YRKESHÖGSKOLAN
NOVIA

Parametrinen mallintaminen Grasshopper-Tekla Live Linkillä

Berglund Kristoffer

Examensarbete för Ingenjör (YH)-examen

Byggnads- och samhällsteknik

Raseborg 2018

EXAMENSARBETE

Författare: Berglund Kristoffer

Utbildning och ort: Byggnads- och samhällsteknik, Ingenjör YH, Raseborg

Inriktningsalternativ/Fördjupning: Projektering och byggnadskonstruktion

Handledare: Mats Lindholm (Novia), Teemu Ahonen (Pöyry)

Titel: Parametrisk modellering med Grasshopper-Tekla live link

---

Datum 15.4.2018 Sidantal 27 Bilagor 1

Abstrakt

Detta examensarbete undersöker metoder att modellera svåra objekt i Tekla Structures. Objekt som har svår geometri eller mycket data att ta sin geometri ifrån kan vara arbetsdryga och svåra att hantera när man modellerar dem.

Därför undersöks en möjligtvis lättare metod att modellera med hjälp av Grasshopper-Tekla Live Link. Detta är en länk som möjliggör algoritmisk modellering genom visuell kodning via Grasshopper till Tekla Structures. Modellering med hjälp av visuell kodning av olika objekt tas fram, så som t.ex. pålars måttavvikelser, terräng och broar genom att läsa in koordinatpunkter från till exempel punktmoln eller koordinater

Det tas även i beaktan hur en ny sort av modelleringssätt kan påverka arbete och process för modellering i Tekla Structures. Arbetet söker svar på vilka möjliga fördelar och nackdelar detta kan föra med sig, samt hur en problemfri övergång till ett nytt modelleringssätt bör hanteras.

Språk: Finska Nyckelord: BIM, Grasshopper, Tekla Structures, Parametrisk modellering

---

OPINNÄYTETYÖ

Tekijä: Berglund Kristoffer

Koulutus ja paikkakunta: Rakennus- ja yhdyskuntatekniikka, Insinööri AMK, Raasepori

Suuntautumisvaihtoehto/Syventävät opinnot: Rakennesuunnittelu

Ohjaajat: Mats Lindholm (Novia), Teemu Ahonen (Pöyry)

Nimike: Parametrinen mallintaminen Grasshopper-Tekla live linkillä

Päivämäärä 15.4.2018 Sivumäärä 27 Liitteet 1

Tiivistelmä

Tässä opinnäytetyössä tutkitaan menetelmiä, joiden avulla voitaisiin mallintaa vaikeita osia ja objekteja Tekla Structuresissa. Objekteilla voi olla monimutkainen geometria tai niissä voi olla paljon dataa. Näitä osia voi olla työlästä käsitellä kun mallinnettaessa.

Tämän myötä selvitetään löytyykö mahdollisesti helpompia tapoja mallintaa vaikeita osia Grasshopper-Tekla Live Linkin avulla. Tämä linkki mahdollistaa algoritmisen mallintamisen visuaalisen koodauksen avulla Grasshopperin kautta Tekla Strucuresiin. Eri osien mallintaminen nostetaan esiin visuaalisen koodauksen avulla, esimerkkinä tästä paalujen mittapoikkeamat, maastot ja sillat. Niiden geometria parametrit voidaan lukea esimerkiksi pistepilvistä tai koordinaateista.

Tutkimuksessa pohditaan myös miten uusi mallinnustapa vaikuttaa mallinnustyöhön ja mallinnusprosessiin. Työssä etsitään vastauksia mahdollisiin etuihin ja haittoihin joita se voi tuoda, sekä siitä, miten mahdollinen siirtyminen uuteen mallinnusmenetelmiin pitäisi käsitellä.

Kieli: Suomi Avainsanat: BIM, Grasshopper, Tekla Structures, Parametrinen mallintaminen

---

BACHELOR'S THESIS

Author: Berglund Kristoffer

Degree Programme: Construction Engineering

Specialization: Structural Engineering

Supervisors: Mats Lindholm (Novia), Teemu Ahonen (Pöyry)

Title: Parametric Modeling with Grasshopper-Tekla Live Link

Date 15.4.2018 Number of pages 27 Appendices 1

Abstract

This thesis will investigate different ways to model complex objects in Tekla Structures. Objects with complex geometry or with a lot of data, from where the geometry is derived can be laborious and hard to handle when modeling in Tekla Structures.

This is why the thesis investigates if there is an easier way to model objects with the Grasshopper-Tekla Live Link. This link enables algorithmic modeling with the help of visual scripting via Grasshopper to Tekla Structures. Different objects to model are brought up as example, such as pile deviations, terrain and bridges. The geometry of these objects can be received from point clouds or coordinates.

The way the modeling approach can affect the modeling work and modeling process will be discussed. The thesis will try to find answers for possible advantages and disadvantages for this, and how a smooth transition of a new modeling process should be managed.

Language: Finnish Key words: BIM, Grasshopper, Tekla Structures, Parametric modeling

---

Sisältöluettelo

1 Johdanto ... 1
1.1 Opinnäytetyön tavoitteet ja menetelmät ... 2
2 Parametrinen mallintaminen ... 2
3 Algoritmiavusteinen suunnittelu ... 5
3.1 Algoritmiavusteinen suunnittelu visuaalisen koodauksen avulla ... 7
3.2 Linja Esimerkki ... 7
4 Rhinoceros 3D ja Grasshopper ... 9
4.1 Rhinoceros 3D ... 9
4.2 Grasshopper ... 10
4.1 Työympäristö ... 11
5 Grasshopper-Tekla Live Link ... 13
6 Grasshopper / Tekla esimerkkejä ... 14
6.1 Paalujen mallintaminen paalupoikkeamamittauksista ja näiden vertailu ... 14
6.2 Maaston mallintaminen pistepilvestä ... 16
6.3 Sillan mallintaminen ... 17
7 Päätelmä ... 18
7.1 Yleinen ... 18
7.2 Hyödyt ... 18
7.3 Haitat ... 19
7.4 Haittojen oikaisu ... 20
7.5 Loppupäätelmä ja kehitys ... 20
8 Lähdeluettelo ... 22
Sammanfattning på svenska ... 23
1 Inledning ... 23
2 Parametrisk modellering ... 23
3 Design med algoritm ... 24
4 Design med algoritm genom visuell kodning ... 24
5 Rhinoceros 3D och Grasshopper ... 25
6 Grasshopper-Tekla Live Link ... 26
7 Grasshopper / Tekla exempel ... 26
7.1 Modellering av pålar baserat på avvikelseinmätningar ... 26
7.2 Modellering av terräng från punktmoln ... 26
7.3 Modellering av bro ... 27
8 Slutledning ... 27

Liite 1

Lyhenteet ja selitykset

BIM	Building information modeling, tietomallintaminen
CAD	Computer-Aided design, tietokoneavusteinen suunnittelu
Rhino	Rhinoceros 3D, 3D-Nurbs pohjautuva mallinnusohjelma
Tekla	Tekla Structures, tietomallinnusohjelma
Nurbs	Non-uniform Rational B-spline
Plug-in	Tietokoneohjelman laajennus

1 Johdanto

Siirtyminen kynistä tietokoneperusteisiin työkaluihin eri suunnittelualoilla on tapahtunut kolmen viimeisen vuosikymmenen aikana (Tedeschi 2014). Ensin piirustuspöydän kynistä ja paperista siirryttiin erilaisiin 2D- ja 3D- piirustusohjelmiin, ja siitä tänään käytettyihin tietomallinnusohjelmiin.

Eastman (2011) ja Tedeschi (2014) molemmat käyttävät sanaa epookkinen, kun he kuvailevat siirtymistä uuteen suunnittelutapaan. Eastman käyttää sanaa epookkinen kuvailemaan suunnittelutapojen siirtymistä perinteisestä CAD- ja CADD- ympäristöstä, BIM-ympäristöön. Tedeschi taas käyttää sanaa epookkinen, kun hän kuvailee tulevaisuuden siirtymistä algoritmiavusteiseen suunnitteluun.

Tänä päivänä Eastmanin kuvaileva epookkinen siirtyminen BIM-suunnitteluun on jo aika lailla tehty. Jos haluttaisiin saada samanlainen epookkinen siirtyminen BIM:istä parametriseen mallintamiseen, on meillä tarve saada lisää tietoa kyseisestä tavasta.

Osia joilla on vaikea geometria voi olla ongelmallista ja aikavievää mallintaa. Siksi selvitetään minkälaisia etuja Grasshopperista ja parametrisestä mallintamisesta voi olla kun mallinnetaan geometrisesti vaikeita osia Tekla Structuresissä Grasshopper-Tekla Live Linkin avulla.

Opinnäytetyön tilaaja on Pöyry Finland OY ja ohjaajana yrityksestä on Teemu Ahonen. Työ sai alkunsa kun opiskelija ja Pöyry Finland OY osoittivat mielenkiintoa parametrisesta mallinnuksesta Grasshopper-Tekla Live linkin avulla. Tästä syntyi selvitys minkälainen potentiaali Grasshopper-Tekla Live linkin käytöllä voi olla erilaisissa mallinnustöissä.

Pöyry Finland OY on osa Pöyry Oyj konsernia. Pöyry Oyj on kansainvälinen konsultointi- ja suunnitteluyritys, joka suunnittelee ja toteuttaa energia-, teollisuus- ja infratoimialojen hankkeita. Yrityksellä on toimistoja 40 eri maassa ja se työllistää noin 5500 henkilöä. (Pöyry Oyj)

Pöyry Finland Oy:llä on suunnittelijoita rakennusalan eri aloilla. Näillä eri suunnittelualoilla käytetään BIM:iä suunnittelutapana päivittäin. BIM-työkaluja kehitetään Pöyryn eri suunnittelualoilla, jotta voidaan seurata tämän päivän ja tulevaisuuden BIM-teknologia. (Teemu Ahonen, Pöyry Finland OY)

1.1 Opinnäytetyön tavoitteet ja menetelmät

Tässä työssä mallinnetaan osia Tekla Structuresissä Grasshopper-Tekla Live linkin avulla. Työssä tutkitaan minkälaisia hyötyjä linkki voi antaa, liittyen seuraaviin asioihin:

- Mallinnustyö ja -prosessi
- Geometrisesti vaikeiden osien mallintaminen
- Mallinnustyöltään työläiden osien mallintaminen

Mallinnusgeometrian pääosat perustuvat koordinaateista ja/tai pistepilvidatasta.

Ohjelmat joita on käytetty tässä opinnäytetyössä:

- Tekla Structures 2017
- Rhinoceros 5.0
- Grasshopper
- Grasshopper-Tekla Live Link

Opinnäytetyö keskittyy pääasiallisesti mallintamiseen Tekla Structuressa Grasshopperin avulla. Ennen kuin itse mallinnukseen tutustutaan, selvitetään tärkeitä käsitteitä, jotka liittyvät parametriseen mallintamiseen.

Työssä käytetään käsitettä *parametrinen mallintaminen*. Tämä tarkoittaa parametrista mallintamista käyttäen algoritmiavusteista suunnittelua hyödyntämällä visuaalista koodia. Käsitteitä liittyen parametriseen suunnitteluun on paljon ja ne ovat erilaisia koska tämänkaltainen mallinnus tai suunnittelu on vielä kohtalaisen uutta.

2 Parametrinen mallintaminen

Davis (2013) kirjoittaa selkeän määritelmän mitä parametrinen malli voisi olla ja mitkä ovat parametrisen mallintamisen perustoiminnot: *”Thus, a parametric model can be defined as: a set of equations that express a geometric model as explicit functions of a number of parameters”*, eli parametrista mallia voisi kuvata sarjana yhtälöitä jotka ilmentävät geometristä mallia, tiettyjen parametrien eksplisiittisinä funktioina.

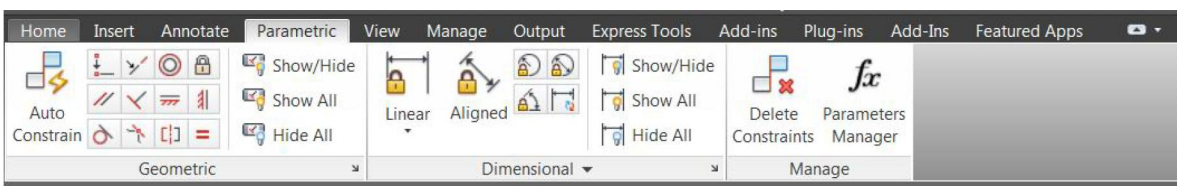
Erilaiset käsitteet parametrisessa suunnittelussa ovat vielä epäselviä. Käsitettä parametrinen suunnittelu käytetään yleensä samassa merkityksessä kuin algoritmiavusteinen suunnittelu. Käsitteistä tulee vielä vaikeampia käyttää kun englannin kielessä käytetään myös *Parametric modelling*, mikä yleensä ei viittaa mallintamiseen BIM-ohjelmassa, vaan mallintamiseen 3D-mallinnusohjelmassa. Tässä opinnäytetyössä käytetään silti käsitettä parametrinen mallintaminen, minkä voi perustella sillä, että Suomessa käytetään käsitettä *mallintaminen* tietomallintamisen yhteydessä.

Käsitteitä algoritmiavusteinen suunnittelu ja parametrinen suunnittelu käytetään joskus samassa merkityksessä, mutta algoritmiavusteinen suunnittelu on yksi tapa toteuttaa parametrinen suunnittelu.

Parametrisen suunnittelun voi myös toteuttaa manuaalisesti, kuten esimerkiksi Antoni Gaudi teki, ripustamalla erilaisia kankaita kattoon. Siten hän sai esille erilaisia holvimuotoja. Parametreinä, Gaudi käytti naruja, jotka oli kiinnitetty kankaiden kulmiin ja painoja kankaiden keskipisteessä. Säätämällä narujen pituuksia ja painojen määrät, syntyi erilaisia holvimuotoja. (Davis 2013; Tedeschi 2014).

Ensimmäinen tietokoneohjelma mikä mahdollisti parametrisen suunnittelun, oli ohjelma nimeltä Sketchpad. Vuonna 1963 Ivan Sutherland kehitti Sketchpad kuvitetun tietokoneen graafisena kommunikaatiosysteeminä. Sketchpadilla oli samanlaisia toimintoja kuten tämän päivän CAD-ohjelmissa ja tämän lisäksi ohjelma oli perustunut assosiatiiviseen logiikkaan. Tämä mahdollisti toimintoja mikä loi suhteita ja riippuvuuksia osien välillä; siirtämällä yhtä pistettä mistä kaksi erilaista linjaa oli piirretty, molempien linjojen muoto päivittyi. Tämän tapaiset assosiatiiviset ominaisuudet saivat tukea vasta melkein kolmen vuosikymmenen jälkeen suositetuissa CAD-ohjelmissa. (Tedeschi 2014)

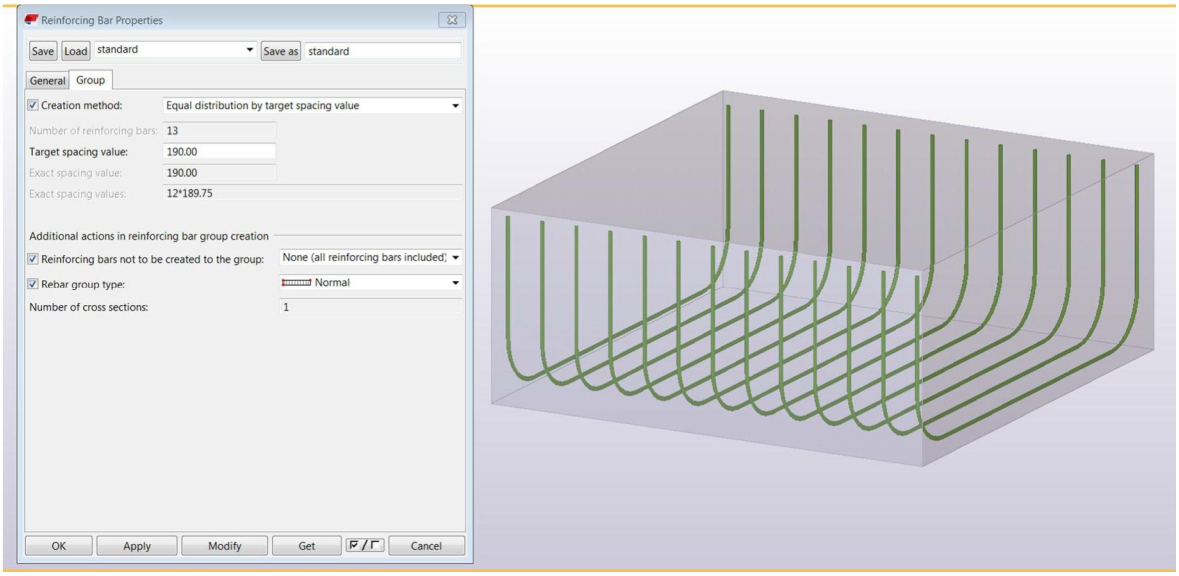
Nykyisin CAD-ohjelmat mahdollistavat parametrisen suunnittelu ohjelman käyttöliittymässä. Esimerkiksi, Autocadilla on kokonainen kohta nimeltä *Parametric*, kuten kuvassa 1 esitetään. Näillä toiminnoilla voi kytkeä osia toisiinsa, ja tehdä riippuvuuksia osien tai linjojen välillä. (Tedeschi 2014).



Kuva 1 Parametrisia vaihtoehtoja Autocadissa.

BIM-ohjelmilla kuten Tekla Structuresillä on monta matalantason parametrisia käyttöjä jotka ovat valmiiksi sisäänrakennettu ohjelman komponentteihin tai työkaluihin. Nämä mahdollistavat nopeita muutoksia osille ja kuuluville osille. (Eastman 2011).

Kuvassa 2 on näytetty raudoitustyökalu Teklassa, se on parametrisesti kiinnitetty betoniosaan. Parametrit jotka ohjaavat raudoituksen geometriaa ovat muun muussa raudoituksen halkaisija, betonisuojapeite ja pisteet jotka on kiinnitetty betoniosaan. Raudoituksella on parametreja jotka on määritetty betoniosan geometriasta, mutta myös parametreja, jotka on ohjattu manuaalisesti kirjoittamalla tai valitsemalla parametrit raudoitustyökalussa. Raudoituksen voi saada mallinnettu niin, että se seuraa anturan kokoa ja sijaintia.



Kuva 2 Esimerkki parametrisesta ominaisuudesta Tekla Structuressa, jos anturan koko muuttuu, raudoitus mukautuu anturan kokoon.

Eastmanin (2011) mukaan, tärkeitä ominaisuuksia parametrisesta BIM-osasta ovat:

- Geometrialla on assosiatiivisia tietoa ja sääntöjä.
- Parametrisiä sääntöjä muokkaa automaattisesti muuttuva geometria.
- Osille voi määrittää tietyn hierarkian.
- Parametrisiä sääntöjä voi tunnistaa jos tiettyä muutosta ei voi suorittaa.
- Osilla on kyky kytkeä, vastaanottaa, siirtää ominaisuuksia ja tietoa.

BIM-ohjelmiin sisäänrakennettuja matalatasoisia parametrisia ominaisuuksia on ohjelmissa työkaluina tai komponenttien muodoissa. Näitä työkaluja ja komponentteja on rajoitettu sen mukaan miten ne ovat ohjelmoitu toimimaan. Algoritmiavusteisen suunnittelun avulla voidaan laajentaa BIM-ohjelmien parametriset ominaisuudet teksti- tai visuaalisen koodauksen avulla.

3 Algoritmiavusteinen suunnittelu

(Tedeschi 2014) kirjoittaa: *”An algorithm is a procedure used to return a solution to a question – or to perform a particular task – through a finite list of basic and well defined instructions.”*, eli algoritmiavusteinen mallintaminen on menettely missä palautetaan ratkaisu kysymykseen – tai suoritetaan tietty tehtävä – finiittisen listan avulla jossa on vaatimattomat ja hyvin määritellyt ohjeet.

Mallintaminen perustuu perinteisesti virtuaaliseen manipulaatioon, eli käyttämällä hiirtä ja näppäimistöä digitaalisessa ympäristössä. Sitä vastoin, algoritmiavusteinen suunnittelu perustuu logiikkaan käyttämättä manuaalista muokkausta. Algoritmit ovat riippuvaisia luomaan käsitteellisiä yhteyksiä geometrian ja matematiikkaan välillä, eli muokataan dataa digitaalisten osien sijaan. Jotta algoritmin ohjeet onnistuisivat, on tarve saada input ohjeille. Inputien pitää olla juuri niitä jotka ohjeessa on määritelty. Esimerkiksi teksti ei saa olla numero jos ohje on määrittänyt että se haluaa ainoastaan tekstiä. Ohjeet pitää olla hyvin määritelty että inputit tietävät mitä tule tapahtumaan ja missä järjestyksessä, muuten algoritmia ei voi saada toimimaan. (Tedeschi 2014)

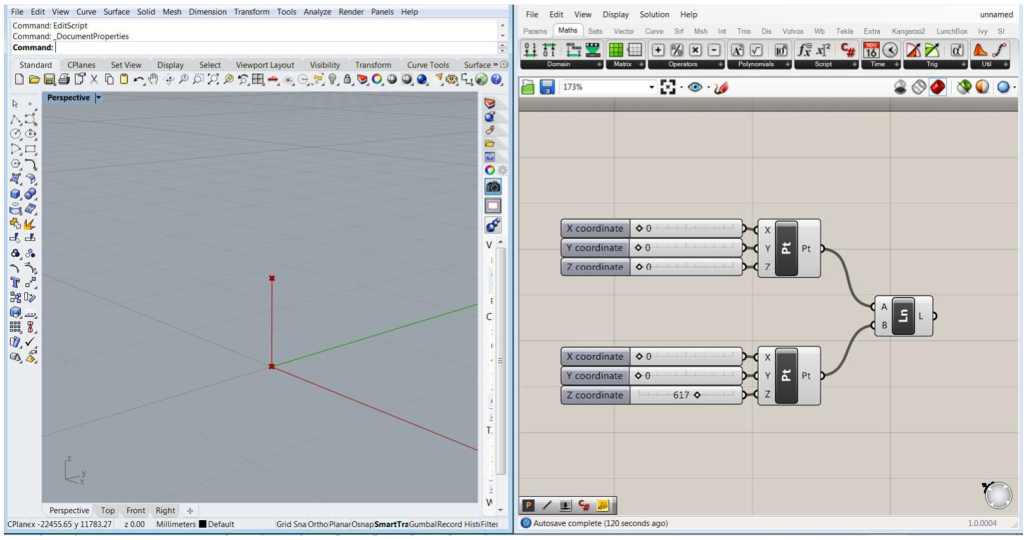
Kuva 3 Skemaattisen kuvaus algoritmista.

Kun tekee työtä algoritmiavusteisen suunnittelun kanssa tietokoneella, jonkinlaiselle koodille on tarvetta. Tämä koodi määrittää mitä algoritmissa pitäisi tehdä. Tämä koodaus koostu kahdesta eri työympäristöstä:

1. Koodauseditori

2. Mallinnusympäristö

Koodauseditorille koodataan mitä halutaan näyttää mallinnusympäristössä. Se mikä mallinnusympäristössä näytetään esimerkiksi geometriana, on näytetty myös koodauseditorissa algoritmina, eli nämä ympäristöt vastaavat toisiaan ja output tulee molemmille.



Kuva 4 Mallinnusympäristö Rhinossa vasemmalla ja koodauseditori Grasshopperissa oikealla.

Koodi voi koostua tekstikoodista, visuaalisesta koodista tai näiden yhdistelmästä. Tässä työssä keskitytään enimmäkseen visuaaliseen koodaukseen.

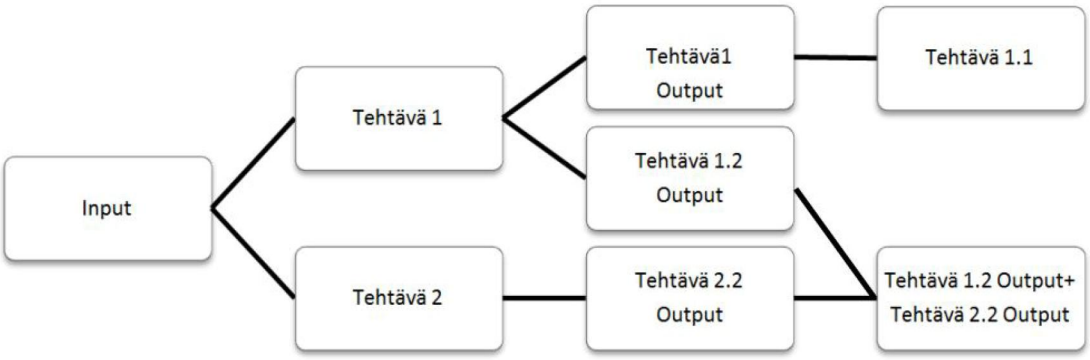
```
Option Explicit
'Script written by <insert name>
'Script copyrighted by <insert company name>
'Script version Friday, 08 December 2017 23:49:58

Call Main()
Sub Main()

    Dim arrStart, arrEnd

    arrStart = Rhino.GetPoint("Start of line")

    If IsArray(arrStart) Then

        arrEnd = Rhino.GetPoint("End of line")

        If IsArray(arrEnd) Then

            Rhino.AddLine arrStart, arrEnd

        End If

    End If


End Sub
```

Kuva 5 Linjan luominen tekstikoodina.

Kuva 6 Linjan luominen visuaalisena koodina.

3.1 Algoritmiavusteinen suunnittelu visuaalisen koodauksen avulla

Visuaalinen koodi koostuu yhdestä tai monesta parametrisesta tehtävästä, joissa on yhteneväisyyksiä ja riippuvuuksia parametreissä, ohjeissa ja outputeissa. (Woodbury 2010; Tedeschi 2014). Skemaattinen kuvaus tästä on annettu kuvassa 7

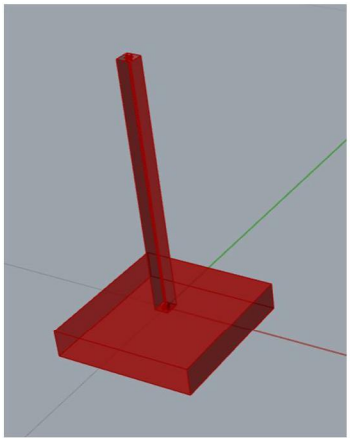


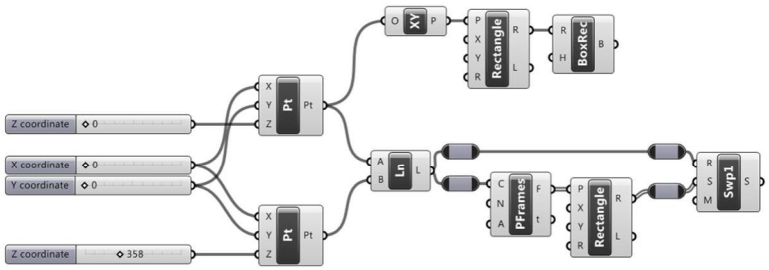
Kuva 7 Skemaattinen kuvaus algoritmiavusteisesta suunnittelusta visuaalisen koodauksen avulla.

Visuaalinen koodaus on hyödyllinen työkalu kun tekee töitä algoritmiavusteisen suunnittelun kanssa. Tietoa tekstikoodauksesta ei välttämättä tarvitse, mutta tekstikoodausta voi käyttää visuaalisen koodauksen yhteydessä mikä tehostaa algoritmia eri tavoin. (Bachman 2017)

3.2 Linjaesimerkki

Jos geometria kuvassa 8 esittää pilaria ja anturaa, tämä pilari pitäisi piirtää uudelleen tai kopioida manuaalisesti oikeaan paikkaan suunnitteluun mukaan sekä osiin liittyvät osat.

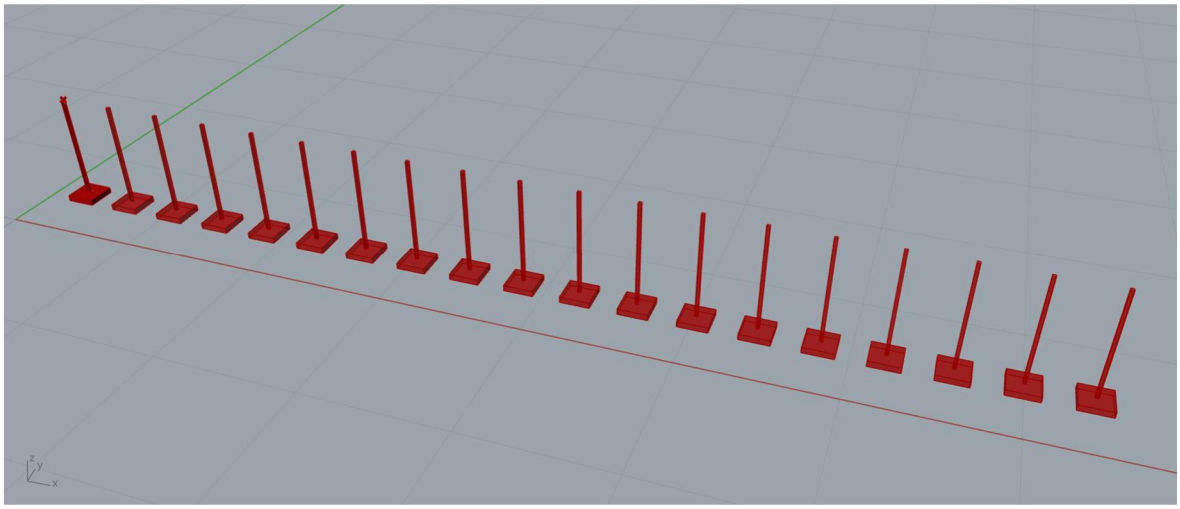




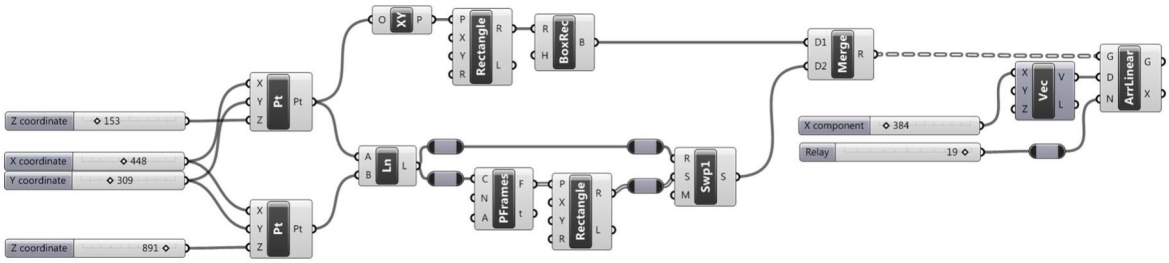
Kuva 8 Rhino-geometria . **Kuva 9 Visuaalinen koodi kuvan 8 geometriasta.**

Kun hyödyntää parametrista mallintamista geometriaan tai kuten tässä esimerkissä pilarille ei tarvitse manuaalisesti antaa aloitus- ja päätypisteitä mallinnusympäristössä. Pilarin geometria voi sen sijaan olla määritelty siihen liittyvien osien pisteille tai linjoille ja pilarin geometria voi taas määritellä muiden siihen liittyvien osien geometrian tai sijainnin.

Tässä yksinkertaisessa esimerkissä anturan yläpään keskipiste on kiinnitetty pilarin alapään keskipisteelle eli jos pilarin sijainti muuttuu X-, Y- tai Z-suunnassa, antura seuraa sen mukana, vaikka pilarin sijaintia muuttaisi miten haluaa. Kuvassa 10 ja 11 on myös näytetty miten helposti osien kopiointi sujuu.



Kuva 10 Rhino-geometria. Pilari on kytketty anturalle.



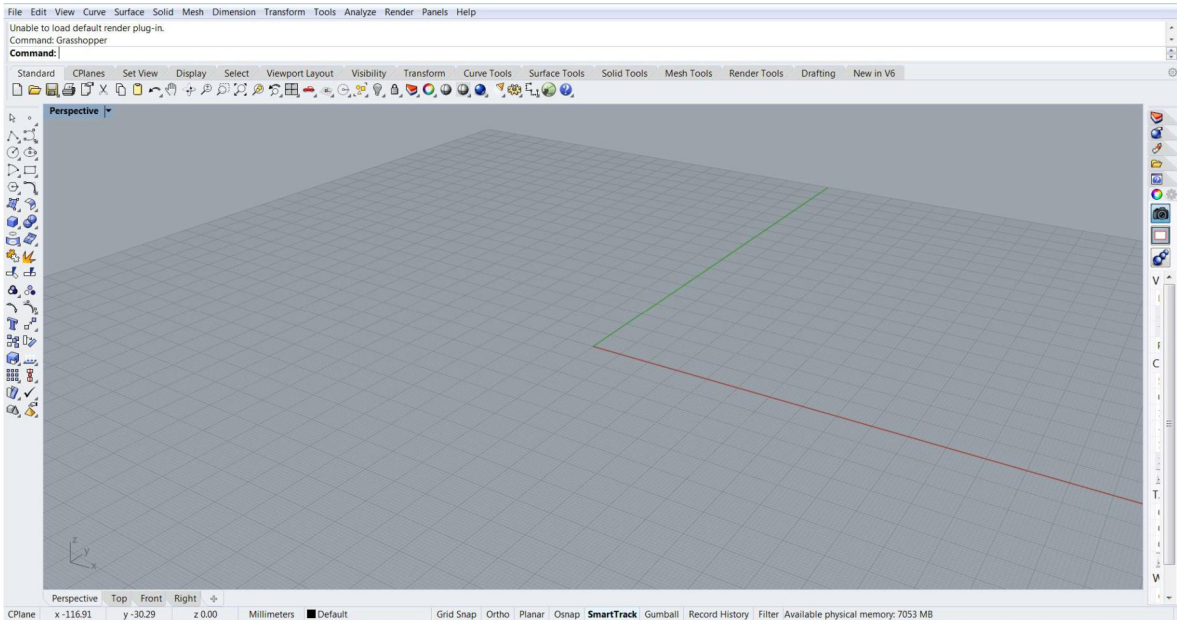
Kuva 11 Visuaalinen koodi geometriasta kuvassa 10.

Tämänkaltainen parametrinen mallintaminen voi vielä jatkua muille osille. Esimerkiksi moduulilinjan risteyskohdalle voi antaa parametrisia arvoja pilarille ja pilarin geometria voi antaa parametiset arvot kattopalkeille, seinille ja muille niihin liittyville rakennusosille. Kattopalkille voisi antaa parametriset arvot kattorakenteisiin ja niin edelleen. Tämä voi olla hyödyllistä kun yhteen osaan liittyviä muutoksia voi päivittää muihin osiin saman tien.

4 Rhinoceros 3D ja Grasshopper

4.1 Rhinoceros 3D

Rhinoceros 3D tai myös pelkällä Rhino-nimellä tunnettu ohjelma on suosittu NURBS-geometriapintamallinnusohjelma (kuva 12), joka julkaistiin vuonna 1998. Siitä on tullut erittäin suosittu arkkitehtien ja eri muotoilijoiden piireissä. Rhino tukee moninaisia pintamallinnuskykyjä tekemällä, muokkaamalla ja analysoimalla erilaisia monimutkaisia pintoja. Näillä kyvyillä voi tehdä erilaisia monimutkaisia muotoja. (Eastman 2011, 207-209)



Kuva 12 Rhino-käyttöliittymä.

Tässä työssä ei perehdytä Rhinoon, mutta pitää omistaa käyttölisenssin Rhinoon jos haluaa käyttää plug-in Grasshopper. Rhino voi saadaa ilmaiseksi www.rhino3d.com sivustolta 90 päiväksi.

4.2 Grasshopper

Grasshopper joka on plug-in Rhinoceros 3D:lle, on visuaaliseen koodaukseen pohjautuva muokkausohjelma, jonka David Rutten kehitti Rober McNeel&Associatesin kanssa. Alkuperäisnimi Grasshopperille oli Explicit History joka julkaistiin vuonna 2007 Rhino 4.0:lle ja vuonna 2008 nimi muutettiin Grasshopperiksi. Grasshopper on ilmainen plug-in-ohjelma Rhinolle. (Tedeschi 2014, 33).

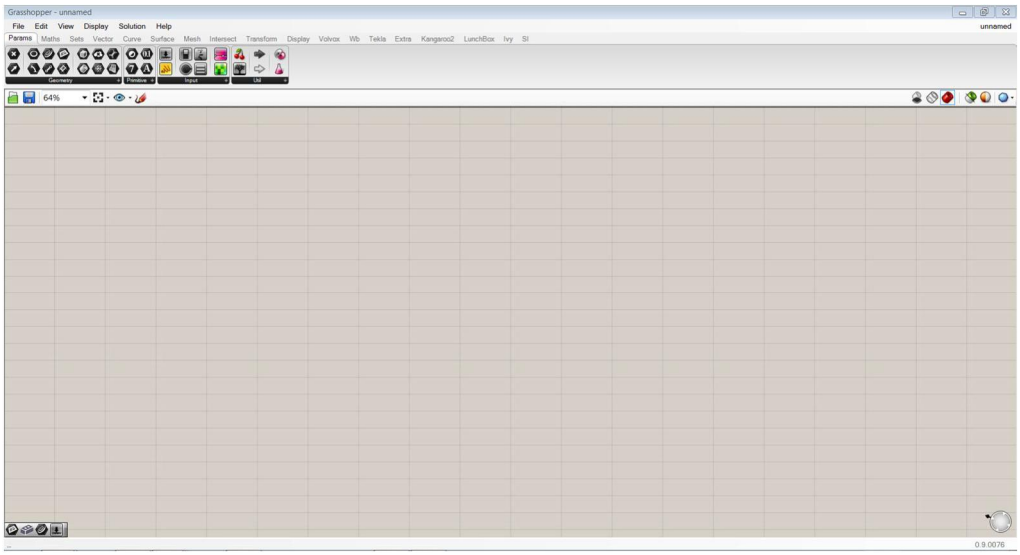
Grasshopperilla käyttäjä pystyy tekemään visuaalisen koodin vetämällä komponentteja canvakselle ja yhdistämään näitä johtojen kanssa. Vaikka on kyse visuaalisesta koodauksesta, ei ole tarve tietää mitään koodauksesta. Grasshopperillä voi tehdä loputtomia muokkauksia työhön, ja koodille voi antaa lähtötietona esimerkiksi numeroita tai muita parametrisiä arvoja. (Bachman D. 2017 2-3)

Grasshopper voi reaaliaikaisesti päivittää muita ohjelmia erilaisten plug-inien avulla. Tunnetuimpia ohjelmia tähän tarkoitukseen on esimerkiksi Excel, Photoshop, Autodesk Revit, Tekla Structures, Archicad ynnä muita. (Tedeschi 2014, 34).

On olemassa myös plug-inejä Grasshopperille, muun muussa Kangaroo, Lunchbox ja Weaverbird, jotka tehostavat vielä enemmän jo tehokasta Grasshopper. Nämä plug-init mahdollistavat esimerkiksi fysiikka tai rakenteellisia simulaatioita ja verkkojen muokkausta. Nämä plug-init ovat saatavilla www.food4rhino.com-sivustosta.

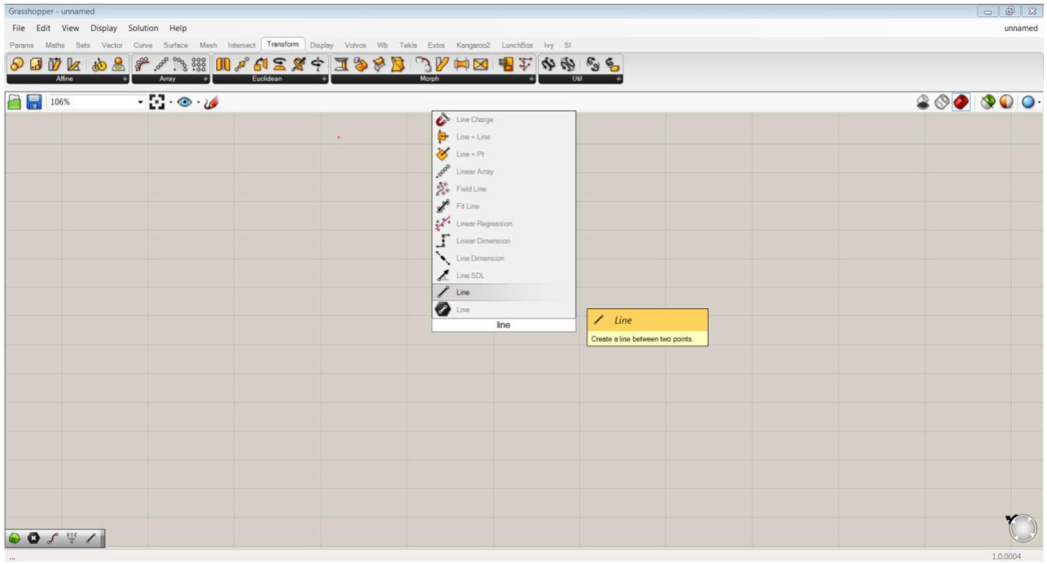
4.1 Työympäristö

Canvas (kuva 13) on alusta Grasshopperissa jolla työ suoritetaan. Canvakselle voi vetää tarvittavia komponentteja minne haluaa, mutta hyvä pääsääntö on että visuaalinen koodi tehdään vasemmalta oikealle.

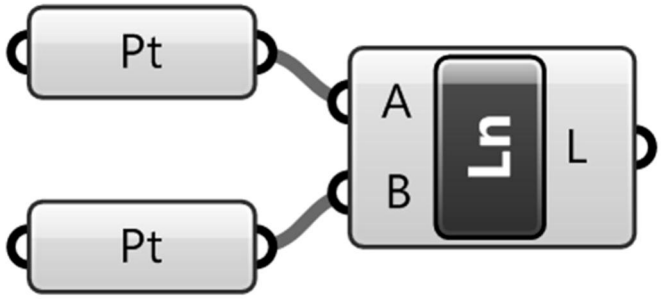


Kuva 13 Grasshopper-canvas.

Komponentit haetaan komponenttipaneelista joka on canvaksen yläpuolella. Komponentit voidaan myös hakea kun kirjoittaa canvakselle (kuva 14) haluamansa komponentin nimen. Komponentit yhdistetään johdoilla työn mukaan (kuva 15).

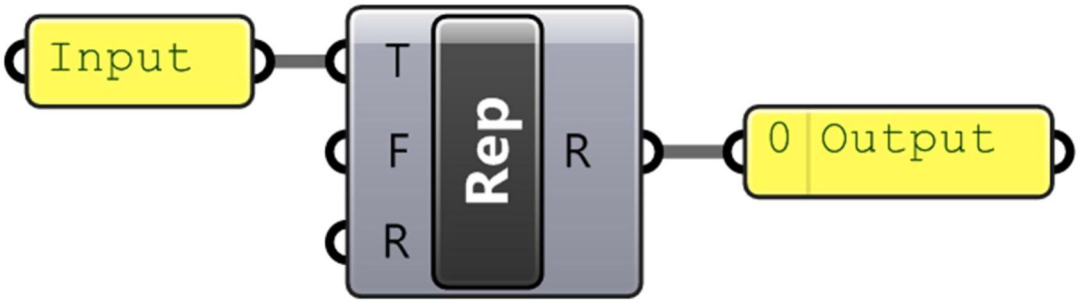


Kuva 14 Yksi tapa hakea tarvittava komponentti.



Kuva 15 Komponentteja, jotka on kytketty toisiinsa johdoilla.

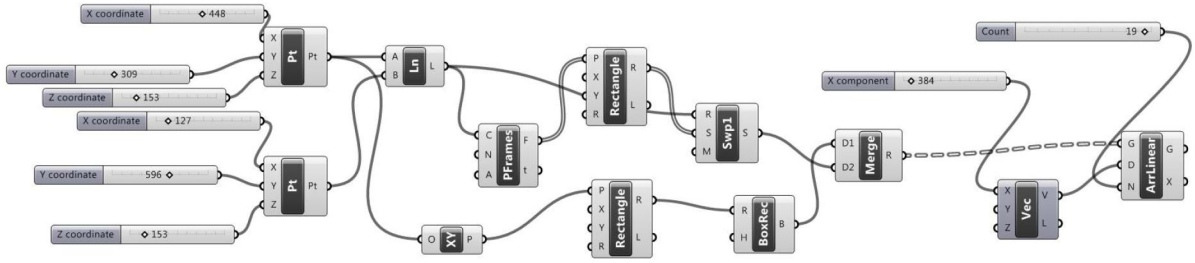
Input ja output komponenteille voi olla joko geometriaa, dataa tai operaatioita. Tietoa virtaa aina komponentissa vasemmalta oikealle, joten johdot, jotka on kytketty komponentin vasemmalle puolelle edustaa inputia ja johdot, jotka on kytketty oikealle puolelle edustavat outputia. (kuva 16)



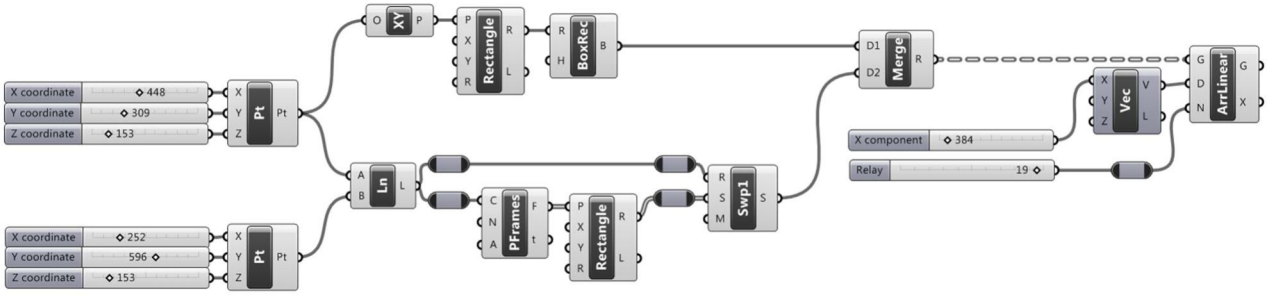
Kuva 16 Komponentti *Replace text.* Kuvassa komponentti vaihtaa kirjaimet *In* kirjaimille *Out.*

Visuaalisen koodin täytyy olla hyvin strukturoitu selkeällä hierarkialla. Kun on käytössä iso tai pitkä visuaalinen koodi, erilaisia komponentteja voi ryhmittää tehtävän mukaan ja näille komponenteille voi määritellä selkeästi ja loogisesti mitä ne tekevät. Tämä helpottaa jos joku muu käyttäjää halua käyttää jonkun muun tekemää algoritmia, tai jos algoritmin viimeisestä käytöstä on kulunut aikaa. (Bachman 2017)

Kuvassa 17 ja 18 on algoritmin output täysin sama, mutta algoritmi on rakennettu eri tavalla. On tärkeää tehdä selkeä algoritmi, että tietää mistä parametrit tulevat ja mihin ne menevät, muuten algoritmi voi epäselvyyden takia epäonnistua tulevaisuudessa kun sitä taas käytetään.



Kuva 17 Epäselkeä ja huonosti tehty visuaalinen koodi.



Kuva 18 Selkeämmin tehty visuaalinen koodi kuin kuvassa 17 on näytetty.

5 Grasshopper-Tekla Live Link

Grasshopper-Tekla live link mahdollistaa algoritmisen mallintamisen Tekla Structuresiin-ohjelmaan käyttämällä Rhino/Grasshopperia. Linkki on ominaisuus Grasshopperin komponenteista mikä voi luoda ja vuorovaikutta osia reaaliaikaisesti Tekla Structures. (Trimble Solutions Corporation 2018)

Linkistä löytyy melkein kaikki samat työkalut mikä on Teklassa paitsi raudoitus-, pultti- ja hitsaustyökalut. Mallintamisen näistä voi tehdä käyttämällä sopivia Tekla-komponentteja Grasshopperissa.

6 Grasshopper/Tekla esimerkkejä

Grasshopper-Tekla Live Linkin avulla olen tehnyt kolme erilaista esimerkkiä jotka mallintavat erilaisia osia Teklassa:

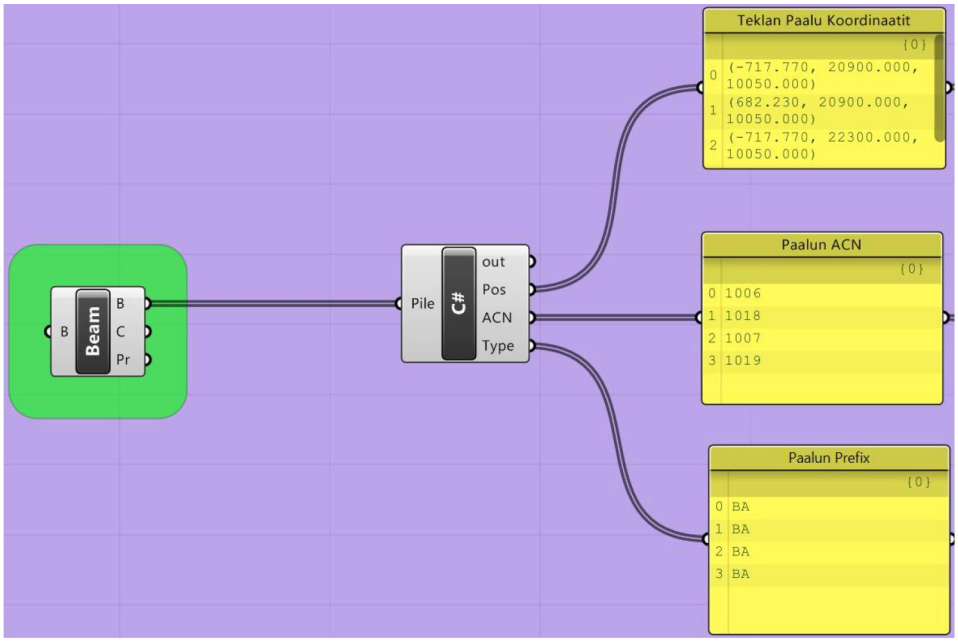
1. Paalujen mallintaminen paalupoikkeaman mittauksista ja näiden vertailu
2. Maaston mallintaminen pistepilvestä
3. Sillan mallintaminen

6.1 Paalujen mallintaminen paalupoikkeamamittauksista ja näiden vertailu

Paalujen sijaintipoikkeamien mallintaminen voi olla työläistä ja aikavievää. Paalujen mitatut sijainnit työmaalla voi antaa Excel-muodossa josta paalujen tunnus, koordinaatit, kallistuma ja suunta käy ilmi.

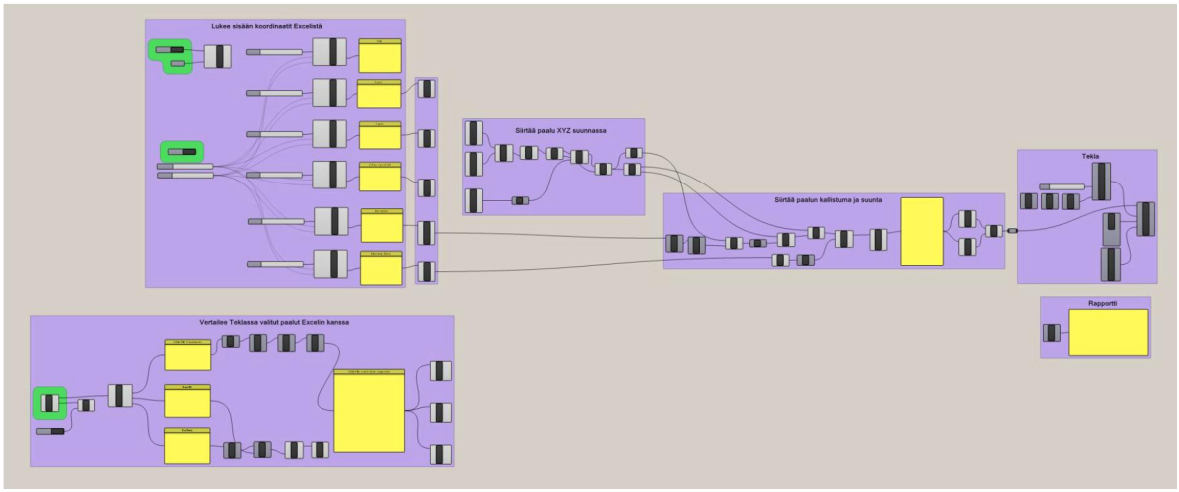
Tämän perusteella, algoritmista jonka olen kehittänyt Grasshopperissa voi lukea sijaintipoikkeamat Excel taulukoista ja siitä sijoittaa ne tarkkaan todelliselle paikalle Teklassa. Algoritmi voi päivittää paalujen koordinaatit, kallistumat, suunnat ja tarvittaessa muut ominaisuudet Teklassa, kuten esimerkiksi paalun nimen ja luokan.

Oletuksena oli että paalut on jo mallinnettu Teklassa käyttäen betonipalkkikomponenttia. Betonipalkin geometria Teklassa on otettu kahdesta pisteestä; alkupiste ja päätypiste. Palkin profiiliin on sitten kiinnitetty alku ja päätypisteelle, jotka muodostavat linjan. Betonipalkin geometria hyödynnetään, eli siirretään Teklasta Grasshopperiin halutut palkkitiedot (kuva 19). Kun valitsee Teklassa ne paalut jotka halutaan sijoittaa todelliseen sijaintiin, algoritmi vertaa Teklassa olevien paalujen koordinaatit, kallistuman ja suunnan.



Kuva 19 C#-komponentin avulla voi hakea paaluista erilaisia tietoja.

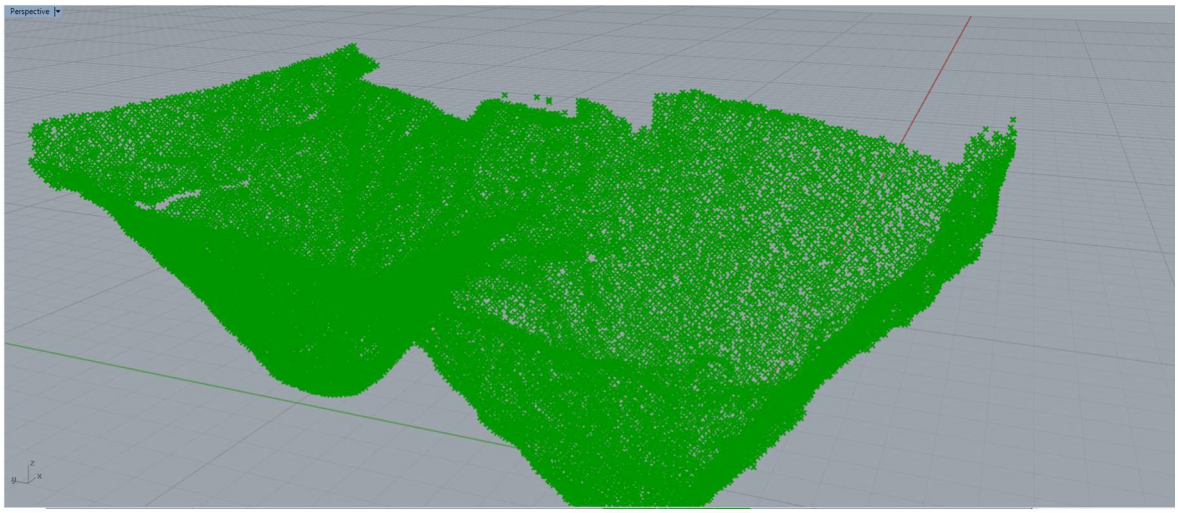
Alkuperäiset paalut, jotka on mallinnettu Teklassa suunniteltuun sijaintiin, eivät muuta paikkaa. Algoritmi mallintaa uusia paaluja josta saadaan paalujen todelliset sijainnit paalujen sijaintipoikkeaman perusteella. Syynä on että paalujen suunnitellut sijainnit on hyvä pitää mallissa oikealla paikalla, jos esimerkiksi haluaa tarkistaa lähtötilanteen. Kuvassa 20 on näytetty algoritmi, joka mahdollistaa paalujen sijoittamisen malliin paalupoikkeamamittauksien avulla.



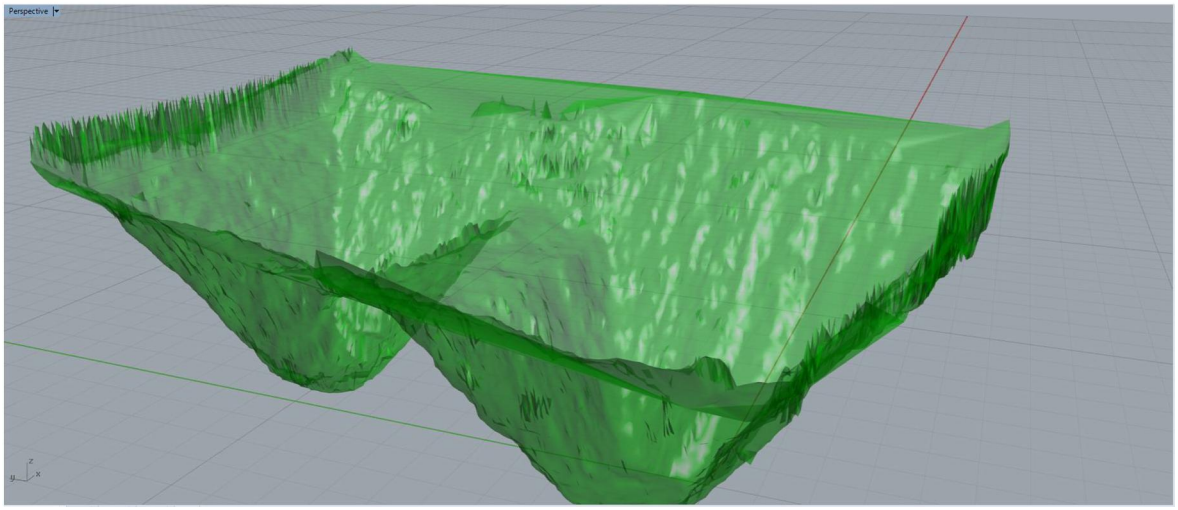
Kuva 20 Algoritmi joka mahdollistaa paalujen mallintamisen lukemalla koordinaatteja Excelistä.

6.2 Maaston mallintaminen pistepilvestä

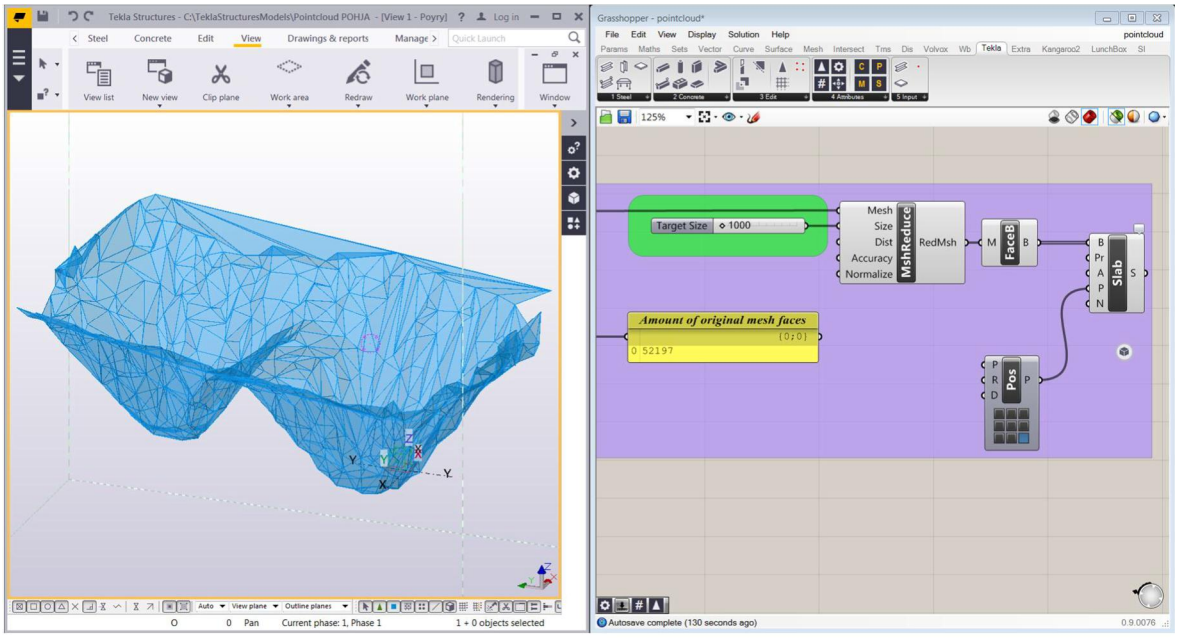
Oletuksena käytin pistepilvitiedoston .las muotona, mistä vein export-komennolla tiedoston .pts tiedostotyypille Autodesk Recapissa. Näin sain X-, Y-, ja Z-koordinaatit Notepadille. Jokaiselle pisteelle pistepilvitiedostossa on X-, Y, ja Z-koordinaatit. Niistä algoritmi lukee koordinaatit notepadista ja tekee pisteitä Grasshopperiin (kuva 21). Kun pisteet on tehty Grasshopperissa, voidaan sitoa ne yhteen käytämällä Delaunay triangulation-komponenttia joka tekee kolmioita pisteiden välille (kuva 22). Kytkemällä kaikki muodostuneet kolmiot, Teklan *Slab*-komponentilla saadaan tehtyä verkko pistepilvestä Teklassa (kuva 23). Kuvassa 24 on näytetty algoritmi, joka mahdollistaa tämän.



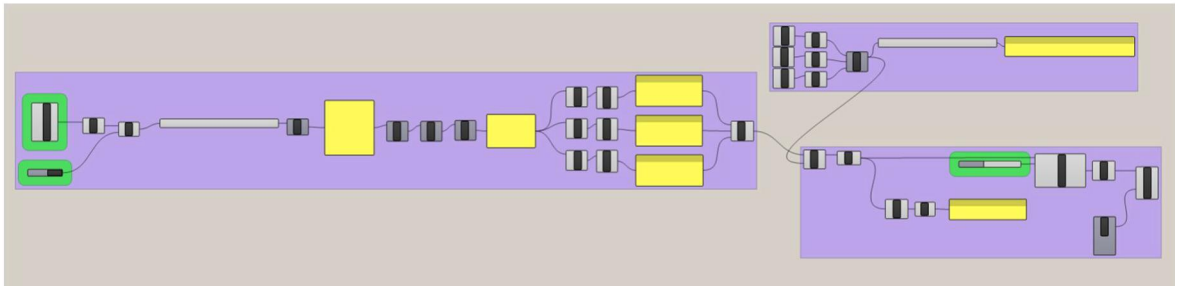
Kuva 21 X-, Y-,Z- koordinaatit luettuina pisteinä Grasshopperissa.



Kuva 22 Delaunay kolmiomittaus.



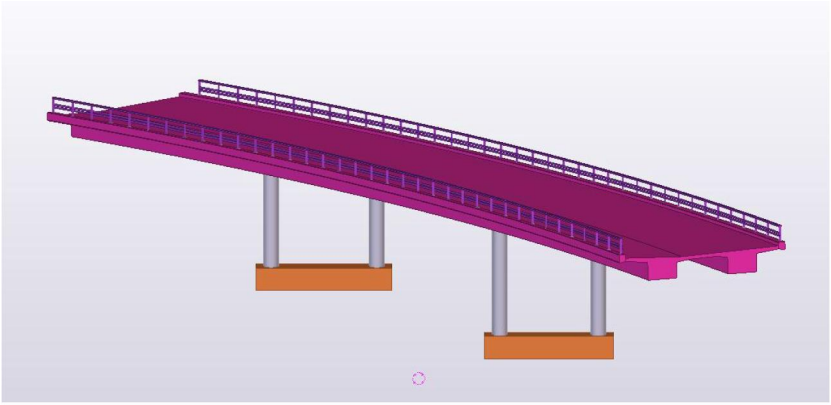
Kuva 23 Pistepilvi esitettynä Teklassa.



Kuva 24 Grasshopperin algoritmi joka mahdollistaa maasto-pistepilvien siirtämisen datasta suoraan Teklaan.

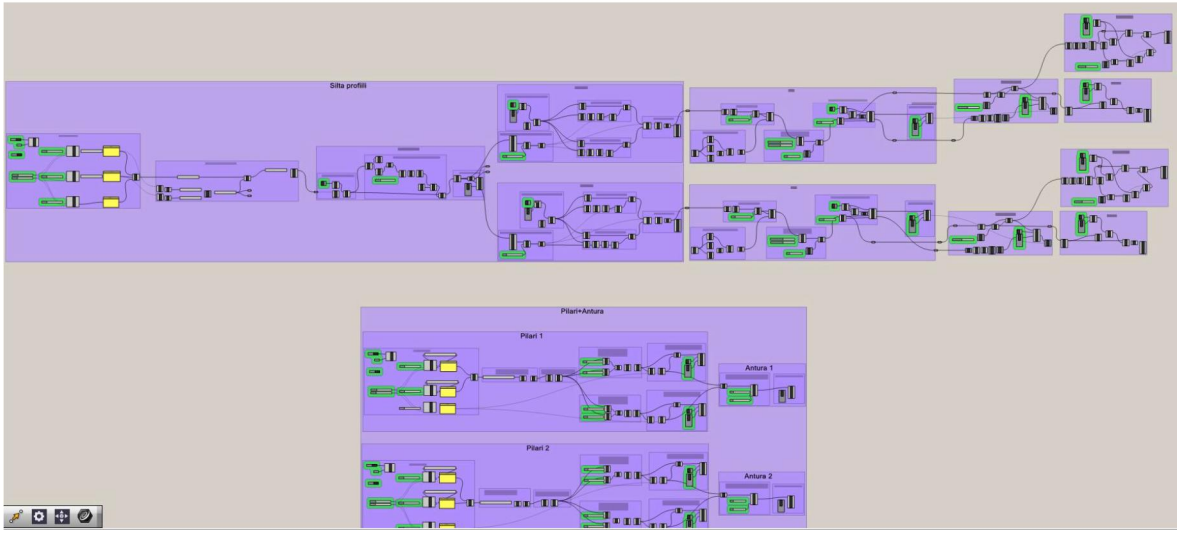
6.3 Sillan mallintaminen

Oletuksena käytin betonisillalle tielinjaa joka oli Excel-muodossa X-, Y-, ja Z-koordinaatteina. Siirtämällä koordinaatit Grasshopperiin voidaan koordinaattipisteestä interpoloida linja. Linjaan kytketään sillan poikkileikkaus, päällysrakenne ja reunapalkki ja näihin kolmeen osaan on kytketty sillan muut osat. Tietty hierarkia on näin saavutettu, mikä mahdollistaa että yksittäisiä osakokonaisuuksia voidaan muokata helposti. Tämä hierarkia mahdollistaa myös koko sillan päivittämisen jos haluaa muuttaa linjan X-, Y-, ja Z- pisteitä, eli tielinjaa siirtämällä päivittää samalla sillan kaikki osat. Kuvassa 25 on mallinnettu silta Tekla-ohjelmassa ja kuvassa 26 on algoritmi, joka on mahdollistanut sen.



Kuva 25 Sillan mallintaminen Grasshopper-Tekla live linkillä.

Sillan pääprofiilin siirsin Tekla-ohjelmaan käyttämällä *Define cross section using DWG file* Teklassa. Se on hyvä tapa aikaansaada erilaisia profiileja, jotka eivät ole Teklan oletuksena. Profiilin voi sitten sijoittaa ja kytkeä tielinjalle.



Kuva 26 Algoritmi mikä mahdollisti betonisillan mallintaminen Teklassa.

7 Päätelmä

7.1 Yleinen

Tämän opinnäytetyön päätavoitteena oli selvittää, mitä hyötyjä voisi saada aikaan mallintamalla parametrisesti Grasshopper-Tekla Live-linkin avulla Tekla Structures-ohjelmassa. Erityisesti kiinnosti selvittää mitä hyötyjä mahdollisesti olisi saavutettavissa mallintamalla parametrisesti osia joilla on vaikea geometria tai osia joiden mallinnustyö

voi olla aikaavievää. Työn tavoitteet voi mieltää jokseenkin onnistuneiksi koska algoritmeja jotka helpottavat ja nopeuttavat mallinnustyötä on pystytty luomaan.

Kuitenkin pitää nostaa esiin myös ne vaikeudet joita esiintyi. Yksi tärkeimmistä on algoritmien selkeys. On erittäin tärkeätä tehdä algoritmit selkeiksi jotta niistä heti näkee kokonaisuuden.

7.2 Hyödyt

Havaittuja hyötyjä kun mallintaa Grasshopper-Tekla Live-Linkillä avulla:

- Parametrinen mallintaminen käy helposti kun parametrit ovat oikein asennettu.
- Mallintaminen koordinaateista.
- Mallintaminen osista joiden geometriat muuttuu.
- Visuaalista koodia voi käyttää uudelleen samanlaisissa tapauksissa.
- Mallintaminen käy nopeasti.
- Datan käsittely on tehokas ja looginen.

Jos mallintaa Grasshopperin kautta Teklaan on monta erilaisia hyötyjä, ehkä tärkein on muokattavuus ja Grasshopperilla voi mallintaa vapaasti miten haluaa. Mallinnustyötä Grasshopperissa voi muokata esimerkiksi ihmisien, projektien tai yritysten toiveiden mukaisesti.

7.3 Haitat

Havaittuja haittoja kun mallintaa Grasshopper-Tekla Live-Linkillä avulla:

- Kestää aikaa tehdä mallinnusosille ensimmäinen visuaalinen koodi.
- Vähän tietoa mallintamisesta Grasshopper-Tekla Live linkillä.
- Epäselvä algoritmi aiheuttaa erilaisia ongelmia.
- Kestää aikaa tottua uuteen mallinnustapaan ja ohjelmaan.

Kun ensimmäisen kerran tekee osille tai isommalle kokonaisuudelle visuaalisen koodin se voi olla työlästä ja hankalaa. Täytyy muistaa mitä halutaan savuttaa ja mikä on juuri sen osan haluttu tulos; jos ei, algoritmi voi olla epäonnistunut ja se ei toimi niin kuin pitää. Epäselvä algoritmi voi aiheuttaa sen että manuaalisesti syötettävät parametrien osat voivat hukkua algoritmissa ja algoritmin muokkausta ei voida tehdä onnistuneesti.

7.4 Haittojen oikaisu

Osan tai kokonaisuuden parametrit ovat mallinnustyön yksi tärkeimmistä ominaisuuksista. Ennen kuin tekee visuaalista koodia, täytyy määrittä parametrit ja siitä lähteä rakentamaan koodia. Parametrit pitää aina olla taustalla mielessä, koska tämä on se mikä määrittää esimerkiksi osan geometrian.

Algoritmi täytyy tehdä loogisesti ja hierarkkisesti. Grasshopperin komponentit voidaan ryhmittää riippuen tehtävistä erilaisiin ryhmiin ja näille ryhmille voi kirjata mitä mikäkin ryhmä tekee. Komponenteille voi myös kirjata selkeästi mikä komponentti tekee ja/tai mitä se tekee, koska jos toinen kuin algoritmin tekijä itse rupeaa käyttämään jonkun muun tekemää algoritmia voi epäselvyyksiä syntyä.

7.5 Loppupäätelmä ja kehitys

Parametrinen mallintaminen käyttäen algoritmeja osittain visuaalisen koodauksen avulla tulee todennäköisesti olemaan seuraava askel mallintamisessa tai suunnittelussa, rakennusalan eri suunnittelualoilla. Siirtyminen kokonaan tähän lähitulevaisuudessa on epätodennäköistä koska tämänkaltainen suunnittelu on vielä epätavallista ja liian vähän tietoa on saatavissa. Mutta pieniä parametrisiä laajennuksia kuten esimerkit tässä työssä, voidaan nykyisin jo käyttää tehostamaan mallinnustyötä.

Tässä työssä mallinnusesimerkit olivat pieniä jos ajattelee osien parametrisia suhteita toisiinsa, joten jo pienillä suhde ja riippuvuusmäärillä voidaan saavuttaa tehostettu mallinnusprosessi. Siltaesimerkki tässä työssä oli se jossa oli eniten parametrisia riippuvuuksia osien välillä, mutta kaikilla osilla oli silti ainoastaan yksi pääparametri; tielinjan koordinaatit.

Saavuttaakseen siirtymisen parametriseen mallintamiseen osien geometrian kannalta kokonaisuutena pitää pieniä parametrisisä suhteita osien välillä tehdä ja käyttää. Kun pieniä parametrisiä suhteita on pystytty luomaan varmasti ja käyttämään turvallisesti, niitä pystytään sitten yhdistämään ja suurentamaan. Näin saavutetaan suurempi parametrinen suhde isojen rakennuskokonaisuuksien välillä. Tietty ja selkeä hierarkia täytyy olla määritelty, jotta isojen kokonaisuuksien parametriset suhteet voivat onnistua.

Siirtyminen täysin parametriseen mallintamiseen käyttäen visuaalista koodia ei ole varmaan mahdollista, koska vaihtoehto geometrian manuaaliseen muutokseen on hyvä olla olemassa. Joskus manuaalinen muutos voi olla nopeampi, varsinkin jos on kyse pienistä määristä ja yksittäisistä muutoksista.

Tässä työssä tutkittiin ainoastaan parametrista suunnittelua mallinnusnäkökohdasta. Täytyy myös pohtia miten muut näkökohdat voi sisällyttää suunnitteluprosessiin parametrisessa suunnittelussa, ei ainoastaan mallinnusta. Ainoastaan mallinnuksen voi tehdä suhteellisen helpoksi parametrisesti, koska tarkoituksena on luoda suhteita ja riippuvuuksia rakennusosien geometrian ja tiedon välillä.

Jos haluaa hyödyntää parametrista suunnittelua kokonaisvaltaisesti, täytyy muut näkökohdat ottaa mukaan parametriseen suunnitteluun. Hyödyntämällä ainoastaan mallinnusta parametrisesti voidaan edesauttaa ja lyhentää mallinnusaikaa huomattavasti, mutta jos huomioidaan rakennusosien mitoitus, määräykset, vaatimukset ja samankaltaiset näkökohdat, parametrisen mallinnuksen hyödyt voivat olla vielä suuremmat.

Kokonaisuuden suunnittelu kaikista näkökulmista käyttämällä parametrista suunnittelua voi olla vaihtoehto tulevaisuudessa koska prosessit kehittyvät ajan myötä kokoajan. Esimerkki tästä voi olla parametrinen suunnittelu jossa projektin kaikki parametrit on sijoitettu yhteen tai useampaan algoritmiin jotka toimivat yhdessä.

8 Lähdeluettelo

Bachman D. 2017. *Grasshopper: Visual Scripting for Rhinoceros 3D.* United States of America: Industrial Press

Davis D. *A history of parametric.* (Online) http://www.danieldavis.com/a-history-of-parametric (Luettu: 10.1.2018)

Davis D. 2013. *Modelled on Software Engineering: Flexible Parametric Models in the Practice of Architecture.* Doctor of Philosophy. School of Architecture and Design College of Design and Social context RMIT University

Eastman C. 2011. *BIM Handbook, Second Edition.* United States of America: John Wiley & Sons, Inc.

Tedeschi A. 2014. *AAD_Algorithms-Aided Design.* Italia: Le Penseur

Woodbury R. 2010. *Elements of Parametric Design.* United States of America: Routledge

Trimble Solutions Corporation, *Grasshopper-Tekla Live Link.* (Online) https://www.teklastructures.support.tekla.com/not-version-specific/en/ext_grasshopperteklalink (Luettu: 26.12.2017)

Pöyry Oyj (Online) http://www.poyry.com/ (Luettu: 21.1.2018)

Sammanfattning på svenska

1 Inledning

I detta examensarbete undersöks metoder för att lättare kunna modellera objekt med svår geometri i Tekla Structures. Objekt som har svår geometri kan vara arbetsdryga och tidskrävande att modellera. Därför skall detta arbete ta fram alternativa sätt för att kunna modellera dessa.

Sättet som undersöks är parametrisk modellering genom visuell kodning med programmet Grasshopper. Trimble Solutions lanserade år 2017 en länk som möjliggör att modellera med hjälp av visuell kod från Grasshopper till Tekla Structures.

Sättet att modellera som undersöks är relativt nytt eftersom denna länk har endast funnits tillgänglig i ungefär ett år, därför tas det även upp i detta arbete vilken effekt detta kan ha på modelleringsjobbet samt modelleringsprocessen.

2 Parametrisk modellering

Davis (2013) beskriver en parametrisk modell som: *”Thus, a parametric model can be defined as: a set of equations that express a geometric model as explicit functions of a number of parameters”*, det vill säga att en parametrisk modell kan vara beskriven som: ett antal ekvationer som uttrycker en geometrisk modell genom tydliga funktioner av ett antal parametrar.

Parametrisk modellering kan även fungera manuellt genom att formge olika former fysiskt med fysiska parametrar, vilket gjordes före datorer fanns.

Idag har de flesta CAD och främst BIM program någon sort av parametriska möjligheter inbyggda i programmet färdigt. Dessa parametriska egenskaper möjliggör snabba förändringar för en mängd olika delar i programmet.

Som exempel kan användas armeringsverktyget i Tekla Structures. Detta verktyg kan armera olika betongkonstruktioner, där det finns färdiga manuellt styrda parametriska värden från listor programmerade in i verktyget. Parametriska värden kan vara armeringens diameter, betongskyddsskikt, stål kvalitet samt mycket mer. Förutom dessa värden bör det även väljas delen som skall armeras och armeringens slut och startpunkt. Genom detta

verktyg kan armeringen fås att följa med betongdelen vart än den flyttas eller ifall delens geometri ändras.

3 Design med algoritm

Tedeschi (2014) skriver: *”An algorithm is a procedure used to return a solution to a question – or to perform a particular task – through a finite list of basic and well defined instructions.”*, det vill säga att en algoritm är förfarande där en lösning fås till en fråga – eller att utföra en specifik uppgift – genom en ändlig lista av simpla och väl definierade instruktioner.

Modellering baserar sig generellt på virtuell modifiering med hjälp av mus och tangentbord i en digital miljö. Till skillnad från detta är algoritmisk design baserad på logik utan att använda sig av manuell modifiering. Information och data modifieras av användaren istället för digitala delar och objekt.

För att en algoritm med visuell kod skall kunna fungera bör en funktion vara väl definierad med funktionens input och output. Inputen bör vara det som funktionen kräver. Till exempel text får inte användas där funktionen behöver numeriska värden. Algoritmisk design i samband med modellering kräver åtminstone två olika jobbmiljöer:

1. Kodeditor

2. Modelleringsmiljö

Till kodeditorn kodas vad som vill visas i modelleringsmiljön. Det som är visat som kod i editorn kan till exempel vara visad som geometri i modelleringsmiljön. Koden kan bestå av text kod, visuell kod eller en kombination av dessa bägge. I detta arbete fokuseras främst på visuell kod.

4 Design med algoritm genom visuell kodning

Visuell kod består av en eller flera parametriska uppgifter, som har relationer och förhållande med parametrar, uppgifter och output. Visuell kod kan betraktas som textkodning eftersom samma mål kan uppnås med båda, men visuell kodning kan uppfattas som ett lättare sätt eftersom arbetsflödet och färdiga komponenter finns färdigt för ens behov. Detta leder till att man inte behöver någon större kunskap om kodning fast man

visuellt kodar. Förstås kan kunskap om kodning och textkod vara till stor nytta när man använder sig av visuell kod.

Med hjälp av visuell kod kan manuell modellering genom fysisk interaktion med delar i en modell delvist eller helt undvikas. Detta kan underlätta förändringar som uppstår i de modellerade delarna, eftersom parametriska relationer kan göras i den visuella koden för stora eller mindre helheter.

5 Rhinoceros 3D och Grasshopper

Rhinoceros 3D eller även känt som enbart Rhino lanserades 1998. Det är ett program som möjliggör NURBS geometrisk ytmodellering. Programmet har blivit populärt bland arkitektet och formgivare. Rhino tas enbart översiktligt fram i detta arbete på grund av att Grasshopper är en plug-in till Rhino, så för att använda Grasshopper bör man äga en lisens av Rhino.

Grasshopper är en visuellt kodningsplug-in till programmet Rhinoceros. Ursprungliga namnet för Grasshopper var Explicit History och lanserades år 2007 för Rhino 4.0.

I Grasshopper kan användaren göra en visuell kod genom att dra komponenter till en canvas och där ansluta komponenterna till varandra för att bygga upp sin parametriska visuella kod. Grasshopper kan med direkt anslutning uppdatera eller uppdateras av andra program. Anslutningar för program finns till exempel för Excel, Autodesk Revit, Tekla Structures, Archicad och många fler.

Förutom Grasshoppers egna komponenter finns det även olika plug-in som möjliggör att flera komponenter kan användas. De kändaste är Kangaroo, Lunchbox, Weaverbird och Karamba. Dessa extra komponenter kan bland annat göra simulationer och analyser för konstruktion och fysik för geometrin definierad i Grasshopper.

Den visuella koden bör vara väl strukturerad med en tydlig hierarki. När man arbetar i en stor visuell kod kan komponenter grupperas efter dess uppgifter efter en klar logik. Detta underlättar att man ser hur informationsflödet är i visuella koden. I en dåligt strukturerad visuell kod är det stor sannolikhet att koden misslyckas när förändringar behövs eller koden behöver utvigdas.

6 Grasshopper-Tekla Live Link

Grasshopper-Tekla live link möjliggör algoritmisk modellering i Tekla Structures genom att använda Rhino och Grasshopper. Länken kan i realtid uppdateras från visuell kod definierad i Grasshopper.

I Grasshopper fås Tekla komponenter i form av Tekla verktyg och komponenter. De flesta verktyg stöds av länken men armerings-, bult-, och svetsverktyget stöds ännu inte. Detta kan dock kringgås genom att använda Teklakomponenter som innehåller dessa delar.

7 Grasshopper/Tekla exempel

Med hjälp av Grasshopper-Tekla live länken har jag gjort tre olika exempel på vad som kan modelleras i Tekla:

1. Modellering av pålar baserat på avvikelseinmätningar

2. Modellering av terräng från punktmoln

3. Modellering av bro

7.1 Modellering av pålar baserat på avvikelseinmätningar

Pålinmätningar kan fås i Excel format där positions-, lutnings-, och riktningsdeviation är framställt. På grund av det tidskrävande och arbetsdryga modelleringsjobbet för dessa har jag lagt fram en visuell kod i Grasshopper som kan läsa in dessa deviationer och placera pålarna enligt pålinmätningar i Tekla.

7.2 Modellering av terräng från punktmoln

Punktmolnets punktkoordinater läses in i Grasshopper där punkterna trianguleras med hjälp av *Delaynay Triangulation* komponenten i Grasshopper. Trianglarna kan sedan modifieras efter behov. Därefter används Teklas *Slab* eller *Item* komponent för att köra ut geometrin till Tekla.

7.3 Modellering av bro

Brons väglinje kunde interpoleras av koordinater i Grasshopper. Till väglinjens ändor ansluts sedan en brodäcks profil som kan fås till en brodäcks helhet genom att använda komponenten *Sweep*. Till brodäcket ansluts sedan brons tillhörande delar och på så sätt har en viss hierarki uppstått som möjliggör att förändringar i väglinjen uppdaterar brons alla tillhörande delar.

8 Slutledning

Detta examensarbetes huvudmål var att undersöka vilka fördelar som kan uppnås genom att modellera parametriskt med Grasshopper-Tekla live länken i Tekla Structures. Speciellt för delar som har svår geometri eller delar vars modellering är arbetsdryg eller tidskrävande. Arbetets huvudmål tolkas vara uppfyllda eftersom koder kunnat skapas, som gör modelleringsjobbet i Tekla snabbare eller effektivare.

Parametrisk modellering med hjälp av algoritmer i form av visuell kod kan tänkas vara nästa delsteg inom modellering och projektering för byggnadsbranschens olika projekteringsdiscipliner. En total övergång till modellering med visuell kod är sannolikt inte ännu möjlig eftersom denna typ av modellering eller projektering ännu är relativt ny. Små parametriska expanderingar av processer kan däremot redan idag uppnås, vilket visas i modelleringsexemplen i detta arbete. Fastän dessa parametriska relationer och förhållanden var små och några, kunde en effektivare modellering göras.

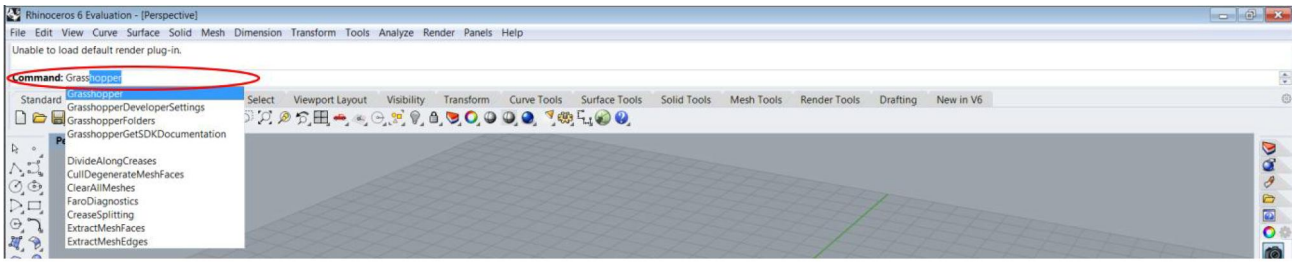
I detta arbete togs endast modelleringshänsyn i beaktan, men genom att ta andra aspekter i beaktan kan projektering möjligtvis effektiveras ännu mera. Som ett exempel på detta kan framtiden föra med sig projekt där en byggnads alla projekteringsprocesser är styrda från visuella koder som arbetar tillsammans i samtliga berörda projekteringsdiscipliner.

Liite 1

Grasshopper-Tekla live link

Asennus ja käynnistys

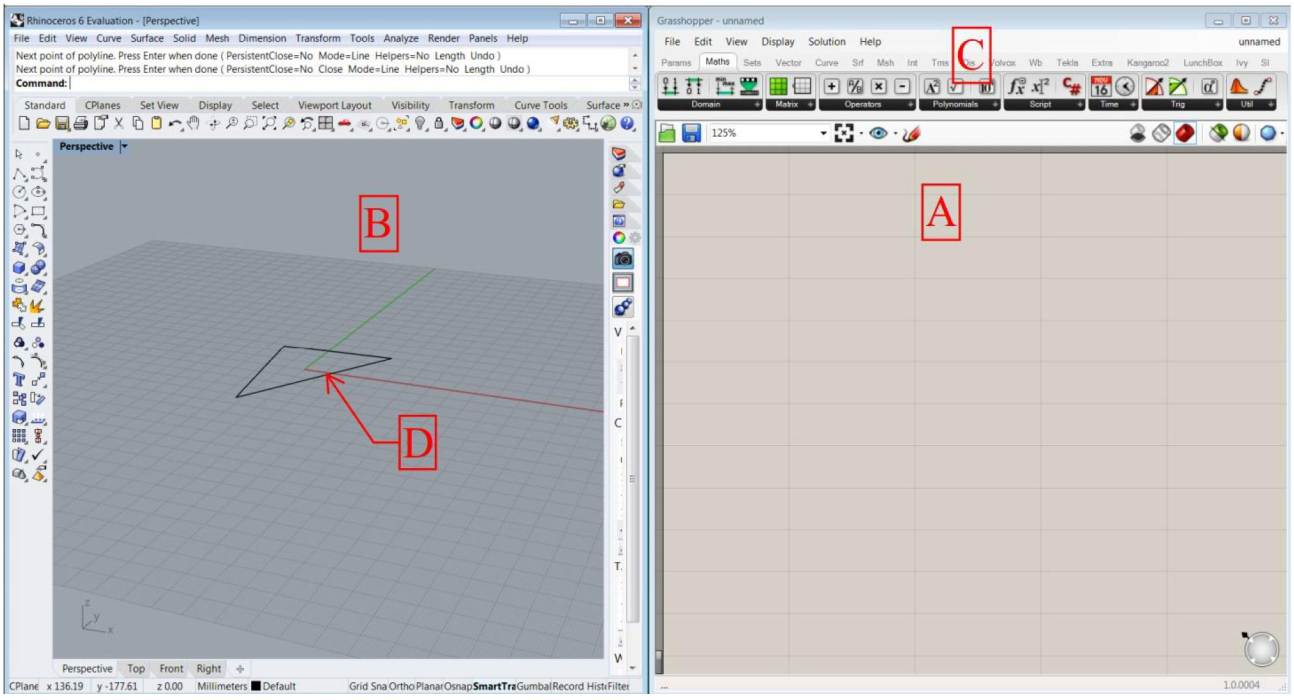
Grasshopper on plug-in Rhinoceros 3D:lle. Asennustiedosto haetaan ilmaiseksi sivustolta www.grasshopper3d.com, mutta Grasshopper vaatii Rhinoceros 3D 5.0 lisenssin tai uudempi että se toimisi. Kun Grasshopper on asennettu sen voi käynnistää kirjaamalla *Grasshopper* Rhinon komento ruutuun.



Grasshopper tutoriaaleja saa esimerkiksi www.grasshopper3d.com ja www.thinkparametric.com sivustolta.

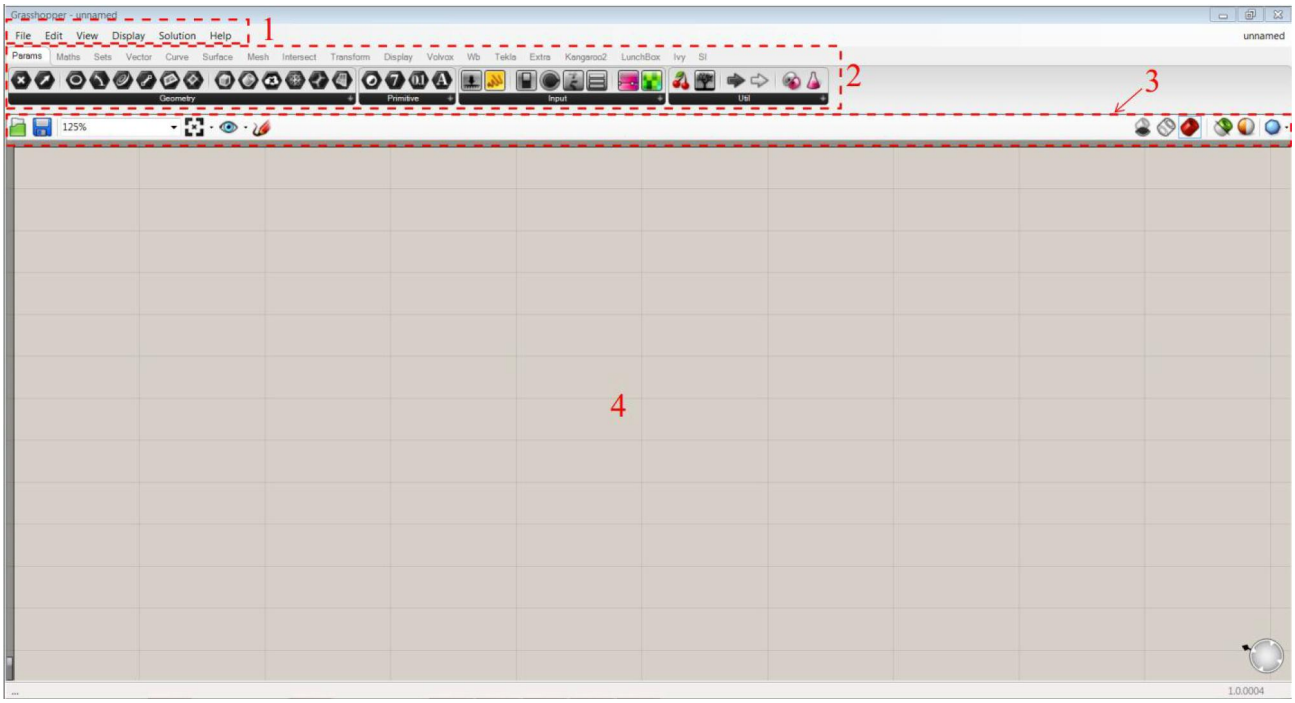
Grasshopper käyttöliittymä

Grasshopper (A) toimii rinnakkaisesti Rhino (B) mallinnusympäristöön. Editorissa voit tehdä visuaalisen koodisi kytkemällä erilaisia komponentteja (C). Kytkemällä komponentteja saadaan tehtyä parametrinen diagrammi mikä määrittää geometrian Rhinossa (D). Rhinossa voi myös tehdä geometria ja lukea sitä Grasshopperiin.



A. Grasshopper
B. Rhino
C. Grasshopper komponentteja (Parametrinen diagrammi)
D. Rhino Geometria

Grasshopper on jaettu neljään osaan:



1. Menupaneeli

Perustoimintoja kuten tallentaa/avata Grasshopper tiedostoja, y.m.

2. Komponenttipaneeli

Täältä haetaan halutut komponentit, ne saadan toimimaan vetäämälla ne canvakselle(4).

3. Canvas työkalupalkki

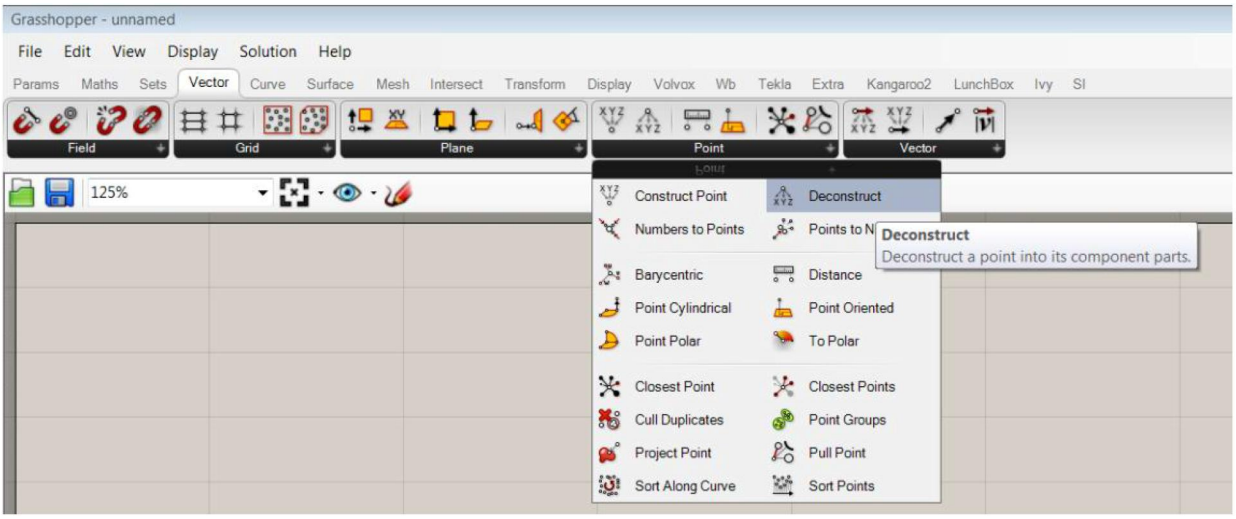
Graafisia muokkauksia Rhinolle tai Grasshopperille, ”Quick”save/load.

4. Canvas

Täällä tehdään parametriset diagrammit. Suurin osa työstä tapahtuu täällä.

Komponenttipaneeli ja canvas

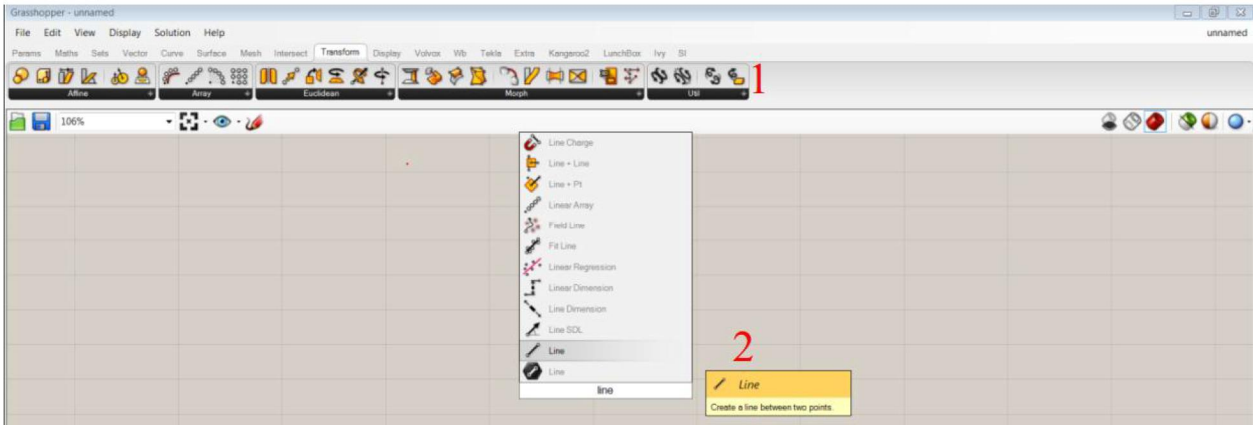
Komponentit ovat jaettu erilaisiin paneeleihin tehtävänsä mukaan. Paneeleissa on alaluokkia, riippuen siitä minkälaisia komponentteja siellä on. *Vector* paanelissa on viisi alaluokkia: Field, Grid, Plane, Point, Vector.



Alaluokat saa näkyviin klikkaamalla mustaa valikkopalkkia, ja täältä haetaan komponentti. Komponentit kytketään toisiinsa johdoilla.

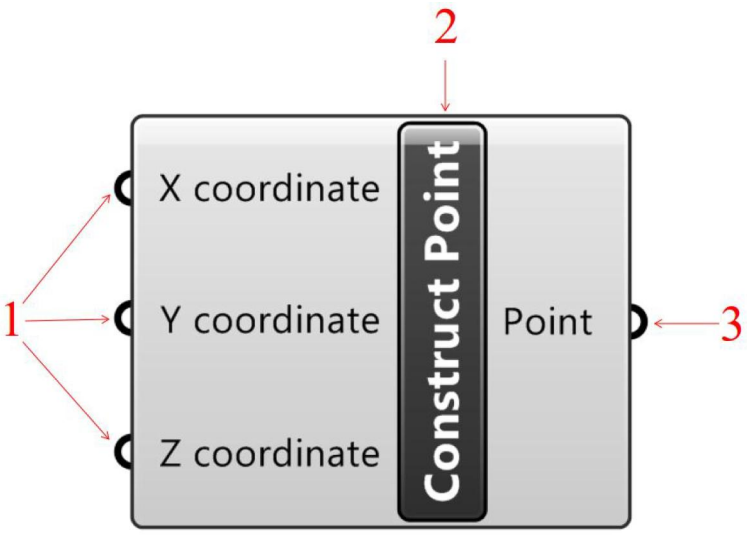
Canvas on se alue missä tehdään parametrinen diagrammi. Komponentit saadaan canvakselle kahdella tavalla:

1. Vetämällä komponenttisymboli komponenttipaneelista(1) canvakselle(2).
2. Tuplaklikkaamalla canvas(2), näin saadaan avattua työkaluluettelo. Tänne kirjoitetaan halutun komponentin nimen tai osan siitä. Grasshopper antaa ehdotuksia työkaluksi.



Komponentti

Komponentit perustuvat yleisesti kolmesta osasta: *input(1)*, *nimi(tehtävä)(2)* ja *output(3)*.



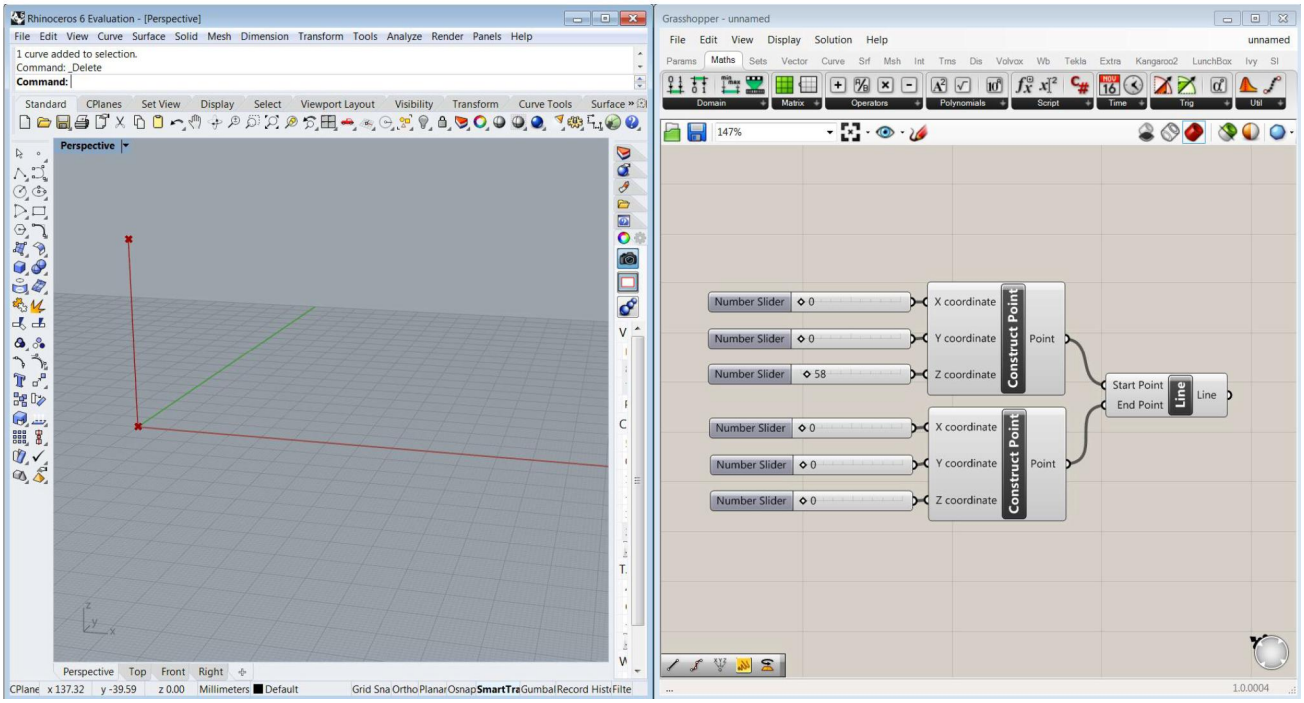
Esimerkkinä ***Construct point*** komponentti

Input osassa kytketään esimerkiksi johtoja muusta komponentista.

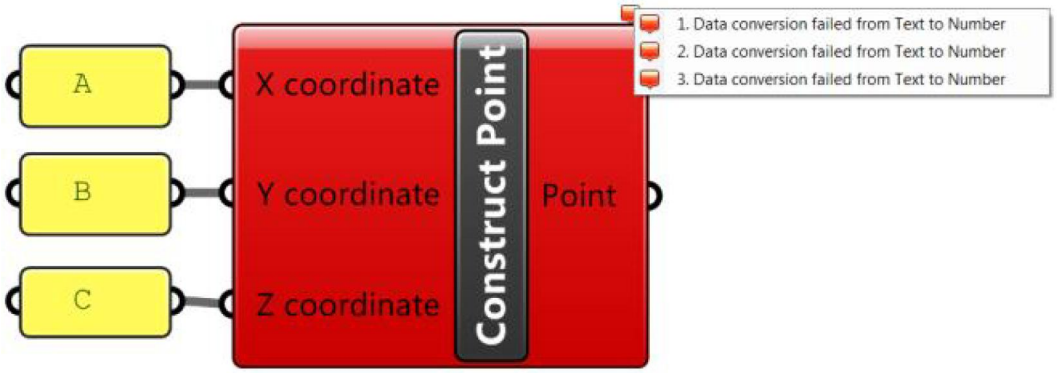
Nimi voidaan muuttaa jos halutaan tuplaklikkaamalla komponentin nimeä.

Output tekee pisteen Rhinolle.

Tämä on kaikkien komponenttien perustoiminta, eli komponentit tarvitsevat jonkinlaisen input tiedon ja siitä saa komponentin tehtävän kanssa output tuloksen.

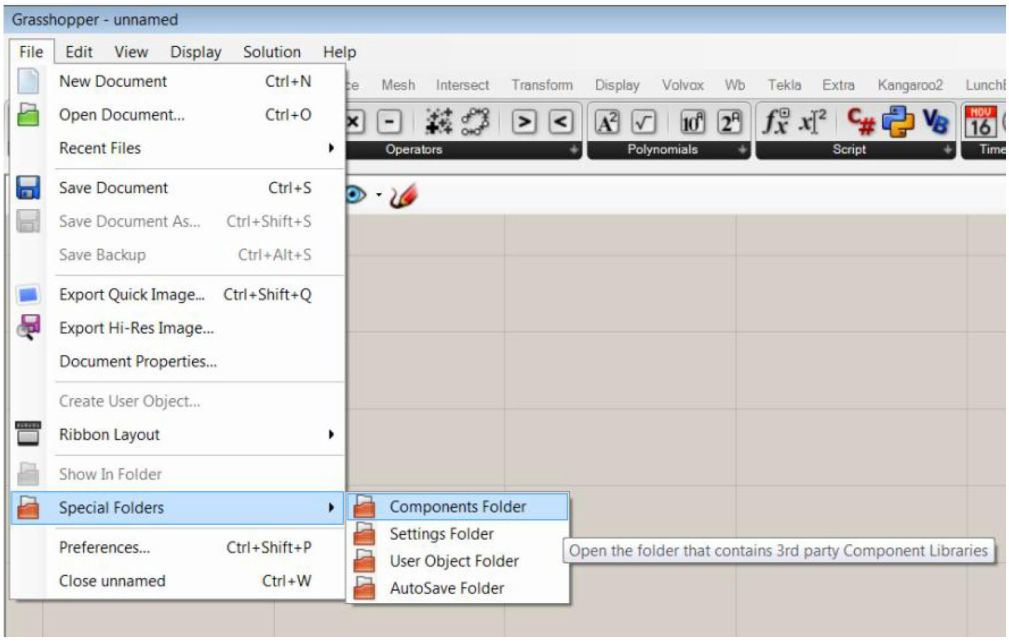


Jos komponentti jostain syystä ei toimi, Grasshopper ilmoittaa siitä näyttämällä komponentin väri oranssina tai punaisena ja selitys saadaan komponentin oikeassa yläkulmassa. Jos näin käy, on yleensä kyse että komponentin tarvittava input on vääränlainen tai puuttuu. Esimerkiksi jos komponentti tarvitsee inputina numero, sille ei voi antaa tekstiä inputina.



Käyttäjätehtyjä komponenttejä tai Grasshopper-extensionit voi saada food4rhino.com sivustolta. Sieltä saa erittäin käytettäviä komponenttejä, mikä helpottaa työtä Grasshopperissa.

Nämä komponentit saadaan toimimaan lataamalla ne tietokoneelle ja laittamalla ne Grasshopper special folderin component folderille.



10 Grasshopper-Tekla Live Link

Grasshopper-Tekla Live Linkin saa Teklan warehouseista

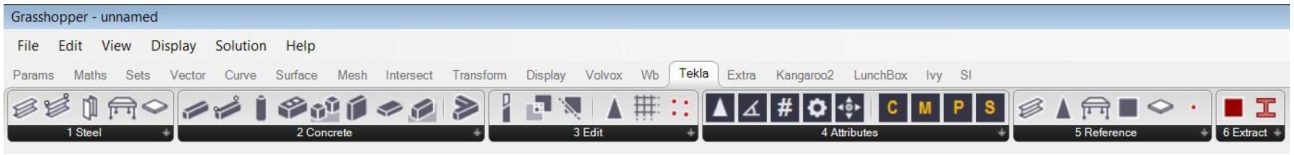
https://warehouse.tekla.com/#/catalog/details/b901f77d-cfe8-4a97-894b-f4053829c297

Teklan oma ohjeet linkistä löytyy täältä:

https://teklastructures.support.tekla.com/not-version-specific/en/ext_grasshopperteklalink

10.1 Käynnistys

- Lataa koneelle oikea Grasshopper-Tekla Live Link versio. Mikä on oikea versio, riippuu siitä mitä Tekla versiota on käytössä
- Laita tiedosto GrasshopperTeklaLink.gh Special folder à Component folder
- Käynnistä aina ensimmäiseksi Tekla ja malli kokonaan
- Käynnistä vasta sitten Rhino ja Grasshopper
- Uusi *Tekla*-niminen paneeli pitäisi nyt olla Grasshopperissa ja linkki pitäisi toimia

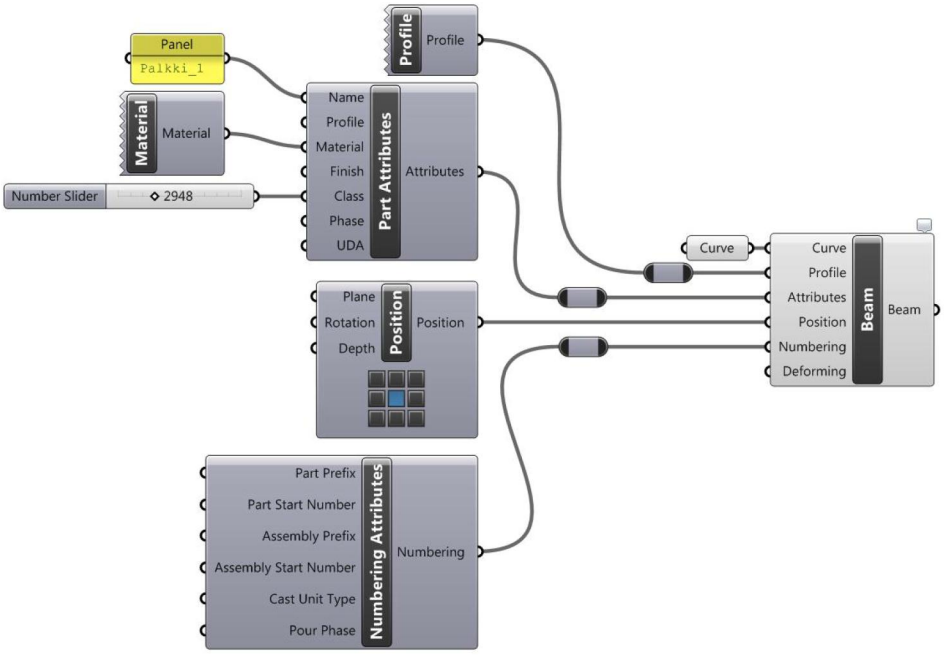


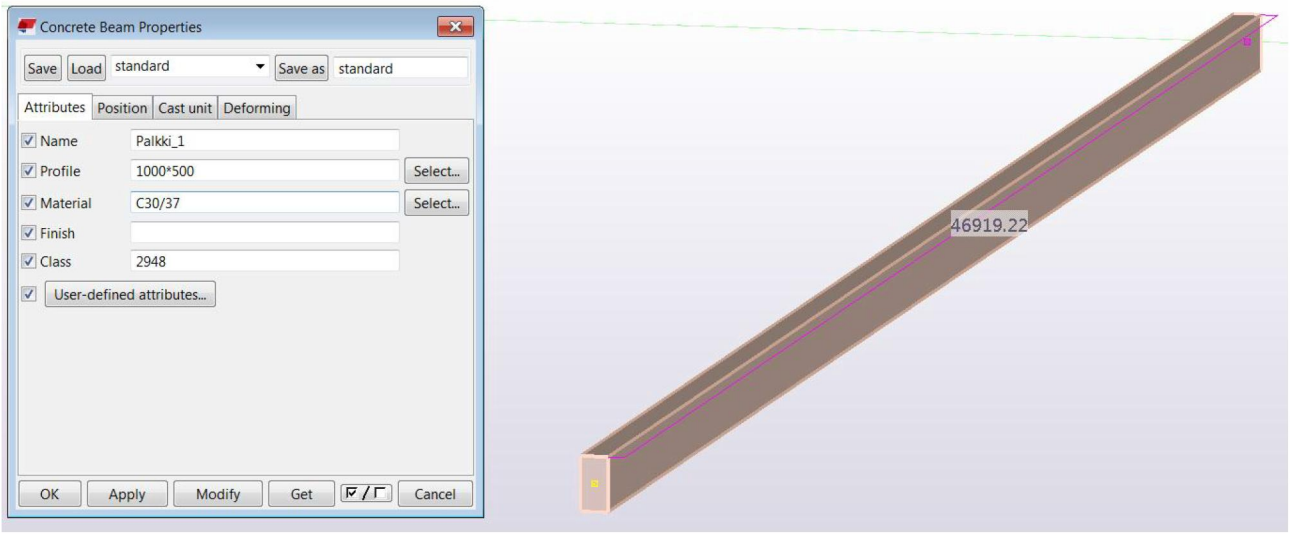
10.2 Grasshopperin-Tekla Live Linkin komponentit

Grasshopper päivittää niin kuin linkin nimi sanoo; livenä. Sen minkä teet grasshopperissa päivittyy Teklassa. Joskus päivittyminen tapahtuu pienellä viiveellä riippuen kuinka iso muutosta olet tekemässä.

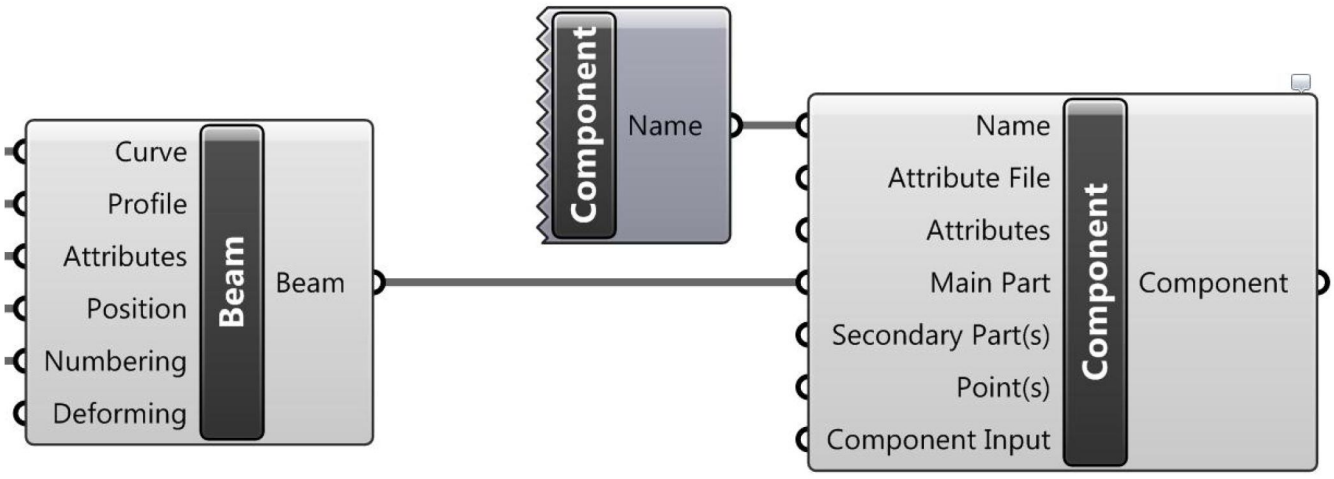
Grasshopperissa löytyy melkein kaikki komponentit, mitkä on Teklassa.

Esim. Concrete beam toimii samalla tavalla kuin tavallisesti Teklassa, siihen tarvitaan linja, profiili, attributes, y.m. Osan ominaisuudet saadaan käyttämällä sopivat Grasshopper-komponentit, kuten kuvassa on näytetty:

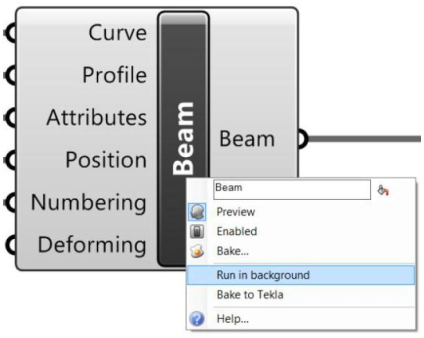




Linkki ei vielä tue Teklan raudoitus-, pultti- tai hitsityökaluja, mutta ne voi ohittaa käyttämällä esimerkiksi jonkun Teklan raudoituskomponenteista Grasshopperissa.



Mahdollistamaan output Grasshopperissa tehty komponentista, pitää poistaa *Run in background* Grasshopperissa. Näin voidaan saada tietoa esimerkiksi osan geometriasta minkä sitten voi kytkeä johonkin muuhun osaan, näin saadaan parametrisia riippuvaisuuksia eri osien kanssa.



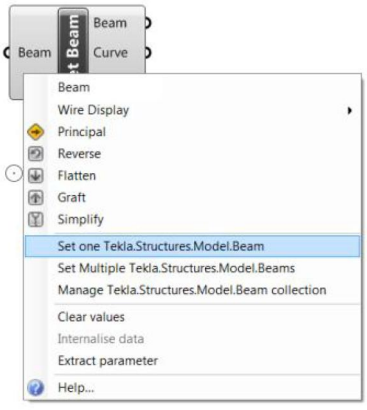
11

12 Hyvä tietää

Get Beam

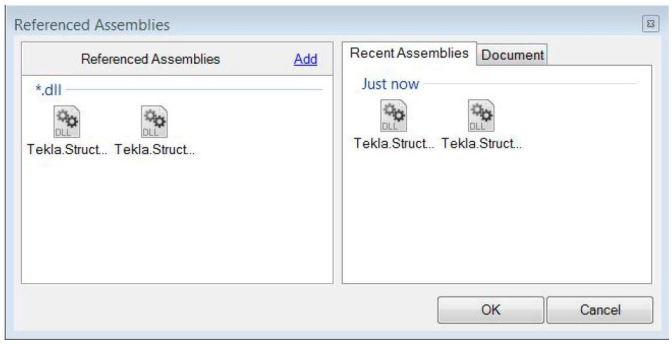
Teklassa tehtyjöä osia ei voi muokata Grasshopperissa, mutta osan tietoja voi saada esille käyttämällä Grasshopperissa Get Beam ja C# skriptaus komponentti. Get Beam komponentista saadaan helposti palkin linja ja profiili, mutta jos haluaa enemmän tietoa palkista, täytyy käyttää C# komponenttia. Tuplaklikkaamalla saadaan esille C# skripatuseditori, missä voi kirjoittaa Teklan Open API komentoja ja niin saada enemmän tietoa palkista.

1. Valitse ensiksi haluamasi palkki Teklasta.
2. Jos haluaa käyttää C#-komponentti; Valitse *Manage Asssemblies.*
3. Laita Tekla *Assemblies Reference Assemblies* puolelle.
4. Nyt *C# Script editor* voi tuoda tietoa palkista.

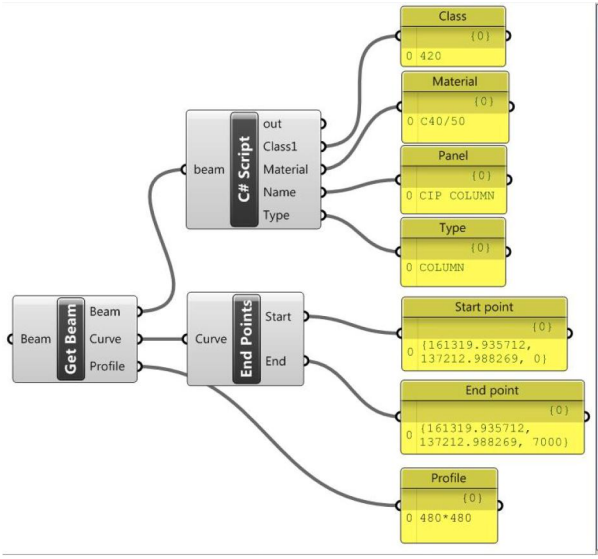


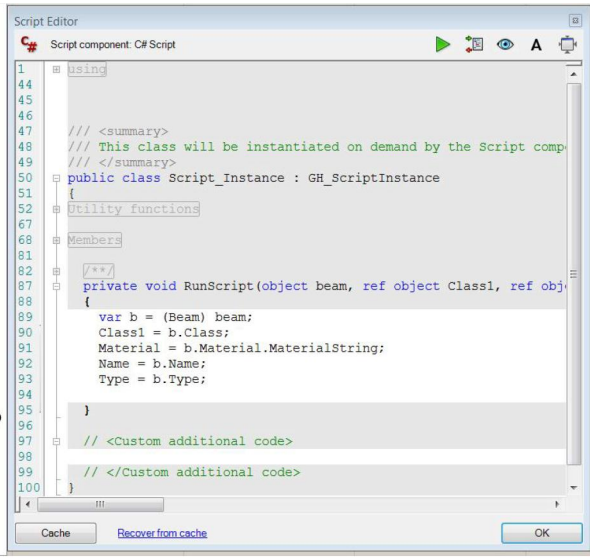
1.

2. 
x
y
out
A
C# Script
C# Script
Preview
Enabled
Bake...
Runtime warnings
Edit Source...
Remove out
Export Source...
Manage Assemblies...
Tooltip Text
Tooltip Description
Destroy Caches (16)
Variable Parameters
Help...

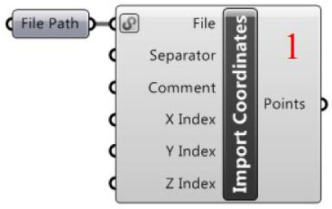
3. 
Referenced Assemblies
Referenced Assemblies
Add
*.dll
Tekla.Struct...
Tekla.Struct...
Recent Assemblies
Document
Just now
Tekla.Struct...
Tekla.Struct...
OK
Cancel

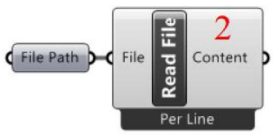
4.

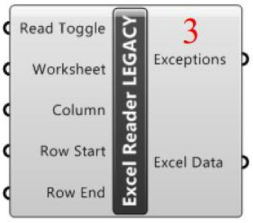




Koordinaattien tuonti Grasshopperiin







Käyttämällä *Import coordinates(1)*, *Read file(2)* tai *Excel Reader LEGACY(3)* voit tuoda koordinaatteja Grasshopperiin. Vaihtoehto 1 ja 2 pitää saada koordinaatit

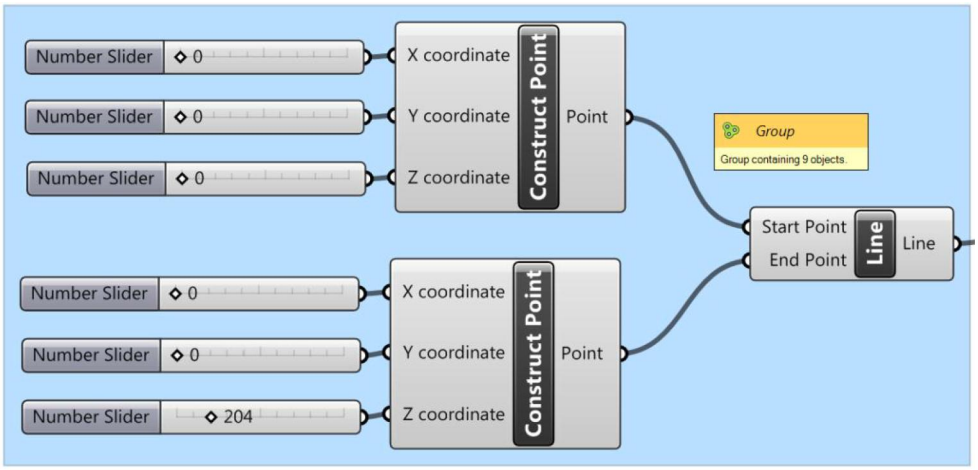
tekstimuotona, esimerkiksi notepadista. Vaihtoehto 3 voi lukea koordinaatit excel taulokoista.

Ryhmitys ja Algoritmin selkeys

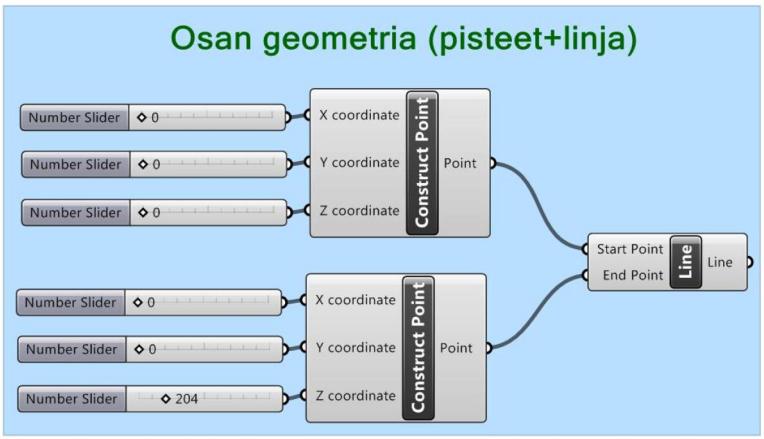
Grasshopperin työnkulku on aika vapaamuotoinen, jolloin komponentit voi laittaa miten ja minne haluaa. Tästä voi syntyä epäselvä algoritmi ja tietojen virtauksesta voi olla vaikeaa saada selvää. Tämän vuoksi kannattaa ryhmittää komponentit ja nimeä näitä ryhmiä loogisesti.

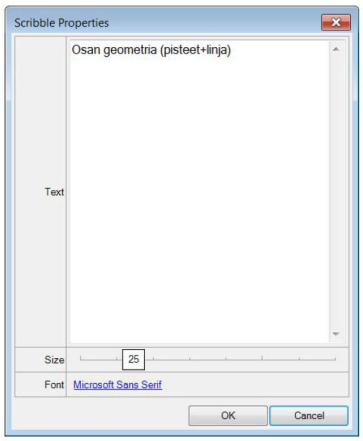
Ryhmitys komponenteista saadaan, painamalla ctrl+G kun olet valinnut halutut komponentit(1). Tämä ei vaikuttaa mihinkään, paitsi että ryhmitys saadaan visuaalisesti Grasshopperiin. Käyttäen komponentti *Scribble* saadaan myös tekstin tehty Grasshopperille(2).

1.

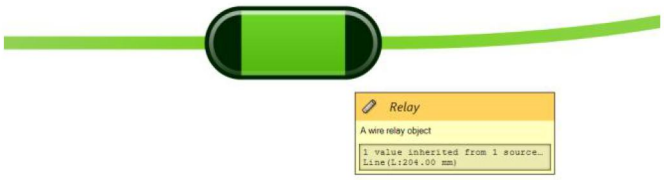


2.

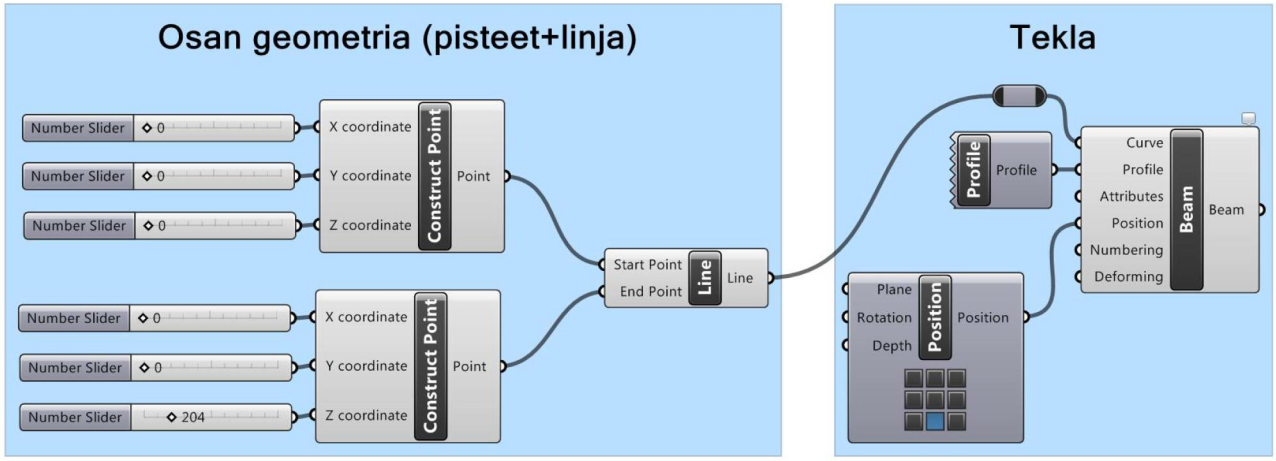




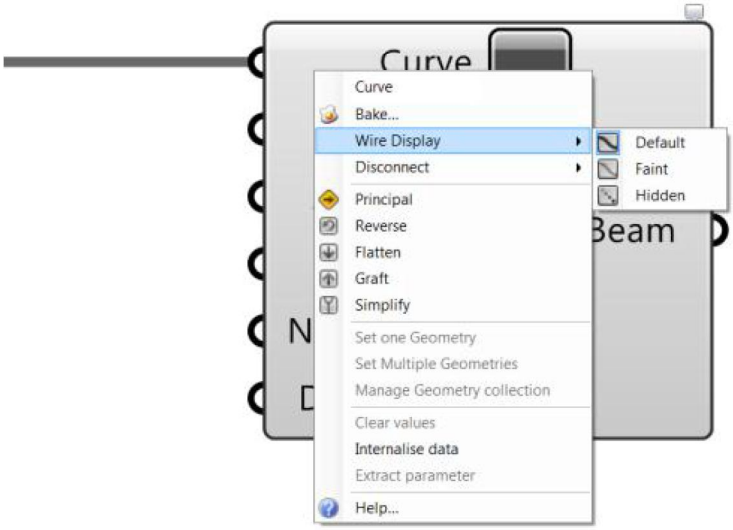
Johdoille voi käyttää *Relay*(3). Sillä voi vetää johdot paremman näköiseksi, niin että johtoja ei mene komponenttien tai muiden johtojen yli. Se saadaan tuplaklikkaamalla johtoa ja sitä voi vetää johdon mukaan minne haluaa.



3.



Johtoja voi myös saada näyttämään eri tavalla:



1. Default

2. Faint

3. Hidden

Vaikka algoritmi näyttää hyvältä kun käyttää *Hidden* johto, se voi tuottaa ongelmia koska tiedon virta ei näy selvästi.

Algoritmin selkeys on tärkeää koska algoritmi voi nopeasti muuttua erittäin epäselkeäksi kun algoritmi kasvaa isommaksi. Jos jotain menee väärin epäselkeässä algoritmissa, virheen korjaaminen voi kestää kauan ja on työlästä.