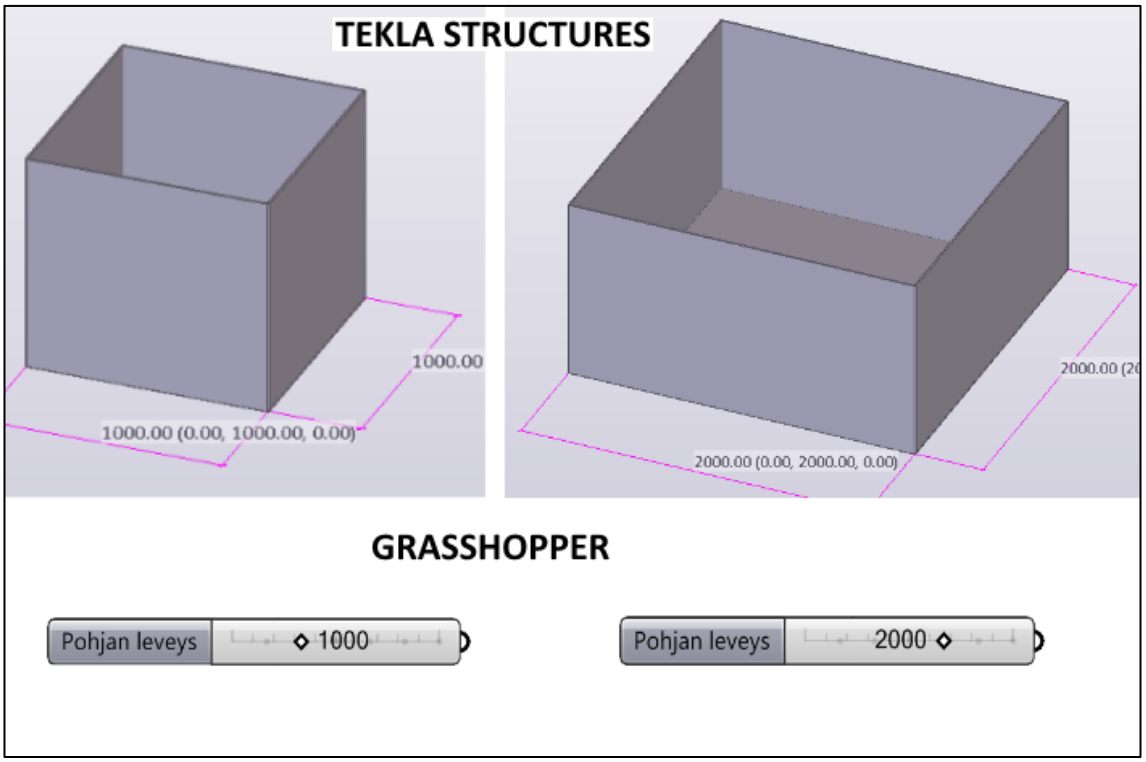
Algoritmi tarkoittaa yksinkertaisuudessaan sitä, että suoritetaan tietty komentosarja. Se on sarja, joka kuvaa määriteltyjä tehtäviä, ohjeita ja sääntöjä tietyn päämäärän saavuttamiseksi. Tehtävät määritellään tietyssä järjestyksessä halutun lopputuloksen saamiseksi. Lopputulos on aina sama jokaisen suorituskerran jälkeen, jos algoritmin lähtöarvoja ja komentosarjaa ei muuteta. (Tanska & Österlund 2017, 20.)

Algoritmi oli alun alkaen matemaattinen käsite, mutta nykyisin se liittyy yleensä tietokoneiden ohjelmointiin ja tietojenkäsittelyyn. Tietokoneella toteutetut algoritmit ovat huomattavasti nopeampi ja tarkempi toteuttaa kuin manuaalisesti algoritmiin annetut ohjeet. Algoritmi voidaan myös käsittää tosielämässä tietynlaisena tehtäväsarjana. Esimerkiksi tosielämän ruokareseptin ohjeiden noudattaminen on tietynlainen algoritmi. Kun reseptiä noudatetaan, saadaan haluttu lopputulos. Reseptin toimiminen perustuu siihen, että tuodaan oikeat ruoka-aineet, ohjeiden mukaan mitataan oikeat mittasuhteet, valmistetaan tietyssä reseptin määräämässä järjestyksessä ja saadaan haluttu lopputulos. (Tanska & Österlund 2017, 20.)

3.2 Parametrinen mallintaminen

Termi parametrinen on peräisin matematiikasta, mutta ei tiedetä tarkalleen milloin suunnittelijat aloittivat sen käytön (Davis 2013). Parametrinen mallinnus tarkoittaa sitä, että asetetaan tiettyjä riippuvuussuhteita, jotka muokkaavat algoritmia tai algoritmin tiettyä osaa halutulla tavalla. Suunnittelija voi parametrista mallinnusta hyväksi käyttäen tehdä parametrisointeja, jossa jo yhden parametrisen arvon muuttaminen muokkaa koko mallin geometriaa tai vain haluttua geometriaa. (Tanska & Österlund 2017, 13.) Parametrisen mallinnuksen avulla voidaan siis esimerkiksi tehdä teräslevyinen laatikko, jossa seinien ja pohjan mitat ovat riippuvuussuhteessa. Kun pohjan leveyttä muutetaan, muuttuu myös seinän sijainti automaattisesti suhteessa pohjan leveyteen. Tässä tapauksessa parametrina on terälevyisen laatikon leveys. Kuvassa 4 on esimerkki parametrisestä mallinnuksesta,

jossa pohjan leveys on parametri. Tekla-malli päivittyy automaattisesti, kun arvoa muutetaan Grasshopperin liukusäätimessä eli ”sliderissa”.



KUVA 4. Yksinkertainen parametrisoitu Tekla-malli

3.3 Algoritmiavusteinen suunnittelu

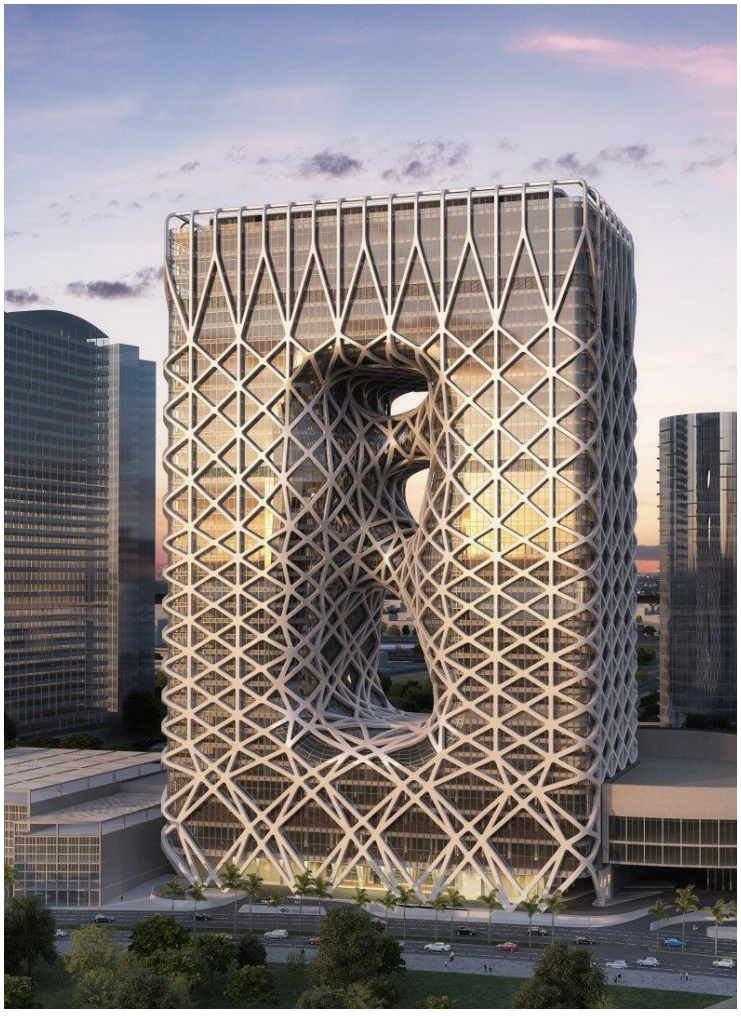
Algoritmiavusteinen suunnittelu tarkoittaa sitä, että suunnittelija käyttää hyväksi, esimerkiksi mallintamisessa, algoritmeja ja toistuvia sääntöjä rakenteiden luomiseksi. Algoritmiavusteisen suunnittelun avulla voidaan luoda visuaalista skriptausta hyväksi käyttäen algoritmi, jolla luodaan kolmiulotteinen parametrinen malli. Näillä saadaan aikaan haluttu prosessin lopputulos, joka voi olla esimerkiksi teräsrakenteisen sillan pääkannattimet ja poikkipalkit. Suunnittelija määrittelee lähtötiedot, mittaviivat ja geometriat. Algoritmin suunnittelija suunnittelee sääntö ja ohje kerrallaan algoritmin. Grasshopperilla voidaan siis luoda algoritmi visuaalisella skriptauksella, jonka lähtötietoina ovat mittaviivat ja lopputuloksena saadaan Tekla objekteja.

Algoritmiavusteinen suunnittelu on arkkitehtien tehokas työväline, koska sillä saadaan tehtyä monia erilaisia vaihtoehtoratkaisuja yhden algoritmin avulla sekä mallinnettua geometrialtaan haastavia kohteita. (Tanska & Österlund 2017, 36.) Algoritmiavusteista

suunnittelua voi hyödyntää myös rakennesuunnittelijat ja mallintajat, mutta aiheesta on tehty vähän tutkimuksia. Tässä opinnäytetyössä on hyödynnetty algoritmiavusteista suunnittelua osana mallinnusta.

3.4 The Morpheus Hotel

The Morpheus Hotel (kuva 5) on Kiinan Macaoon suunniteltu hotelli, joka on suunniteltu algoritmiavusteisesti ja se on parametrisoitu Grasshopperia ja Rhinoa käyttäen. Hotellin julkisivu on geometrialtaan hyvin monimuotoinen ja siinä on käytetty ulkoisena tukirakenteena terästä, joka on verhoiltu alumiinilla. Rakennus on 40-kerroksinen kaksitorninen hotelli, jonka keskellä on kolme vapaamuotoista aukkoa. Rakennuksen arkkitehtuurin suunnitteli Zaha Hadid Architects ja julkisivun tuotannon konsulttina toimi Front Inc. Rakennuksen ennakoidaan valmistuvan vuonna 2018. (The Morpheus Hotel: From Design to Production: Live Webinar 2017).



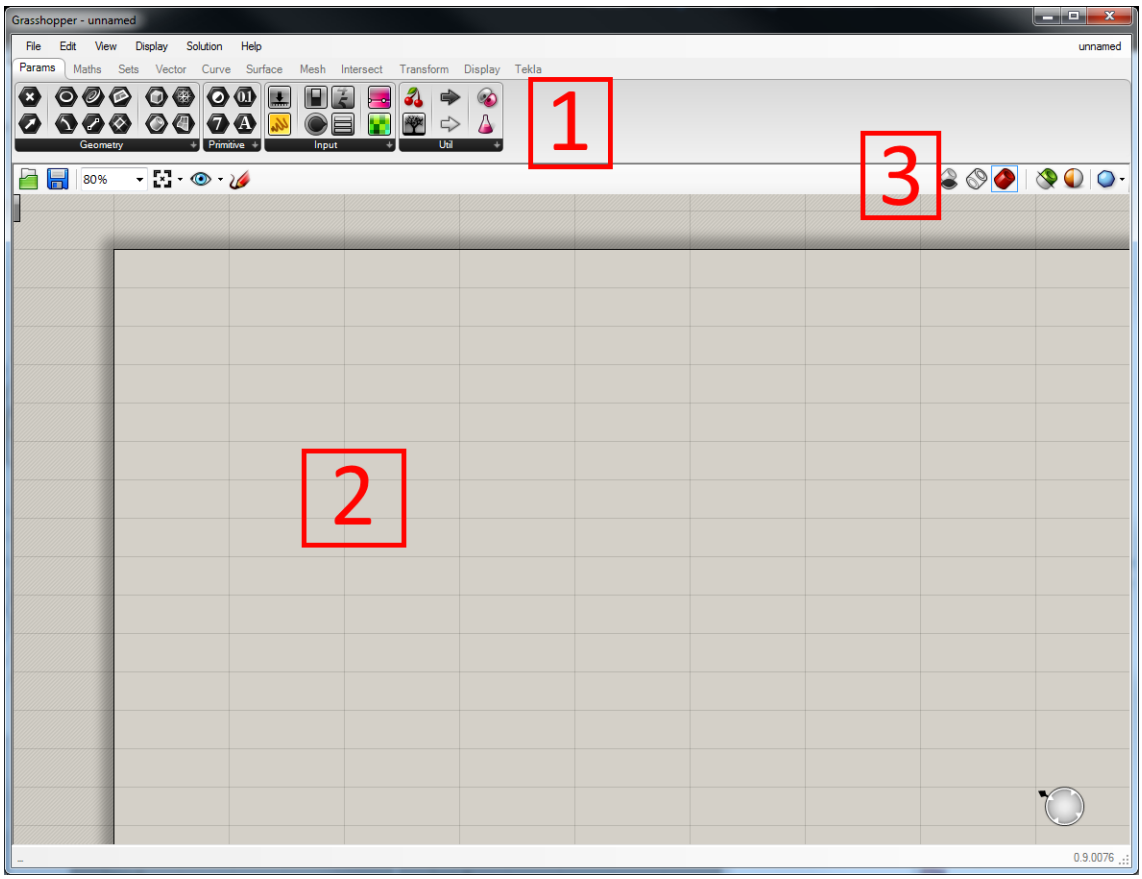
KUVA 5. Morpheus -hotelli Kiinassa (Zaha Hadid Architects)

Ulkoinen tukirakenne on geometrialtaan erittäin haastava, joten mallintamisessa päädyttiin ratkaisuun, jossa tehtiin monta eri mallia rakennuksesta. Rakennuksesta tehtiin 3522 Rhino 3D-mallia, jonka mallintamiseen käytettiin apuna 462:ta erilaista Grasshopper tiedostoa. Yhden tiedoston käyttäminen olisi ollut vaikeaa organisoida, se olisi ollut liian raskas ja sitä olisi voinut käyttää ainoastaan yksi ihminen kerrallaan. Projektissa ulkoista tukirakennetta oli suunnittelemassa neljä ihmistä samanaikaisesti. Ulkoisesta tukirakenteesta laadittiin Grasshopperin avulla automaattisesti 350 000 konepajapiirustusta ja 150 000 kokoonpanopiirustusta. Rakennuksen monimuotoisuuden vuoksi, siitä tehtiin FEM-analyysit siirtämällä data automaattisesti Rhinosta ja Grasshopperista laskentaohjelmaan. (The Morpheus Hotel: From Design to Production: Live Webinar 2017).

Hotellin suunnittelu on erinomainen näyttö siitä, mitä Grasshopperilla pystytään saamaan aikaan. Arkkitehtien asettamat reunaehdot pystyttiin määrittämään ja parametrisoimaan muuttujat Grasshopperin avulla. Mallintamisen apuna käytettiin Grasshopperin visuaalista skriptausta, mikä säästää aikaa verrattuna perinteiseen koodaamiseen. Projekti antaa mielenkiintoisen lähtökohdan parametrisen suunnittelun maailmaan.

3.5 Yleistä Grasshopperista

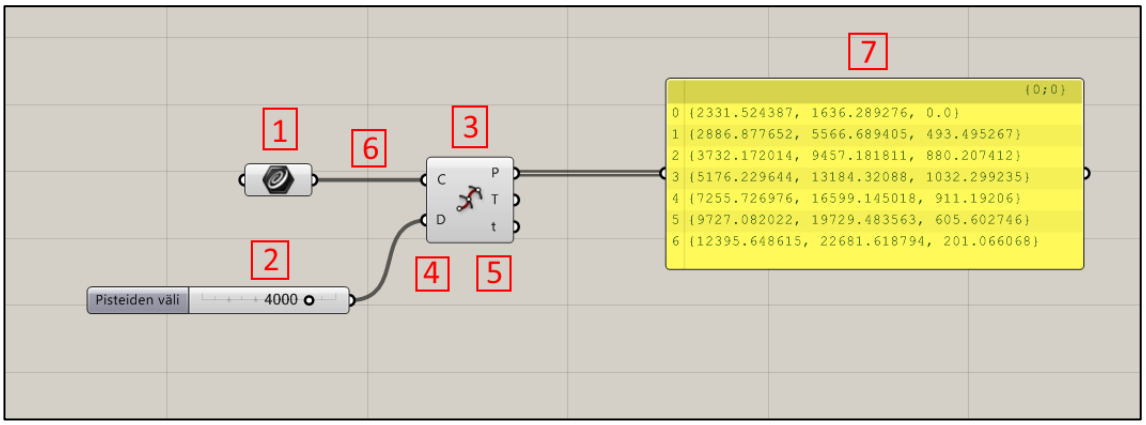
Tässä kappaleessa esitetään ja käydään läpi Grasshopperin käyttöliittymä ja peruskäsitteet. Kuvassa 6 on esitetty käyttöliittymä mikä aukeaa, kun Rhinon lisäosa Grasshopper avataan, kirjoittamalla Rhinon komentoriviin Grasshopper. Kuvassa 6 alueessa 1 on component menu eli kaikki Grashopperin ohjelmakomponentit, jotka sijaitsevat eri välilehdillä. Tekla live linkin asentamisen jälkeen välilehdeksi ilmestyy myös Tekla-välilehti.



KUVA 6. Grasshopper käyttöliittymä

Alueesta 2 löytyy canvas eli piirtoalue, johon tehdään visuaalinen skriptaus ohjelmakomponenttien avulla. Alueessa 3 on valintapainikkeet, joilla voidaan hallita näkymäasetuksia Rhinon puolella.

Kuvassa 7 on esitetty Grasshopperin perusperiaate ja keskeisimmät käsitteet. Tässä lyhyessä algoritmissa on laskettu käyrältä 4000 mm etäisyydellä toisistaan olevien pisteiden koordinaatit. Koodi etenee vasemmalta oikealle, eikä silmukoita (*eng. loop*) voida tehdä.



KUVA 7. Grasshopper algoritmi

TAULUKKO 1. Kuvan 7 selitykset

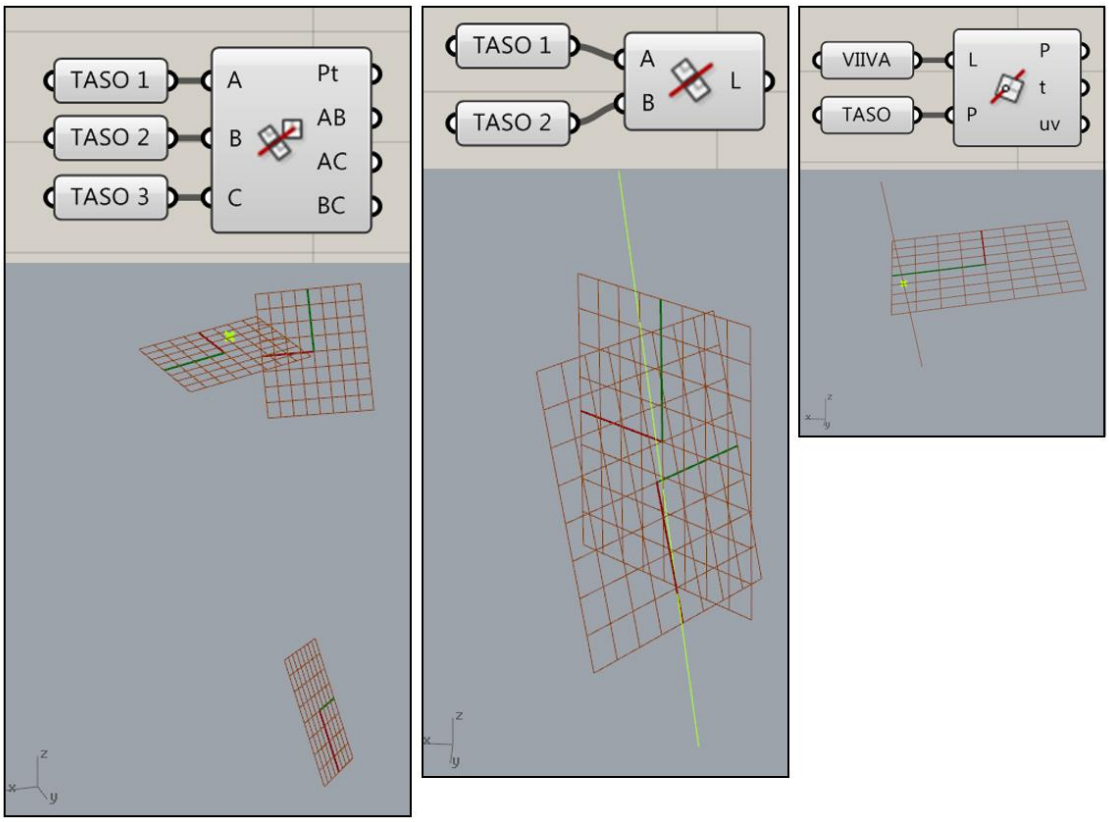
Nro.	Työkalu	Selite
1	Data input	Lähtötiedot Rhinoon piirretyltä käyrältä
2	”Slideri” (*eng. num slider*)	Parametri, komponentille välitettävä muuttuja
3	Ohjelmakomponentti	Sisältää valmiiksi määriteltyä koodia
4		Komponentin datan sisääntulo, input
5		Komponentin datan ulosmeno, output
6	Lanka (*eng. wire*)	Komponenttien yhdistämistä toisiinsa
7	Paneeli (*eng. panel*)	Voidaan lukea mm. komponenttien outputin sisältämää dataa

3.6 Matematiikka Grasshopperin apuna

Grasshopperissa luotu algoritmi käyttää ohjelmakomponentteja, jotka sisältävät itsessään tietyn matemaattisen kaavan. Ohjelmakomponenteilla tehtyyn visuaaliseen skriptaukseen tarvitaan matemaattista ymmärrystä, josta tärkeimmät ovat vektorit, yksikkövektorit ja tason koordinaatiston suunnat avaruudessa, joita voidaan hyödyntää objektien mallintamisessa. Esimerkkitapauksessa luvussa 4 visuaalisessa skriptauksessa käytettiin hyväksi mm. ohjelmakomponenttien outputista saatavia yksikkövektoreita. Yksikkövektoreja kertomalla saatiin mallinnettua objekteista halutun mittaisia ja yksikkövektoreista saatiin myös tärkeitä suuntia, jota käytettiin apuna algoritmia suunniteltaessa. Yksikkövektori on sen vektorin kanssa samansuuntainen ja sen pituus on aina yksi (Kangasaho ym. 2015, 170).

Kolmiulotteisessa maailmassa taso on äärettömän iso ja sillä on kolme eri akselia, x-, y- ja z-akselit. Grasshopperissa voidaan ohjelmakomponenttien avulla tarkkaan määritellystä tasosta määrittää x-, y- ja z-akselien yksikkövektorit. Tasoja käytettiin esimerkkitapauksessa luvussa 5, sillä siitä saatiin halutut suunnat yksikkövektoreina. Edelleen näitä kertomalla saatiin halutut pituudet. Tasoja käytettiin erittäin paljon myös siksi, että niitä käytettiin viivojen ja tasojen leikkauspisteiden määrittämisessä.

Koska tasot ovat äärettömän isoja avaruudessa tarkoittaa se sitä, että kolme erisuuntaista tasoa muodostaa ainoastaan yhden leikkauspisteen, kuvassa 8 vasemmalla. Jos kaksi erisuuntaista tasoa leikkaa toisensa, muodostaa tasojen leikkaus yhden viivan, kuvassa 8 keskellä. Jos viiva ja taso ovat erisuuntaiset, muodostaa niiden leikkaus yhden leikkauspisteen, kuvassa 8 oikealla.



KUVA 8. Grasshopper ohjelmakomponentit ja näkymä Rhinossa

4 Esimerkkitapaus: Kruunuvuorensillan pylonin teräsrakenteiden parametrinen mallinnus

Tässä kappaleessa käydään läpi teräsrakenteisen pylonin parametrinen mallinnus Grasshopperin avulla. Mallinnettuja objekteja ja parametreja havainnollistetaan selventävien kuvien avulla. Mallinnus on rakennussuunnitteluvaiheen alkuvaihetta, sillä lopullista mitoitusta ei ole vielä tässä vaiheessa tehty. Parametrisella mallinnuksella voidaan tehdä vielä muutoksia hyvin pienellä vaivalla malliin.

4.1 Yleistä Kruunuvuorensillasta

Kruunuvuorensilta (kuva 9) on osa suurta Kruunusillat-hanketta, joka yhdistää Helsingin kasvavan Laajasalon keskustaan. Laajasalon ja keskustan välille hankkeeseen sisältyy raitiotie sekä kävely- ja pyörätiet. Hankkeen yksi näkyvin osa on Kruunuvuorenrannan ja Korkeasaaren välille rakennettava Kruunuvuorensilta, joka valmistuessaan on Suomen pisin silta. Liittorakenteisen yksipylonisen vinoköysisillan kokonaispituus on noin 1,2 kilometriä, sillan pylonin korkeus on 135 metriä, pisin jänneväli on 260 metriä ja sen suunnittelukäyttöikä on 200 vuotta. Suunnittelukilpailun voitti WSP Finland Oy ja Knight Architects Ltd:n ehdotus Gemma Regalis. Helsingin kaupunginvaltuusto hyväksyi elokuussa 2016 rakentamispäätöksen Kruunusillat-hankkeesta. Rakennustöiden arvioidaan alkavan aikaisintaan vuonna 2019 ja alustava tavoite on, että vuonna 2026 yhteys avataan liikenteelle.

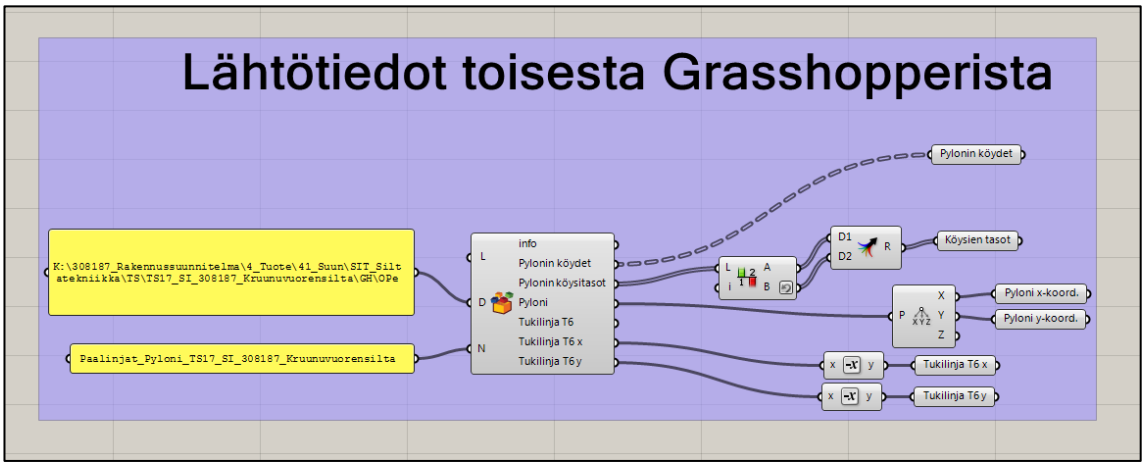
KUVA 9. Kruunuvuorensilta, katusuunnitelman havainnekuva (WSP Finland Oy)

4.2 Lähtötiedot

Koko visuaalisen skriptauksen avulla tehdyn algoritmin lähtötietoina on käytetty ainoastaan yhtä tekstitiedostoa, joka sisältää koordinaattipisteet Kruunuvuorensillan mittalinjasta. Siitä on Grasshopperilla ohjelmakomponenttien avulla skriptattu kaikki muut tarvittavat koordinaattipisteet, viivat ja tasot, joita tarvitaan objektien mallintamiseen. Koko algoritmi lähtee rakentumaan lähtötiedoksi annetusta koordinaattitiedostosta. Tämä mahdollistaa sen, että koko algoritmi muuttuu, jos mittalinjan koordinaatit vaihtuvat projektin edetessä. Lähtötiedot voidaan siis päivittää nopeasti ja se vaikuttaa koko malliin. Visuaalisella skriptauksella on jo mallinnettu sillan teräksiset pääkannattimet ja poikkipalkit. Tämä opinnäytetyö käsittelee ainoastaan sillan pylonin teräskoteloa ja sisälle tulevia ankkurointiosien teräslevyjä.

Algoritmi tehtiin Grasshopperilla kahdessa eri tiedostossa. Ensimmäisessä tiedostossa mallinnettiin teräksiset pääkannattimet, poikkipalkit ja köydet. Toisessa tiedostossa mallinnettiin pylonin teräskotelo ja köysien teräksiset ankkurointiosat. Tämä opinnäytetyö käsittelee jälkimmäistä. Jälkimmäisenä mainittuun Grasshopperilla luotuun algoritmiin saatiin lähtötiedot ensimmäisestä tiedostosta. Tietoa vaihdettiin Grasshopper tiedostojen

välillä siihen asennettavan liitännäisen avulla, joka mahdollistaa ohjelmakomponentin luomisen, jonka avulla voidaan siirtää haluttua dataa. Pylonin teräskotelon ja köysien ankkurointiosien algoritmissa lähtötietoina käytettiin pylonin köysiä, pylonin köysien muodostamaa tasoa, pylonin sijaintia ja tukilinjan T6 yksikkövektoreita (kuva 10).



KUVA 10. Data Unpack -komponentin avulla tuodut lähtötiedot

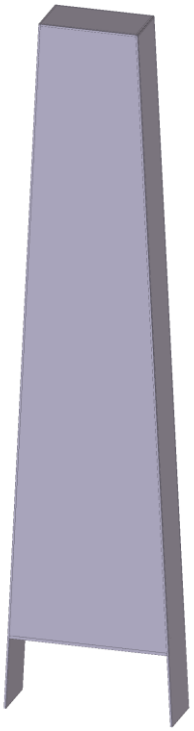
Jos ensimmäisenä luotuun toisen suunnittelijan tekemään algoritmiin tehdään muutoksia, ne päivittyvät myös tässä opinnäytetyössä käsiteltävässä algoritmissa, koska ne ovat ”linkkiketjussa” keskenään. Tämä mahdollistaa sen, että esimerkiksi pylonin köysien tulokulmaa pyloniin voidaan muuttaa ensimmäisenä luodussa algoritmissa. Muutokset päivittyvät esimerkkitapauksen algoritmiin siten, että siihen riippuvuussuhteessa olevat teräsosat päivittyvät myös uuden tulokulman mukaan. Tulokulmien muutos päivittää myös Tekla-mallin reaaliaikaisesti.

Ennen algoritmin luomista visuaalisella skriptauksella on kannattavaa käyttää hetki aikaa mallintamisen suunnitteluun. Kysymyksiä, joita kannattaa itselleen esittää on listattu seuraavaksi:

- Mitä lähtötietoja on käytettävissä?
- Mitä parametreja objekteille tarvitaan?
- Voiko toisen suunnittelijan tekemästä algoritmista saada tarvittavia tietoja?
- Miten objektit ovat riippuvuussuhteessa toisiinsa?
- Mistä objektista visuaalinen skriptaus ja mallintaminen aloitetaan?

4.3 Pylonin teräskotelo

Teräskotelon (kuva 11) mallinnuksessa käytettiin apuna pylonin keskilinjaa. Pylonin keskilinja määriteltiin Grasshopperin avulla, jossa sille annettiin x-, y- ja z-koordinaatit. Teräskotelo on riippuvuussuhteessa pylonin keskilinjan kanssa, joka tarkoittaa sitä, että jos pylonin keskilinjan koordinaatteja muutetaan, muuttuu myös teräskotelon sijainti mallissa. Teräskotelolla on myös toinen riippuvuussuhde, joka liittyy aikaisemmin visuaalisella skriptauksella määritettyyn tukilinjaan. Tukilinjan 6 taso saatiin lähtötiedoista, jonka outputista saadaan teräskotelon sivujen suunnat. Teräskotelo on määritelty kohtisuoraan tukilinjaan nähden. Näin ollen tukilinjan tasosta saatavat yksikkövektorien y- ja x-suunnat ovat tärkeä tieto, sillä siltojen mallintamisessa y-suunta vastaa sillan poikkileikkauksen suuntaa ja x-suunta pituussuunnan tangenttia.



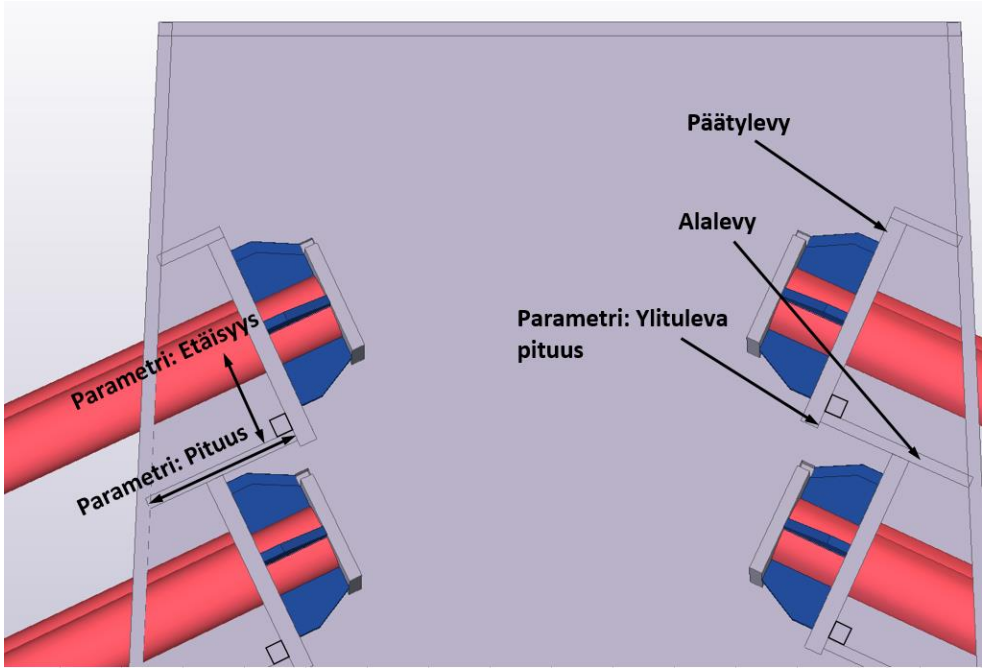
KUVA 11. Mallinnettu teräskotelo

Seuraavaksi suunniteltiin, mitä parametreja teräskotelolle annetaan. Päädyttiin ratkaisuun, jossa teräskotelo olisi lähes kokonaan muokattavissa parametrisesti. Näin ollen teräskotelon yläpinnan korko, alapinnan korko, kotelon ala- ja yläpinnan pituus sekä leveys ovat muokattavissa reaaliaikaisesti. Teräskotelon sivuille määriteltiin myös parametri-

sesti ylitulevan levyn pituus. Parametrisoimisessa täytyi myös huomioida levyjen paksuudet siten, että teräslevyt eivät leikkaa toisiaan. Visuaalinen skriptaus suoritettiin yksinkertaisesti pisteitä siirtämällä Grasshopperin avulla.

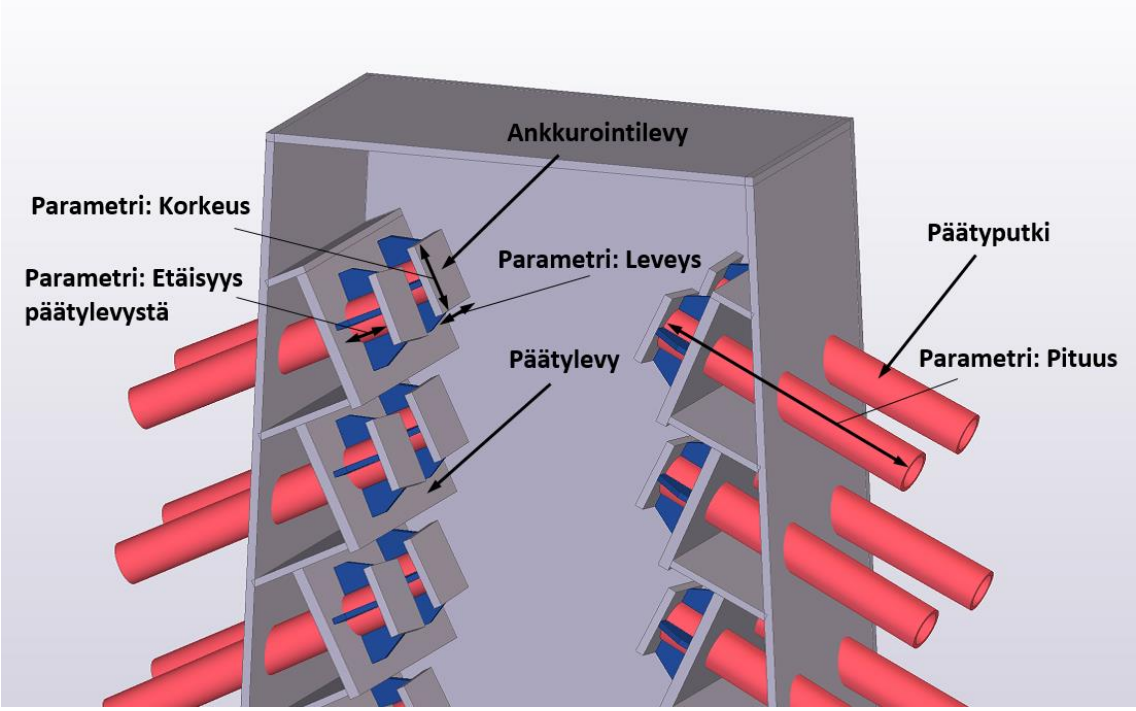
4.4 Pylonin köysien ankkurointiosat

Ankkurointiosien visuaalinen skriptaus ja mallinnus aloitettiin teräksisistä ala- ja päätylevyistä (kuva 12). Teräksiset levyt ovat riippuvuussuhteessa teräskoteloon, joten teräskotelon sijainnin tai koon muuttaminen muuttaa myös levyjen kokoja ja sijaintia mallissa. Ankkurointiköysipareja on 18 molemmin puolin pylonia. Jokainen ankkurointiköysi tulee eri kulmassa teräskoteloon ja näistä on luotu tasot Grasshopperissa. Päätylevyn ehtona on, että se on kohtisuorassa köysiparien muodostamaan tasoon nähden. Alalevy on aina 90 asteen kulmassa päätylevyyn. Visuaalinen skriptaus toteutettiin Grasshopperissa siten, että köysiparien muodostamaa tasoa siirrettiin tietyn etäisyyden päähän alkuperäisestä, jolloin saatiin alalevylle muodostettua taso. Päätylevy toteutettiin niin, että komponenttien avulla luotiin viivoja ja viivojen leikkauspisteitä tasoihin, mistä se mallinnettiin. Pääty- ja alalevy parametrisoitiin niin, että alalevyn pituutta ja etäisyyttä voidaan muuttaa reaaliaikaisesti. Päätylevyn koko muuttuu samalla kun tehdään muutoksia alalevyyn ja se on aina 90 asteen kulmassa siihen nähden. Päätylevyyn parametrisoitiin myös ylituleva pituus alalevystä.



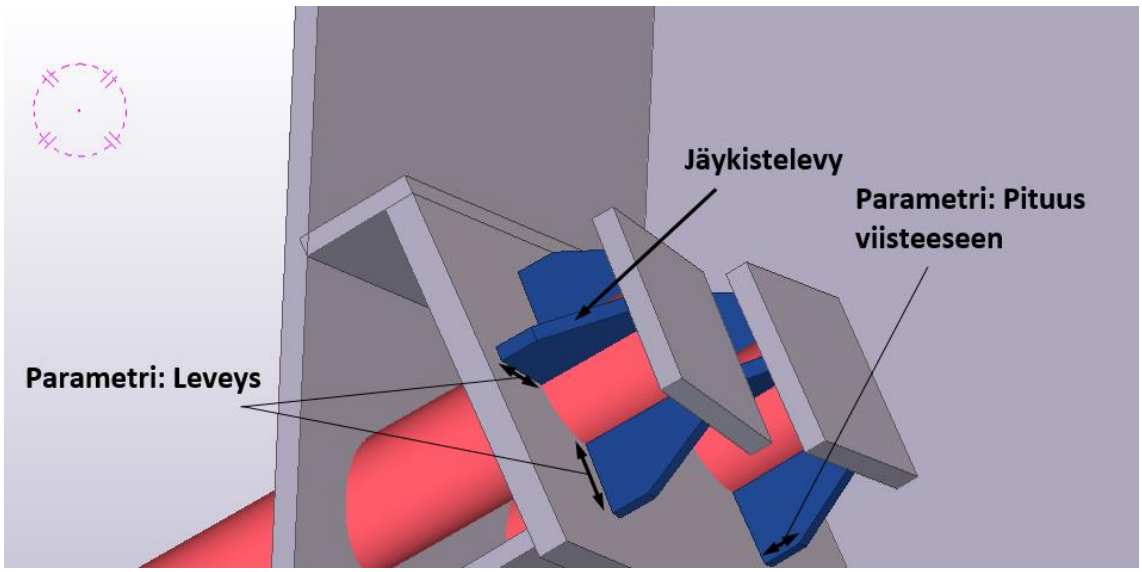
KUVA 12. Ala- ja päätylevyn parametrisoinnit

Seuraavaksi mallinnettiin ankkurointiköysien päihin tulevat teräksiset ankkurointilevyt ja päätyputket (kuva 13). Molemmat ovat riippuvuussuhteessa päätylevyihin ja ankkurointiköysiin. Muutokset edellä mainittuihin objekteihin muuttaa myös ankkurointilevyjen ja päätyputkien sijaintia. Visuaalinen skriptaus toteutettiin pääosin tasoja ja tasojen leikkauksia hyväksi käyttämällä sekä pisteitä liikuttamalla. Ankkurointilevyyn parametrisoitiin leveys, korkeus ja etäisyys päätylevystä. Päätyputkeen parametrisoitiin ainoastaan pituus, sillä putken suunta on sidottu ankkurointiköysiin.



KUVA 13. Ankkurointilevyn ja päätyputken parametrisoinnit

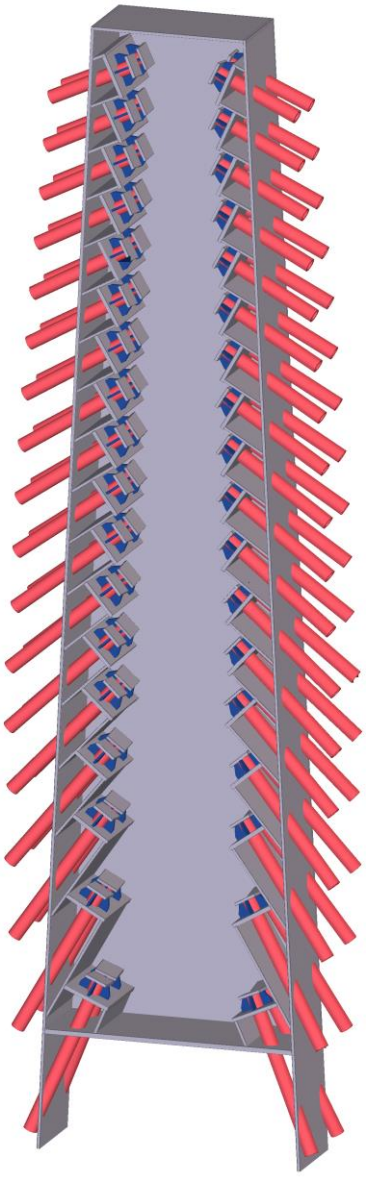
Viimeisenä mallinnettiin ankkurointilevyn ja päätylevyn väliin tulevat jäykistelevyt (kuva 14). Jäykistelevyjä on neljä kappaletta jokaisessa ankkurointilevyssä ja ne sijaitsevat niiden sivujen puolivälissä. Jäykistelevyt ovat riippuvuussuhteessa päätylevyn ja ankkurointilevyn sijaintiin. Mikäli ankkurointilevyjä tai päätylevyjä siirretään, vaikuttaa se myös jäykistelevyn kokoon. Visuaalinen skriptaus toteutettiin jo aikaisemmin luotujen objektien tasoja ja suuntia apuna käyttäen. Näistä luotiin eri komponenttien avulla uusia pisteitä, josta mallinnettiin levyt. Jäykistelevyyn parametrisoitiin leveys ja pituus viisteeseen. Parametrien muutokset vaikuttavat kaikkiin jäykistelevyihin samanaikaisesti.



KUVA 14. Jäykistelevyn parametrisoinnit

4.5 Miksi parametrisoitiin?

Kruunuvuorensillan pylonin teräskotelon teräsosat ovat geometrialtaan hyvin haastavia ja lähes jokainen osa on eri kulmassa tai asennossa, koska sillan kannen geometria on monimuotoinen ja jokainen köysi on eri kulmassa. Grasshopperilla hyvin luodulla algoritmilla voidaan kuitenkin mallintaa samanaikaisesti monta eri osaa kerralla. Perinteisesti Teklaan mallintamalla olisi jouduttu mallintamaan jokainen osa erikseen. Kuvassa 15 näkyy esimerkkitapauksessa mallinnettu kohde, joka suunniteltiin kokonaan Grasshopperin visuaalisen skriptauksen avulla.



KUVA 15. Esimerkkitapaus mallinnettu Grasshopperin avulla

Kun mallintaminen aloitettiin, projekti oli rakennussuunnitteluvaiheen alkuvaiheessa, jolloin tarkkoja suunnitelmia ei vielä ollut. Näin ollen haluttiin luoda hyvin pitkälle parametrisoitu parametrinen malli. Mallintamista ei tehty vielä täysin loppuun asti, joten detaljitason suunnittelua mm. hitsejä ja lovia ei mallinnettu. Suunnitelmiin tehtävät muutokset ovat helppo ja nopea muuttaa Tekla-mallissa muuttamalla algoritmia tai parametreja Grasshopperissa. Parametrisoinnilla haluttiin myös vertailla erilaisia suunnitteluratkaisuja, koska esimerkiksi köysien tulokulma pyloniin nähden ei ollut tiedossa. Grasshopperilla onnistuttiin luomaan algoritmi, jolla voidaan köysien sijaintia ja kulmia muuttamalla vaikuttaa reaaliaikasesti Tekla-mallin kaikkiin pylonin teräskotelon teräosiin siten, ettei mallintamista tarvitse suorittaa uudestaan.

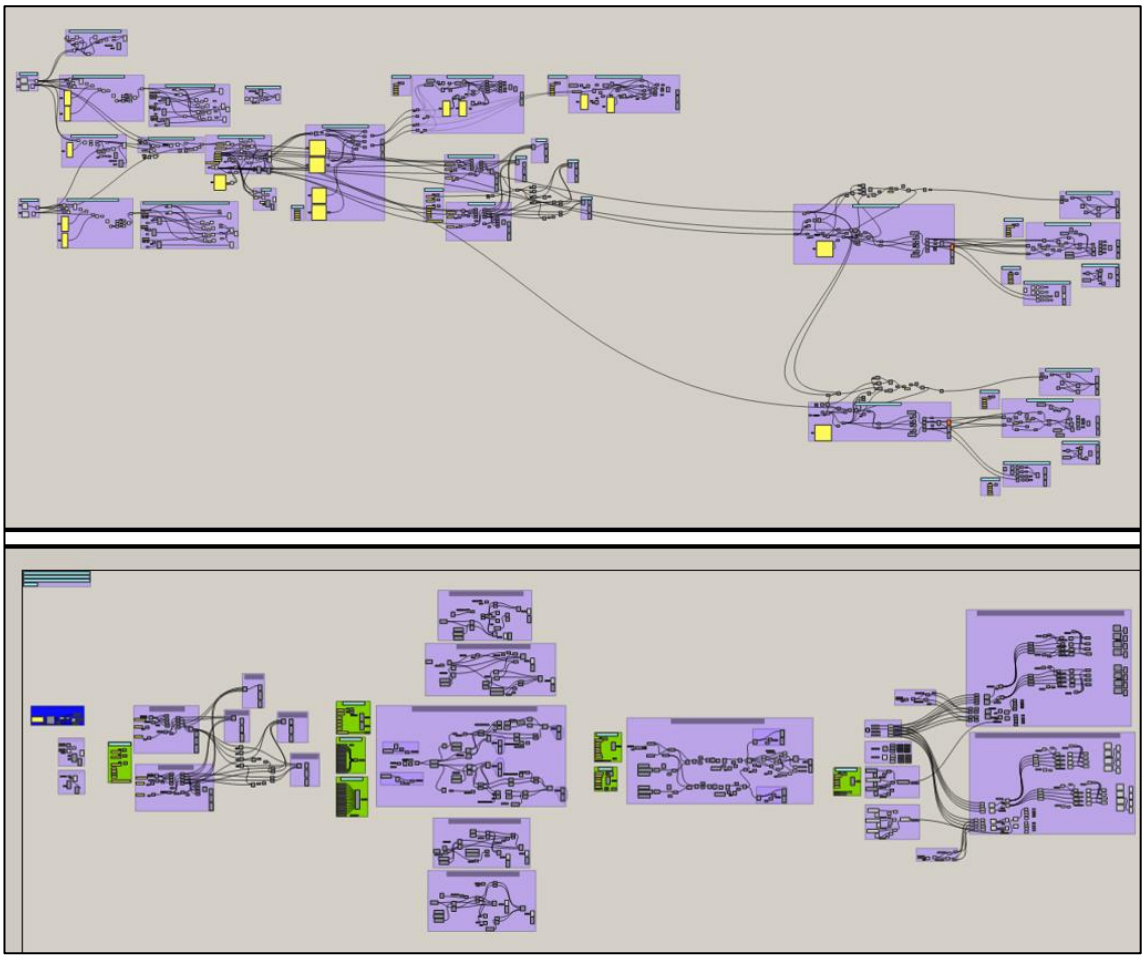
5 Esimerkkitapauksen analysointi

Tässä kappaleessa analysoidaan ja kerrataan esimerkkitapauksessa havaittuja ongelmia ja saatuja hyötyjä. Ongelmiin pyritään löytämään ratkaisuja ja kerrotaan mitä tulisi tehdä toisin algoritmiavusteisessa suunnittelussa. Osiossa käsitellään mitä hyötyä algoritmiavusteisesta suunnittelusta on siltojen mallintamisessa ja sitä onko se kannattava tapa mallintaa siltoja tulevaisuudessa.

5.1 Visuaaliseen skriptaukseen liittyvät ongelmat

Visuaalisessa skriptauksessa algoritmin luettavuus ja selkeys ovat yksi tärkeimmistä asioista, jotta siihen voidaan tehdä muutoksia vielä myöhemmin. Algoritmia on myös vaikea lukea, jos se ei ole selkeä ja toisen tekemää algoritmia voi olla vaikea tulkita. Esimerkkitapauksessa havaittiin ongelma, jossa lähes koko algoritmi jouduttiin tekemään Grasshopperilla uudestaan, koska koodista ei pystynyt lukemaan yksiselitteisesti mitä siinä tapahtuu. Tämän lisäksi algoritmin muutoskyky heikkenee, jos se on vaikeasti ymmärrettävä, eikä muutoksen aiheuttamasta vaikutuksesta ole varmuutta.

Ensimmäinen algoritmi oli visuaalisesti sekava ja siinä oli paljon ylimääräisiä komponentteja, joita ei tarvittu mallin luomiseen. Myös komponenttien välisiä lankoja oli liian paljon, jolloin algoritmista tuli vaikeasti hahmotettava. Turhien ohjelmakomponenttien käyttämistä tulisi välttää ja ne pitäisi poistaa algoritmista heti, jos niitä ei tarvita. Selkeyden kannalta myös datavirta on pidettävä vasemmalta oikealle. Jos algoritmin loppuvaiheessa tarvitaan alkuvaiheessa luodun komponentin dataa, tulisi se siirtää data komponentin avulla. Näin saadaan algoritmia luotua selkeämmäksi ja vältytään turhan pitkiltä langoilta. Kuvassa 16 on ylempänä kuvakaappaus ensimmäisenä luodusta epäselkeästä algoritmista ja alempana selkeämpi toinen versio.



KUVA 16. Epäselkeä ja selkeämpi algoritmi Grasshopperissa

Visuaalisessa skriptauksessa havaittiin myös, että algoritmia luodessa voi unohtua missä on tarvittava data, jos skriptauksesta on kulunut aikaa. Mikäli algoritmi on jonkun toisen suunnittelijan luoma niin se aiheuttaa vielä enemmän ongelmia. Visuaalisessa skriptauksessa päädyttiin ratkaisuun, jossa Grasshopperin algoritmiin sijoitettiin havainnollistavaa tekstiä, joten algoritmia on helpompi lukea jälkeenpäin pitkänkin ajan jälkeen.

Esimerkkitapauksen yhteydessä huomattiin, että Grasshopper ei tue ns. multi-user tilaa. Se ei myöskään ilmoita Grasshopper-tiedostoa avattaessa, jos se on toisella käyttäjällä auki samanaikaisesti. Tämä aiheuttaa sen, että samassa Grasshopper-tiedostossa on lähes mahdoton työskennellä samanaikaisesti, jos algoritmiin tehdään muutoksia.

Kuten esimerkkitapauksessa mainittiin, Grasshopper-tiedostot voidaan yhdistää keskenään siten, että dataa voidaan vaihtaa algoritmien välillä. Kun Grasshopper-tiedostot linkitetään keskenään, täytyy muistaa, että jos muutoksia tehdään, niin kaikki algoritmit täytyy laskea uudestaan ennen kuin ne päivittyvät Tekla-malliin. Tämä aiheutuu siitä, että

algoritmit ovat riippuvuussuhteissa toisistaan. Esimerkkinä tilanteesta on köysien tulokulmat esimerkkitapauksessa. Jos niitä muutetaan toisen suunnittelijan tekemässä algoritmissa, täytyy myös esimerkkitapauksen algoritmi laskea uudestaan ennen kuin Tekla-malli päivittyy täysin uusien geometrioiden mukaan. Mitä enemmän algoritmeja on yhdistelty toisiinsa, sitä suurempi riski on sille, että Tekla-malli ei ole päivittyneenä.

5.2 Ohjelmien yhteensopivuudesta aiheutuvat ongelmat

Kuten aiemmin kerrottiin, Grasshopper-Tekla live link -lisäosa on vielä beta-vaiheessa tätä kirjoittaessa eikä se ole täysin vakaa. Lisäosa mahdollistaa linkin Grasshopperista Teklaan ja jos tämä yhteys katoaa, muutokset algoritmiin eivät enään muokkaa Tekla-mallia. Jotta linkki toimii oikein, ohjelmat täytyy avata tietyssä järjestyksessä. Ensimmäisenä Tekla Structures, sen jälkeen Rhino ja sitten sen lisäosa Grasshopper. Yhteyden katoamisen jälkeen Grasshopperilla luodut objektit tulee poistaa Teklasta ja suorittaa Grasshopperin algoritmi uudelleen, jolloin Tekla-malliin rakentuu algoritmin luomat objektit uudestaan. Tällöin ohjelmien välinen linkki saadaan jälleen toimimaan. Grasshopperilla mallinnetuille objekteille on kannattavaa käyttää samaa ”phasea”, jolloin voidaan helposti poistaa Tekla-mallista ainoastaan Grasshopperilla mallinnetut objektit.

Grasshopperilla mallinnettaessa havaittiin myös, että kun visuaalisella skriptauksella tehdään Tekla komponentin avulla objekti Teklaan, ei objekti poistu Teklasta, jos poistaa Grasshopperista Tekla komponentin. Tällöin Tekla-malliin voi jäädä huolimattomasti objektejä, jota ei sinne haluta. Ratkaisu siihen on oikea työjärjestys. Ensin valitaan Grasshopperissa halutut Tekla komponentit aktiiviksiksi, jolloin objektit aktivoituvat automaattisesti myös Teklassa, jos ohjelmien välinen yhteys toimii. Sen jälkeen objektit poistetaan ensin Teklasta ja vasta sen jälkeen Tekla komponentit Grasshopperista.

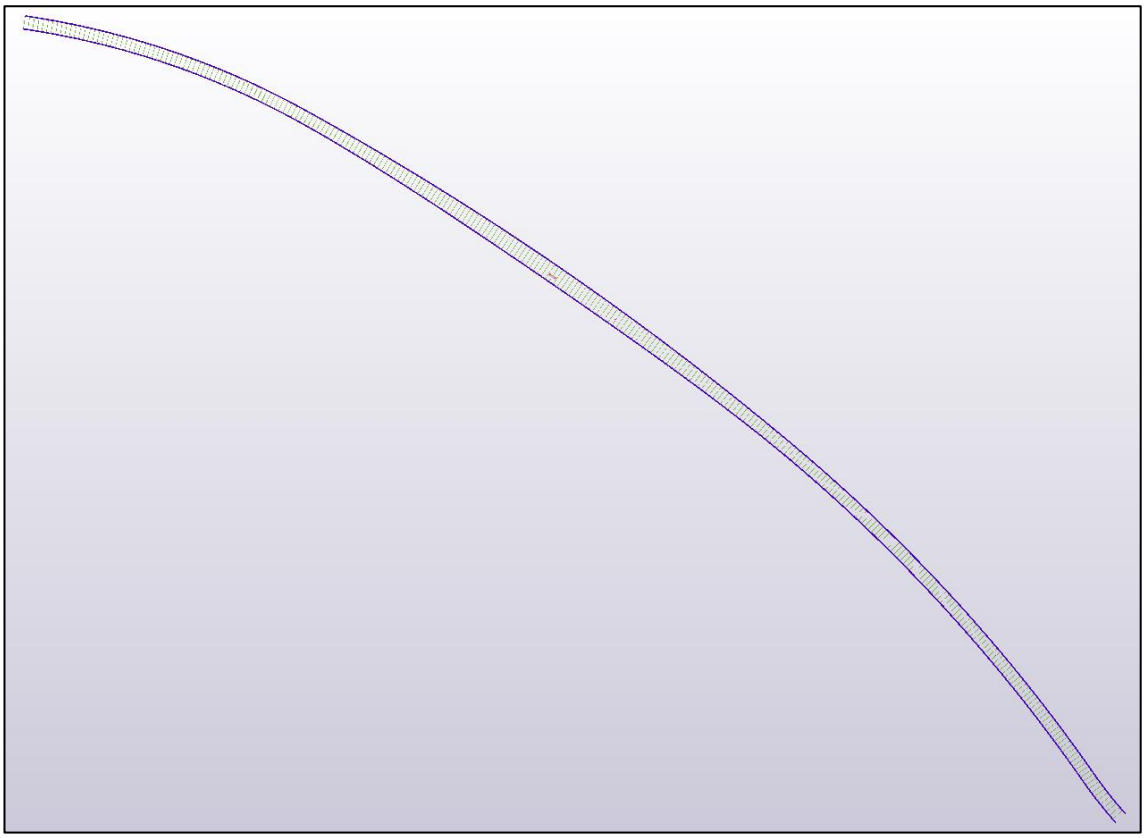
Visuaalisella skriptauksella mallinnettaessa havaittiin myös ongelma, jolloin Tekla-malliin Tekla komponenttien avulla luodut objektit katosivat näkyvistä tai objektin sijoittuivat väärään paikkaan. Tämä johtui siitä, että Teklassa ei ollut käytössä globaalikoordinaatiston työtaso vaan itse määritelty työtaso, joten Grasshopperilla mallinnetut objektit mallintuivat työtason koordinaatiston mukaan. Mallinnettaessa Grasshopperilla objekteja Teklaan on työtason oltava Teklassa origossa.

Ensimmäisenä luodussa algoritmissa tapahtui ongelma, joka aiheutti tupla geometrian Tekla-malliin eli mallissa oli täsmälleen samankokoisia päällekkäisiä objekteja. Tupla geometriaa on äärimmäisen hankala huomata visuaalisen tarkastuksen yhteydessä ja ongelman aiheuttajaa ei yksiselitteisesti löydetty. Kyseisestä ongelmasta on raportoitu myös Grasshopperin keskustelufoorumilla, mutta kyseessä on todennäköisesti käyttäjän huolimattomuudesta aiheutuva ongelma. Tupla geometrian kanssa tulisi olla tarkkana ja Grasshopperin algoritmia suunniteltaessa tulisi välttää Tekla komponenttien kopioimista. Jos koko algoritmi tai algoritmin osa kopioidaan, jossa sijaitsee Tekla komponentti, Tekla-malliin mallintuu tupla geometria.

Tutkimuksessa havaittiin, että algoritmilla mallinnetut objektit ajautuvat Teklassa tehtyjen muutoksien ohi. Esimerkkinä tilanne, jossa halutaan siirtää teräslevyä Teklassa. Tehdään muutos Teklan puolella Grasshopperiin linkitettyyn objektiin. Teräslevy siirtyy Tekla-mallissa, mutta jos algoritmi käynnistetään tai lasketaan uudestaan, objekti siirtyy takaisin algoritmin määrittelemän geometrian mukaan. Mikäli objekteja halutaan siirtää Teklassa perinteisellä tavalla, täytyy algoritmin ja Teklan välinen linkki purkaa, jotta muutokset säilyvät.

5.3 Sillan geometrian aiheuttamat haasteet

Kruunuvuorensilta on geometrisesti erittäin haastava mallinnettava kohde (kuva 17). Se on kaareva sekä pysty- että vaakageometrialtaan. Tämä aiheuttaa sen, että esimerkkitapauksessa mallinnettuun teräskoteloon tulevat teräsosat ovat kaikki sijoittuneet eri tavoin toisiinsa nähden, koska kaikki 72 pyloniin tulevaa köyttä ovat eri kulmissa. Grasshopperilla tehdyssä algoritmissa pystyttiin kuitenkin mallintamaan monta eri levyä samanaikaisesti, vaikka ne ovatkin eri kulmissa.



KUVA 17. Kruunuvuorensillan kannen vaakageometria

Kun dataa virtaa ohjelmakomponenttien välillä paljon, voi suunnittelijan olla vaikea lukea sitä. Grasshopperissa listojen käytöllä onnistuttiin kuitenkin muokkaamaan datavirtaa siten, että ne ovat suunnittelijan haluamassa loogisessa järjestyksessä. Yhden langan sisältämä suuri datamäärä on sekä mahdollisuus mallintaa samanaikaisesti, että haaste algoritmin tekemisessä.

5.4 Visuaalisen skriptauksen haasteet Grasshopperissa

Visuaalisessa skriptauksessa alkuun pääseminen on huomattavasti helpompaa kuin perinteisesti käsin koodaamalla. Suunnittelijan ei tarvitse laskea itse lähes mitään, sillä Grasshopperin ohjelmakomponentit suorittavat laskutoimitukset. Suunnittelijan haasteena onkin luoda algoritmi ohjelmakomponenteista siten, että loppujen lopuksi päästään haluttuun lopputulokseen eli Tekla objektiin. Aloittelevan käyttäjän voi olla vaikeaa käyttää ja yhdistellä ohjelmakomponentteja, jos ei tiedä mitä tietoa komponentti laskee ja mitä dataa siitä saada ulos.

5.5 Parametrisen mallintamisen ja visuaalisen skriptauksen hyödyt

Kuten aikaisemmin kerrottiin, parametriseen mallintamiseen päädyttiin esimerkkitapauksessa siksi, että teräskotelon teräsosia ei oltu vielä lopullisesti mitoitettu. Mallintamisella pystytään myös vertailemaan erilaisia vaihtoehtoja vielä rakennussuunnitteluvaiheessa tekemättä Tekla-mallia alusta asti uudestaan. Parametrisointi ja Tekla-malli pystyttiin toteuttamaan visuaalisen skriptaukseen tarkoitetulla Grasshopperilla. Parametrisoinnilla ja visuaalisella skriptauksella saadaan paljon hyötyjä, jotka on listattu seuraavaksi:

- Tekla-malli on muokattavissa reaaliaikaisesti algoritmin parametrien avulla.
- Algoritmiavusteinen suunnittelu mahdollistaa lähtötietojen muuttamisen nopeasti, siten että se vaikuttaa koko sillan geometriaan, esim. tasausviivan muutos.
- Joissain tapauksissa visuaalisella skriptauksella tehty mallintaminen voi olla nopeampaa kuin suoraan Teklaan mallintaminen.
- Erilaisia suunnitteluratkaisuja voidaan verrata keskenään nopeasti.
- Hyvin tehdyllä algoritmilla voidaan mallintaa Teklaan samanaikaisesti keskenään erilaisia objekteja.
- Sillan geometria on määritelty Rhinoon Grasshopperin algoritmilla.
- Algoritmin avulla mallinnettuja objekteja voidaan siirtää eri Tekla versioiden välillä vaivattomasti.
- Geometria voidaan siirtää suoraan laskentaohjelmaan Grasshopperista.
- Visuaalisessa skriptauksessa ei tarvitse osata koodata.

Visuaalisella skriptauksella voidaan luoda hyvinkin haastavaa geometriaa Grasshopperin monipuolisten ohjelmakomponenttien avulla. Grasshopperin käytöllä vältytään vaikeilta laskuilta ja koordinaattien laskemiselta, koska ohjelmakomponentit sisältävät mitä monimutkaisempia laskukaavoja. Mallintaminen voidaan suorittaa myös ilman objektien parametrisoimista ja käyttää hyväksi mallintamisessa ohjelmakomponentteja geometrian luomisessa.

6 POHDINTA

Suurin haaste Grasshopperista Teklaan mallintamisessa on tiedon rajallinen levinneisyys. Aiheesta on tehty suhteellisen vähän tutkimuksia, aihetta ei opeteta koululaitoksissa ja suoranaisia ohjeita Grasshopperin ja Teklan yhteiskäyttöön ei ole kovinkaan paljon. Ohjelmien välisen yhteyden mahdollistava linkki on ollut käytössä tätä kirjoittaessa vasta vuoden ja käyttäjäpiirit ovat pienet. Aiheesta tulisi tehdä enemmän tutkimuksia, jotta tieto saataisiin liikkumaan rakennesuunnittelijoiden keskuudessa. Näin ollen tutkimustavan valitseminen tapaustutkimukseksi oli erinomainen valinta.

Opinnäytetyössä saatiin tehtyä onnistuneesti parametrinen Tekla-malli Grasshopperin algoritmin avulla sillan pylonin teräskotelosta ja sen teräsosista. Ensimmäinen algoritmi saatiin tehtyä onnistuneesti, mutta koodin epäselkeyden vuoksi sitä voidaan pitää epäonnistuneena. Algoritmi tehtiin alusta alkaen uudestaan ja opittujen virheiden myötä se saatiin kohtalaisen selkeäksi. Tekla-malli saatiin parametrisoitua lähes kokonaan ja on vielä helposti muokattavissa jälkikäteen, jos siihen tapahtuu muutoksia.

Visuaalisessa skriptauksessa saatiin tärkeitä havaintoja ohjelmien välisestä toimivuudesta ja käytöstä. Grasshopperin ja Teklan yhteiskäytössä on paljon huomioitavia asioita, joista tärkeimmät ovat käyttäjän huolellisuus ohjelmia käyttäessä, algoritmin datavirran hallinta, algoritmin selkeys, yhdistettyjen Grasshopper algoritmien hyvä dokumentointi ja ohjelmien yhteistoiminnan tunteminen. Tärkeimpänä hyötynä saavutetaan Tekla-mallin reaaliaikainen geometrian muokkaus ja lähtötietoihin tuleviin muutoksiin nopea reagointiaika.

Tätä insinöörityötä voidaan pitää onnistuneena, sillä esimerkkitapauksen kohde mallinnettiin onnistuneesti ja Grasshopperin käyttöä saatiin edistettyä. Työn tekijä tutustui paljon Grasshopperin käyttöön ja aikaisempaa kokemusta ohjelmasta ei ollut. Tästä saatiin tärkeää tietoa tulevaisuutta ajatellen siirryttäessä parametriseen mallintamiseen. Jatkotutkimuksen aiheena olisi hyvä selvittää mallintamista detaljitasolla Grasshopperilla.

LÄHTEET

Davidson, S. 2017. Grasshopper www-sivu. Luettu 5.7.2017. http://www.grasshopper3d.com/

Davis, D. 2013. A History of Parametric. Luettu 8.8.2017. http://www.daniellavis.com/a-history-of-parametric/

Kangasaho, J., Mäkinen, J., Oikkonen, J., Paasonen, J., Salmela, M. & Tahvanainen, J. 2008. Pitkä matematiikka 14. Kertaus. 2. painos. Helsinki: WSOY Oppimateriaalit Oy.

Moonsoft Oy. 2017. Tuotteet. Robert McNeel & Associates. Luettu 4.7.2017. http://www.moonsoft.fi/products/000833.aspx

Tanska, T. & Österlund, T. 2014. Algoritmit puurakenteissa. Menetelmät, mahdollisuudet ja tuotanto. 1. painos. Oulun yliopisto, Arkkitehtuurin tiedekunta, DigiWoodLab.

The Morpheus Hotel: From Design to Production: Live Webinar. 2017. Video. Rhino Tutorials. Katsottu 7.7.2017. https://wiki.mcneel.com/webinars/morpheus

Trimble Solutions Corporation. 2017. Grasshopper-Tekla Live Link. Luettu 5.7.2017. https://teklastructures.support.tekla.com/not-version-specific/en/ext_grasshoppertekla-link

Trimble Solutions Corporation. Ratkaisut. Rakennesuunnittelijat. Luettu 4.7.2017. https://www.tekla.com/fi/ratkaisut/rakennesuunnittelijat

Trimble Solutions Corporation. Tuotteet. Tekla Structures. Luettu 4.7.2017. https://www.tekla.com/fi/tuotteet/tekla-structures