Petri Helala

SIMULAATION TUOTTAMINEN ENNALTA MÄÄRITETYN BETONIELEMENTIN VALMISTUKSESTA VISUAL COMPONENTS -OHJELMALLA

Opinnäytetyö
Tietojenkäsittely

2019

Kaakkois-Suomen ammattikorkeakoulu

Tekijä/Tekijät	Tutkinto	Aika
Petri Helala	Tradenomi (AMK)	Marraskuu 2019
Opinnäytetyön nimi Simulaation tuottaminen ennalta määritetyn betonielementin valmistuksesta Visual Components-ohjelmalla		31 sivua
Toimeksiantaja Rakennusliike U. Lipsanen Oy		
Ohjaaja Jukka Selin		

Tiivistelmä

Tämän opinnäytetyön tarkoituksena on kertoa betonielementin valmistukseen liittyvistä vaiheista Rakennusliike U. Lipsanen Oy:n toimeksiannosta ja tuottaa siitä simulaatio käyttäen valmiita 3D-malleja ja pohjapiirrosta tehtaasta. Opinnäytetyöhön sisältyy Visual Components -mallinnusohjelman käytön tutkiminen, kuvaus komponenttien mallintamisesta ja siirrosta SketchUp- ja 3ds Max -ohjelmilla. Opinnäytetyö liittyy läheisesti saamaani toimeksiantoon ja sitä kehitettäessä luotuun/luotuihin Python-ohjelmiin. Visual Components versio 4.1 Premiumia käytettiin toimivan ja helposti eri tilanteisiin muunnettavan simulaation aikaansaamiseksi, ja sen valmiiseen komponenttien kirjastoon nojattiin vahvasti opinnäytetyötä tehdessä.

Projekti alkoi siitä, että tutustuttiin Visual Componentsin perustoimintoihin. Käytettävien työkalujen tuntemus oli pakollista ennen kuin simulaatiota alettiin kehittää. Opinnäytetyön aihe liittyy työharjoitteluuni. Opinnäytetyössä dokumentoin eteen tulleet ongelmat. Ongelmista löytyy mm. kuvaus, mahdollisten ratkaisujen testaustulokset ja niistä parhaiten soveltuvan ratkaisun sisällyttäminen simulaatioon.

Opiskelussa neljän vuoden aikana oppimani tekniikat Visual Componentsia vastaavien ohjelmien käytöstä, kuten Unity ja SketchUp olivat avuksi simulaatiota tehdessä. Oliopohjaisten ohjelmointikielten osaamisesta oli hyötyä Python-ohjelmointikieltä käytettäessä. Visual Components käyttää Pythonia käyttöratkaisujen laajentamiseen tekemällä vaativammat tehdas ja myymäläratkaisut mahdollisiksi. Opinnäytetyössä kuvaan, miten näitä tekniikoita on käytetty simulaatiota tehdessä ja mitä uutta tämän prosessin aikana on niistä opittu.

Asiasanat

Visual Components, 3D-malli, Python, simulaatio, komponentti

South-Eastern Finland University of Applied Sciences

Author (authors)	Degree	Time
Petri Helala	Bachelor of Business Administration	November 2019

Thesis title

Producing a simulation of manufacturing a pre-defined concrete element with Visual Components

31 pages

Commissioned by

Rakennusliike U. Lipsanen Oy

Supervisor

Jukka Selin

Abstract

The purpose of this thesis was to describe the manufacturing process of a concrete building element at Rakennusliike U. Lipsanen Oy and to produce a simulation of it by using pre-defined 3D-models and blueprints of the factory. The thesis includes a study on the Visual Components' tools, describing the modelling and the transfer of components with SketchUp and 3ds Max programs. The thesis was closely tied to the assignment and the Python program/s made while developing it. Visual Components version 4.1 Premium was used to make a working and easily modifiable simulation. Visual Components has a library of ready-made components that were used during the making of the thesis.

The project started with getting to know the basics of the Visual Components program. Knowing how to use the tools provided was necessary before moving on to producing the simulation. My thesis is connected to my practical training. I documented every problem I encountered in the thesis. The problems have a description of them, test results of possible solutions and the implementation of the best possible solution into the simulation.

The techniques learned during the studies of the past four years in the degree programme involving the use of programs similar to Visual Components like Unity and SketchUp were helpful while making the simulation. The use of object-oriented programming languages like C Sharp also came in handy when working with the Python programming language. Visual Components uses Python to extend the use cases of Visual Components by making more complex factory and mall solutions possible. The thesis described how these techniques were used and what more was learned about them during the process.

Keywords

Visual Components, 3D-model, Python, simulation, component

SISÄLLYS

1 JOHDANTO

Tämän opinnäytetyön tarkoituksena on kertoa betonielementtien valmistukseen liittyvistä vaiheista Rakennusliike Lipsanen Oy:llä ja tuottaa siitä simulaatio käyttäen valmiita 3D-malleja ja pohjapiirrosta tehtaasta. Opinnäytetyöhön sisältyy Visual Components -mallinnusohjelman käytön tutkiminen, kuvaus komponenttien mallintamisesta ja siirrosta SketchUp- ja 3ds Max -ohjelmilla ja omien ohjelmien tuottaminen projektiin Python-ohjelmakielellä.

Opinnäytetyö liittyy saamaani toimeksiantoon ja erityisesti sitä kehitettäessä luotuihin Python-ohjelmiin. Toimivan ja helposti eri tilanteisiin muunnettavan simulaation aikaansaamiseksi käytettiin Visual Components -ohjelman versiota 4.1 Premium, jonka valmiiseen komponenttien ja toimintojen kirjastoon nojattiin vahvasti opinnäytetyötä tehdessä.

Projekti alkoi sillä, että tutustuttiin Visual Componentsin perustoimintoihin. Käytettävien työkalujen tuntemus oli pakollista ennen kuin simulaatiota alettiin kehittää. Opinnäytetyön työstäminen aikana tuli eteen monta ongelmatilannetta simulaatiota tehdessä ja kaikista eteen tulleista ongelmista on olemassa ongelman kuvaus, niiden ratkaisemiseksi tehty tutkimus, mahdollisten ratkaisujen testaustulokset ja dokumentti niistä parhaiten soveltuvan ratkaisun sisällyttämisestä simulaatioon.

Opiskelun aikana opittuja tekniikoita Visual Componentsia vastaavien ohjelmien käytöstä, kuten Unity ja SketchUp. Lisäksi oliopohjaisten ohjelmointikielten osaamisesta oli hyötyä Pythonia käytettäessä. Pythonia käytetään Visual Componentsin käyttöratkaisujen laajentamiseen tekemällä vaativammat tehdas ja myymälävaatimukset mahdollisiksi. Opinnäytetyössä kuvataan, miten näitä tekniikoita on käytetty simulaatiota tehdessä ja mitä uutta tämän prosessin aikana on niistä opittu.

Projektissa oli tärkeää tehdä lyhyen aikavälin maaleja, että projektin etenemisestä oli selkeää näyttöä. Projektin aikana käytiin useampi kokous Virrake-projektissa mukana olleiden Kaakkois-Suomen ammattikorkeakoulun henkilökunnan ja toimeksiantajien kanssa. Kokousten aikana oli mahdollisuus tuoda esille projektin edistyminen ja siihen liittyviä ongelmia, ongelmanratkontaan käytettäviä välineitä ja niin halutessa myös kuvamateriaalia työstä. Henkilökohtaisia projektin edistymiskeskusteluja käytiin useasti toimeksiantajan kanssa. Tästä oli hyötyä myös ongelmanratkaisutilanteissa. Projektiin kuului myös tutustuminen betonielementtejä valmistavaan tehtaaseen. Tutustumisen aikana oli myös mahdollisuus ottaa havainnollistavia kuvia eri prosesseista.

Visual Components on vuonna 1999 perustettu yritys ja sen 20 vuoden olemassaolon aikana on sen ympärille kehittynyt avulias verkosto asiantuntijoita, joiden aikaansaannosta on Visual Components Academy. Academyä käytettiin lähteenä useassa tilanteessa. Lisäksi hyödynnettiin Visual Componentsin hallinnoimaa keskustelupalstaa, jonka kautta saatiin nopeasti lisätietoja eri ongelmiin liittyen. Visual Components käyttää myös sosiaalisen median alustoja esitellessään eri tuotantoketjujen ratkaisuja. Yksi tällainen on heidän hallinnoimansa Youtube-kanava.

2 VISUAL COMPONENTS -OHJELMAN ESITTELY

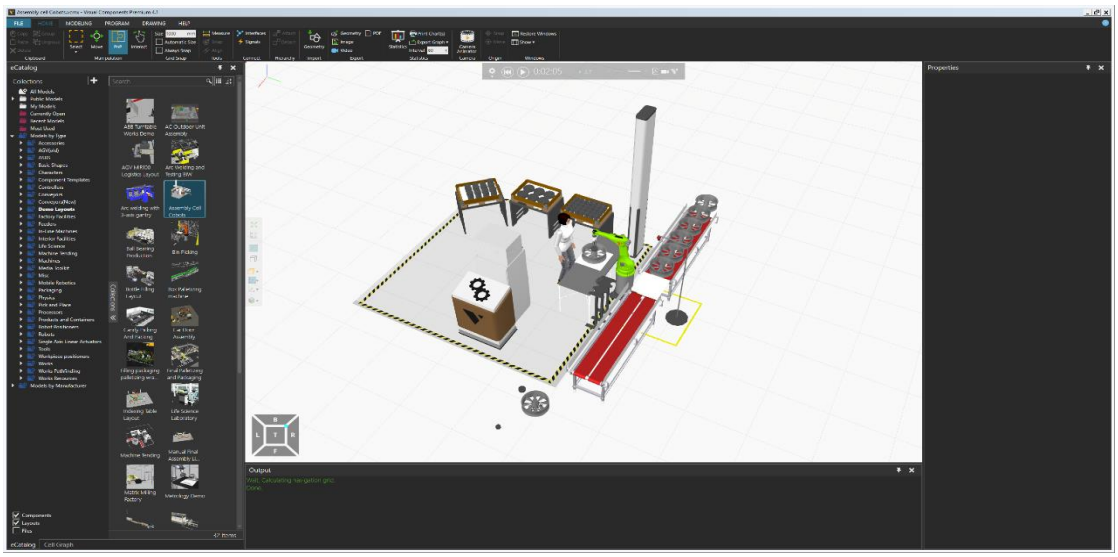
Opinnäytetyössä tuodaan esille, mitä hyötyjä on saavutettu Visual Components-, 3ds Max- ja SketchUp -ohjelmia ja Python-ohjelmointikieltä käyttämällä. Ohjelman käytön opettelun osa-alueita olivat ohjelman käyttö yksinkertaisten objektien ja simulaatioiden luomisessa ja näiden simulaatioiden yhdistäminen. Näiden lisäksi tärkeää oli ohjelman omien valmiiden toimintojen tutkiminen ja miten niitä voisi käyttää projektissa. Myös toimintojen laajentaminen kirjoittamalla omaa Python-ohjelmointikoodia oli merkittävässä osassa.

Python-ohjelmointikieltä alettiin kehittää jo 80-luvulla sen aikaisesta ABC-kielestä, mutta sen nykyisen kaltainen levinneisyys on hyvin tuore ilmiö. Ensimmäinen versio kielestä julkaistiin vuonna 1994, mutta siinä ei esiintynyt kielen tänä päivänä tarjoamia mahdollisuuksia. Vuosituhannen vaihteessa julkaistu toinen versio toi tullessaan kielelle ominaiset muistinkäsittelyyn ja merkkijono-

jen käsittelyyn liittyvät toiminnot. Syntaksia paranneltiin kielen kolmanteen versioon, jota siistittiin poistamalla jo vanhentuneita ominaisuuksia kielestä. (Kasurinen 2009, 7.)

2.1 Visual Componentsin käyttöratkaisut

Visual Components on tarkoitettu laajojen käyttöratkaisujen simuloimiseen ja sen työkalut ovat helppoja käyttää pienten demoratkaisujen esittämisessä. Omien komponenttien luominen suuria tehdasratkaisuja tehtäessä voi olla monessa tilanteessa resursseja kuluttavaa eikä aina kovin hyödyllistä ajankäyttöä. Visual Componentsissa kannattaa käyttää ohjelman omia ratkaisuja mahdollisimman paljon ja katsoa voiko ohjelman valmiit komponentit muuttaa halutun kaltaisiksi. Tiettyjen toimintojen kopioiminen jo valmiista komponentista säästää aikaa. Ohjelman peruskäytön opettelu ja komponenttien toimintojen tarkastelu antavat laajan kuvan siitä mitä mahdollisuuksia ohjelmassa on. Seuraavassa kuvassa on pieni esimerkki vanteiden tuotantoprosessin kuvaamisesta Visual Components -ohjelman avulla.



Kuva 1. Esimerkki vanteiden tuotantoprosessista

Liukuhihnat saadaan pysähtymään yhden kappaleen ollessa käsiteltävänä. Ihmiselle voidaan antaa signaali vanteen tulosta tiettyyn kohtaan liukuhihnaa. Ihminen siirtää kaksi komponenttia pöydälle erikseen, minkä jälkeen robotti saa signaalin siirtää vanne liukuhihnalta pöydälle. Ihminen saa signaalin jatkaa hakemalla uuden komponentin, joka asennetaan vanteen päälle. Robotti

siirtää viimeisen komponentin vanteen sisään ja siirtää vanteen takaisin liukuhihnalle, josta se jatkaa eteenpäin. Robotti ja ihminen jäävät odottamaan seuraavan vanteen saapumista ja prosessi jatkuu samanlaisena loputtomiin.

2.2 Polut ja ihmisten käyttö simulaatiossa

Visual Components -ohjelmaan tutustuminen aloitettiin yksinkertaisen komponentin siirto-operaatiosta tehdyn simulaation avulla. Simulaatiossa siirretään komponentti määriteltyä tietä pitkin paikasta A paikkaan B. Visual Components käyttää ihmisten, työkoneiden ja trukkien liikkumisen logiikkaan polkuja (Paths), joita pitkin voidaan siirtää eri komponentteja simulaatiossa. Visual Componentsissa ihmisille, työkoneille ja trukeille voidaan antaa lupia kulkea eri polkuja pitkin. Poluille voidaan antaa omat määritteet siitä, mitkä mobilisoidut koneet ja ketkä työntekijät saavat kulkea näillä alueilla.

Simulaatiossa pöytä tulee siirtymään polkua pitkin tehtaan toisesta päästä kiertäen tehtaan ennen kuin se päätyy varastoistavaksi. Pöydälle asennetaan muotti yhdessä työpisteessä, josta se muotinasennuksen jälkeen siirretään raudoitusalueelle. Raudoituksen jälkeen pöytä jatkaa betonivalupaikalle, josta se jatkaa uuniin. Betonielementti viettää uunissa tietyn ajan, josta se viedään pesuun ja viimeiseksi varastoidaan. Visual Componentsista löytyy valmiita ratkaisuja, joissa on käytetty koneita, ihmisiä ja liukuhihnoja, joita voidaan käyttää hyväksi omissa simulaatioissa.

Visual Componentsin eCatalog-komponenttikirjastossa on useita komponentteja, joille on valmiiksi annetut logiikat komponenttien kuljettamisessa. Esimerkkinä toimii liukuhihnat, joissa on valmiiksi määritellyt nopeudet, kulkusuunnat ja mahdollisuudet liittää liukuhihnan päät muihin saman siirtologiikan omaaviin komponentteihin. Tämä onnistuu yksinkertaisesti viemällä komponentit lähelle toisiaan, jolloin ohjelma automaattisesti yhdistää ne toisiinsa. (Simulate a process with Human, Machine and Robot 2017).

Visual Components käyttää hyväksi Works TaskControl -komponentteja, jotka hoitavat Visual Componentsin eCatalogissa olevien komponenttien logiikat. Works TaskControl -komponentista pitää löytyä yksi instanssi simulaation si-

sällä, jotta komponenttien logiikat voivat toimia. Polkujen käyttöön liittyen tarvitaan myös komponentti Works Resource Pathfinder, jolla hoidetaan ihmisten, työkoneiden ja trukkien liikkuminen. (Works Library – Basics of Task Creation 2017).

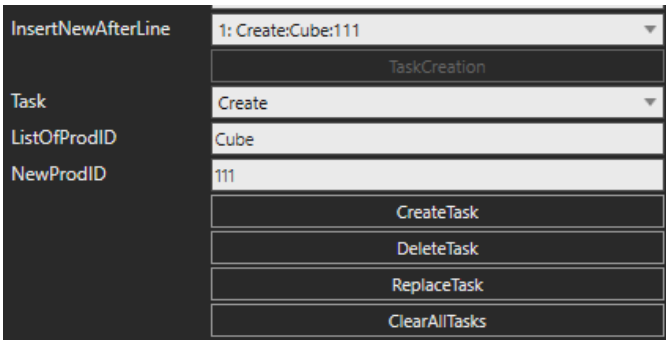
Valmiissa harjoitussimulaatiossa ihminen siirtää määrätyllä nopeudella komponentin Works Process -komponentilta toiselle ja jatkaa tämän tehtävän suorittamista niin kauan, kun simulaation annetaan käydä.

Kuva 2. Simulaation luominen Visual Components -ohjelmalla valmiin 3D-mallin pohjalta

Esimerkin laatikonhaku-simulaation kaltaista komponentin siirto-operaatiota tullaan käyttämään lopullisessa simulaatiossa betonielementin valmistuksessa, muotin osien haussa ja betonielementin rautavahvikkeiden rakentamisessa. Simulaatiossa Works Process -komponentille, jollaiselta sininen kuutio haetaan ja jollaiselle se myös viedään, annetaan tehtäviä (Task).

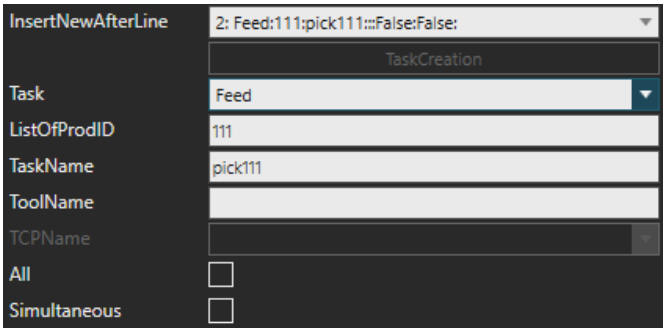
Ensimmäiselle Works Process -komponentille on annettu kaksi tehtävää, ”Create” ja ”Feed”. Create-tehtävässä simulaatioon tuodun sinisen kuution Visual Componentsin komponenttikirjastossa annettu tuotenimi ”Cube” annetaan Works Process -komponentin ListOfProdID-kohtaan tuotetiedoksi. Tälle

kuutiokomponentille kerrotaan myös sopiva uusi ID ja tällä ID:llä Works Process -komponentti tulee luomaan uusia kuutiokomponentteja.



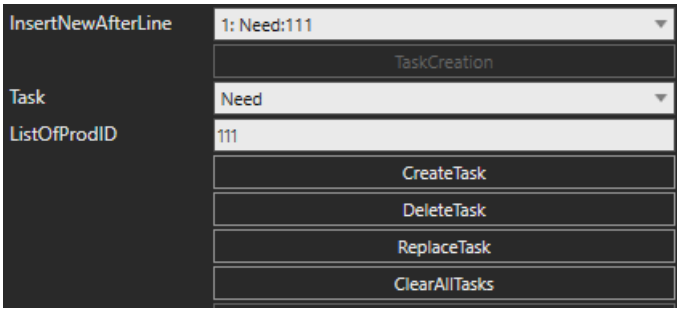
Kuva 3. Works Process -komponentti ja sen tehtävät (vrt. oman simulaation 1. vaihe)

Esimerkkisimulaatiossa ID on ”111”. Feed-tehtävässä tehtävän nimeksi annetaan Create-tehtävässä määritetyn tuoteID:tä mukaileva nimi ”pick111”. Samaan työtehtävään liittyvät nimet komponenteille ja tehtäville kannattaa pitää samankaltaisina väärinkäsitysten välttämiseksi.



Kuva 4. Works Process -komponentti ja sen tehtävät (vrt. oman simulaation 2. vaihe)

Toiselle Works Process -komponentille annetaan vain yksi tehtävä (”Need”). Tälle tehtävälle annetaan Create-tehtävässä määritetty tuoteID ”111” ja luodaan tehtävä CreateTask-painiketta painamalla, kuten muidenkin tehtävien kohdalla. Tässä vaiheessa simulaatioon tuodaan valmiilla logiikalla toimiva Works Human Resource. Tälle resurssille annetaan Tasklist-kohtaan tehtävä pick111 ja varmistetaan, että ”Pathfinding” on ruksattuna ”Optimization”-välilehdellä.

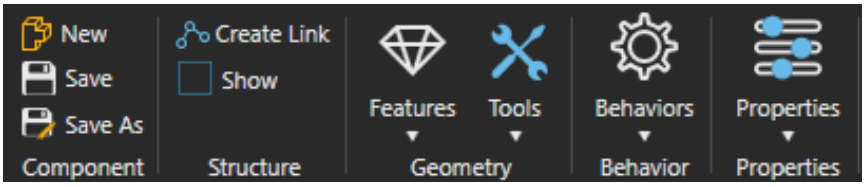


Kuva 5. Works Process -komponentti ja sen tehtävät (vrt. oman simulaation 3. vaihe)

Projektissa haluttiin simulaatioon oikeilla mittasuhteilla määritelty pöytä, jolle kasataan asteittain muotti tietyssä kohtaa simulaatiota. Pöytä liikkuu eteenpäin tehtaan lattiaan upotetuilla rengaskuljettimilla, joiden päällä työntekijät siirtävät pöydän manuaaliohjaimella muotinasennus paikalle. Muotti kasataan asteittain yhden tai useamman työntekijän voimin. Muotin kasattuaan työntekijät siirtävät pöytää eteenpäin raudoituspuolelle, jossa betonin rautavahvikkeet asennetaan paikoilleen ennen betonivalua. Raudoituksen suorittamisen jälkeen muotti siirretään betonivalualueelle, suoritetaan valu ja siirretään muotti uuniin, joka sijaitsee tehtaan toisessa päässä. Valmis elementti pestään ja varastoidaan odottamaan kuljetusta.

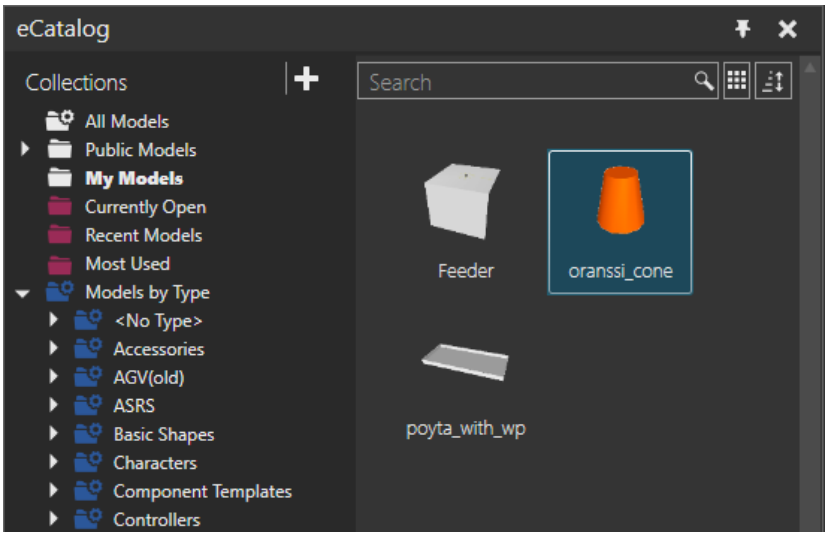
2.3 Komponenttien luonti

Omien komponenttien luominen Visual Components -ohjelmalla tapahtuu tuomalla yksinkertaisia kolmiulotteisia kappaleita maailmaan. Kappaleet ovat määritelty tietyn kokoisiksi ja valittavilta muodoiltaan Visual Componentsin oletuskappaleiden mallisiksi. Komponentti luodaan käyttämällä Visual Componentsin Modeling-ympäristöä, josta löytyy komponentteihin liittyvästä osiosta valinta ”New”. Tällä tavalla komponentille luodaan maailmaan koordinaatti, johon geometriaa voidaan rakentaa. Tällaisia yksinkertaisia geometrioita kutsutaan ohjelmassa ”Primitive Geometry” ja ne löytyvät Features-valikon kautta.



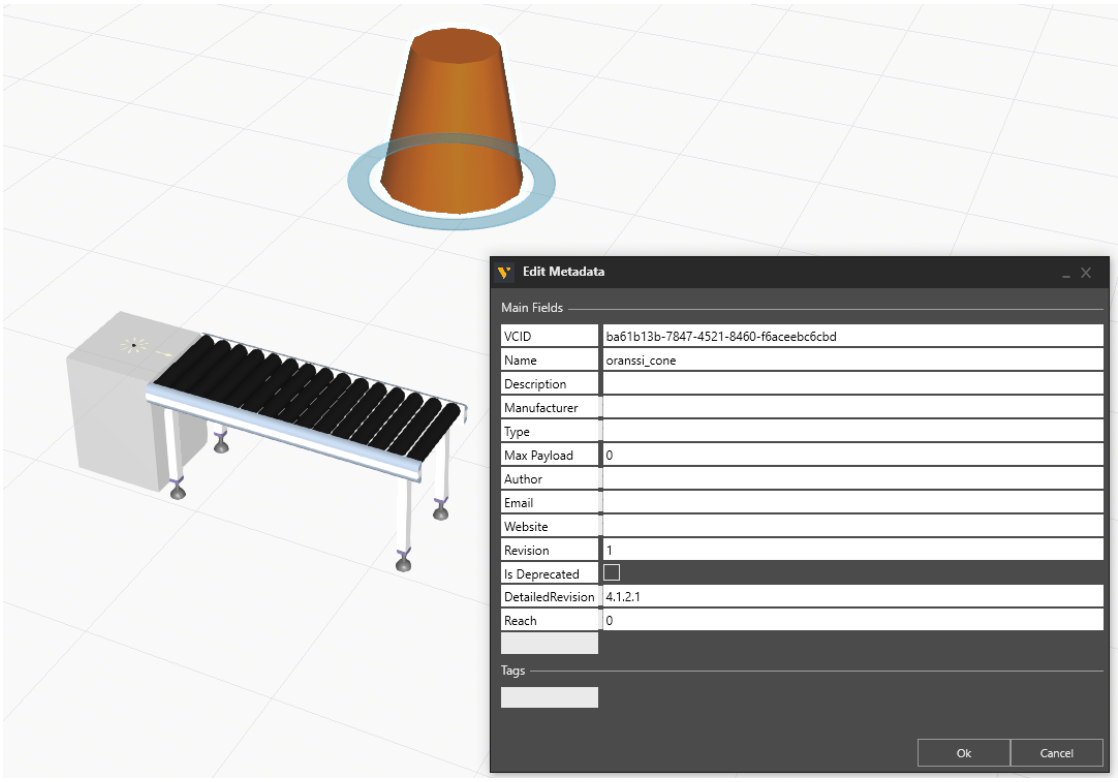
Kuva 6. Visual Components -ohjelman työkaluvalikko

Näitä valmiita kappaleita ovat esimerkiksi laatikko, sylinteri, pallo ja alusta. Tuonnin jälkeen niiden määrityksiä pystyttiin muuttamaan komponenttiin sisällytetyn logiikan mukaisesti. Komponentin voi nimetä ja sen kokoa pystytään muuttamaan kaikkien akseleiden suhteen. Tämä tapahtuu komponentin ominaisuuksista.



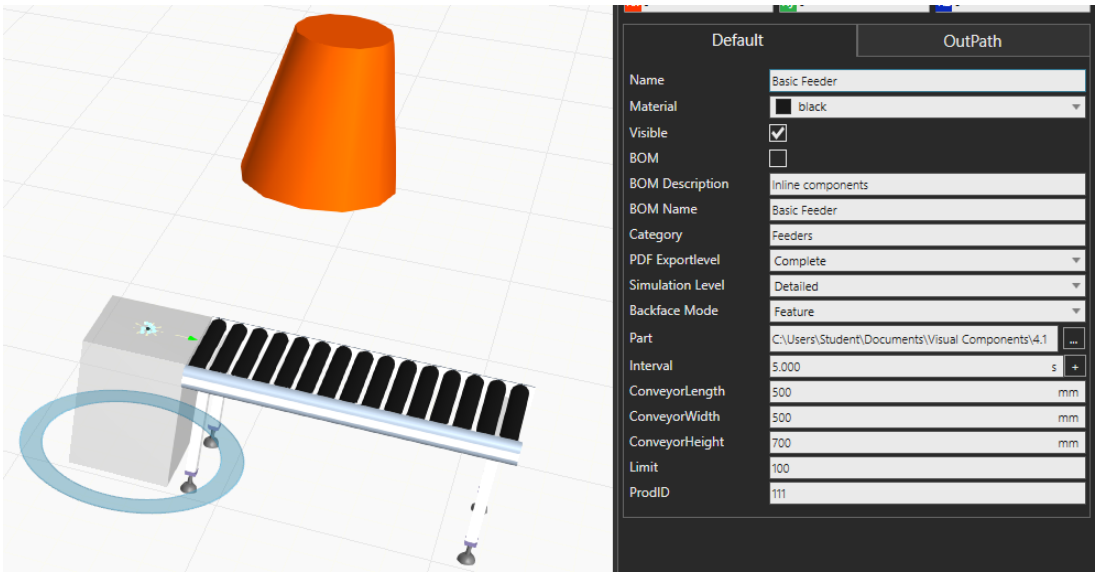
Kuva 7. Omia komponentteja komponenttikirjastossa

Seuraava kuva havainnollistaa, kuinka Visual Componentsin eCatalogista tuodaan maailmaan basic feeder -komponentti. Tämä on yksi monista mahdollisista liukuhihnalogiikkaan liittyvistä komponenteista. Lisäksi kuvassa näky luotuna oranssin värinen kartio ja siihen liittyvät metatiedot.



Kuva 8. Komponentti metatietoineen

Metadatasta on hyötyä, kun halutaan luoda feederillä komponentteja, jotka ovat valmiina eCatalog-kirjastossa, tai kun luodaan itse komponentteja, jotka on tallennettu omiin tiedostoihinsa. Komponentin luominen feederillä vaatii komponentin VCID:n (Visual Components Identification) kirjainnumerosarjan.



Kuva 9. Valitun feederin ominaisuudet, joita voidaan katsoa ja muuttaa

Komponentin VCID:n voi kopioida feederin ominaisuuksista löytyvään Part-osioon kirjoittamalla kenttään ensin ”vcid:” ja liittämällä kirjainnumerosarjan sen perään. Visual Components löytää tunnisteen avulla komponentin tietokoneen tiedostoista ja tulostaa Part-kenttään komponentin tiedostopolun. Halutun komponentin voi hakea myös käyttämällä resurssienhallintaa, joka avautuu Part-kentän oikeassa laidassa olevasta ”…”-painikkeesta.

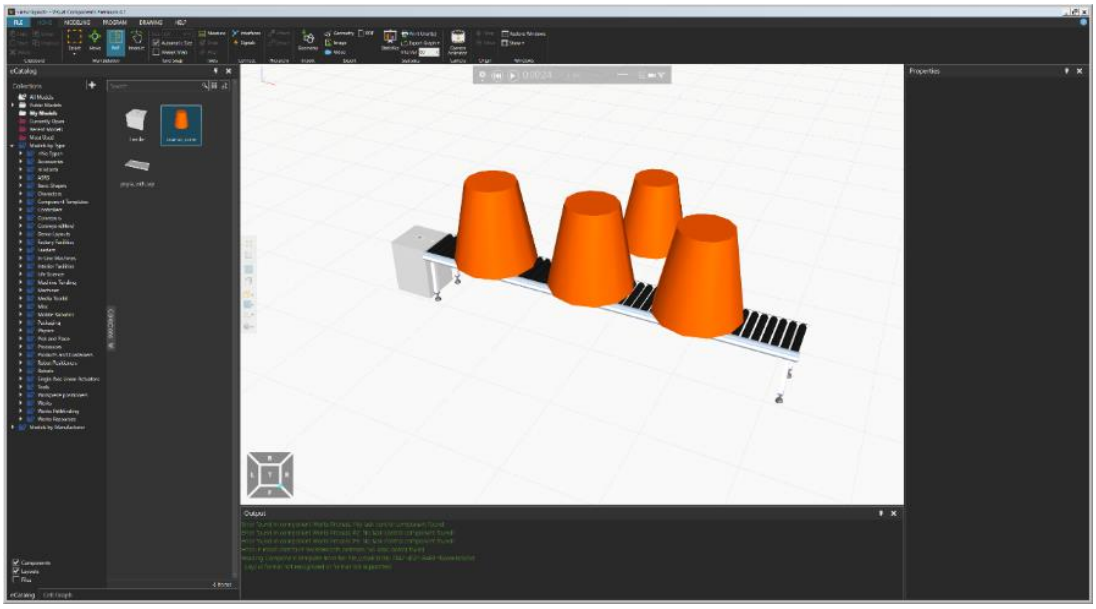
Projekti vaati sen mahdollisuuden, että uusia komponentteja pystyttiin luomaan samalla kun simulaatio oli käynnissä ja näiden komponenttien luomisen aikaväliä piti voida muuttaa. Aikaväliä (Visual Componentsissa nimellä Interval) voitiin muuttaa suoraan komponenttien luomiseen tarkoitetuilla ohjelman omilla komponenteilla, syöttäjillä eli feedereillä. Näillä komponenteilla on valmiit ominaisuudet komponentin luomisen aikavälin muuttamiseen ja niille saa annettua omaa logiikkaa käyttämällä Python-ohjelmointikieltä.

Komponentteja pystytään luomaan maailmaan edellä mainitulla tavalla. Lisäksi Visual Componentsissa on myös oma laaja eCatalog-kirjasto, joka sisältää valmiita komponentteja. Kirjasto on verkkoon yhteydessä ja tekee säännöllisiä verkkokirjaston tarkistuksia hakiessaan mahdollisia verkkoon lisättyjä komponentteja. (Discover what you can do with the Visual Components eCatalog, 2017).

ECatalogin kautta saadaan helposti esimerkkikomponentteja esimerkiksi Visual Componentsin koordinaatiston käytöstä, kappaleiden kulkusuunnista maailmassa ja eri komponenttien liittämisestä toisiinsa.

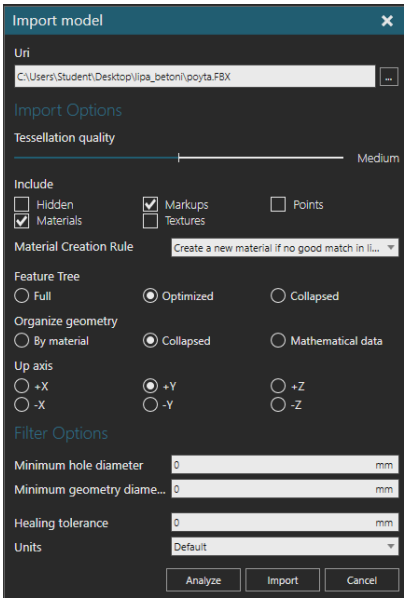
2.4 Komponenttien tuonti muista ohjelmista

Mittasuhteiden merkitys komponenttien tuonnissa muista ohjelmista on suuri. Tämän havainnollistamiseksi kuvassa olevasta kartiokomponentista on tehty huomattavasti suurempi kuin liukuhihna antaisi myöden.



Kuva 10. Komponenttien tuonti Visual Componentsiin

Visual Componentsissa on huomattava määrä komponenttien tuontiin liittyviä ominaisuuksia, jotka ovat tähänkin simulaatioon liittyvän pöydän kannalta tärkeitä. Ohjelmassa on myös tuontiasetus komponentin pinnanmuotojen säilyttämiseen liittyen.

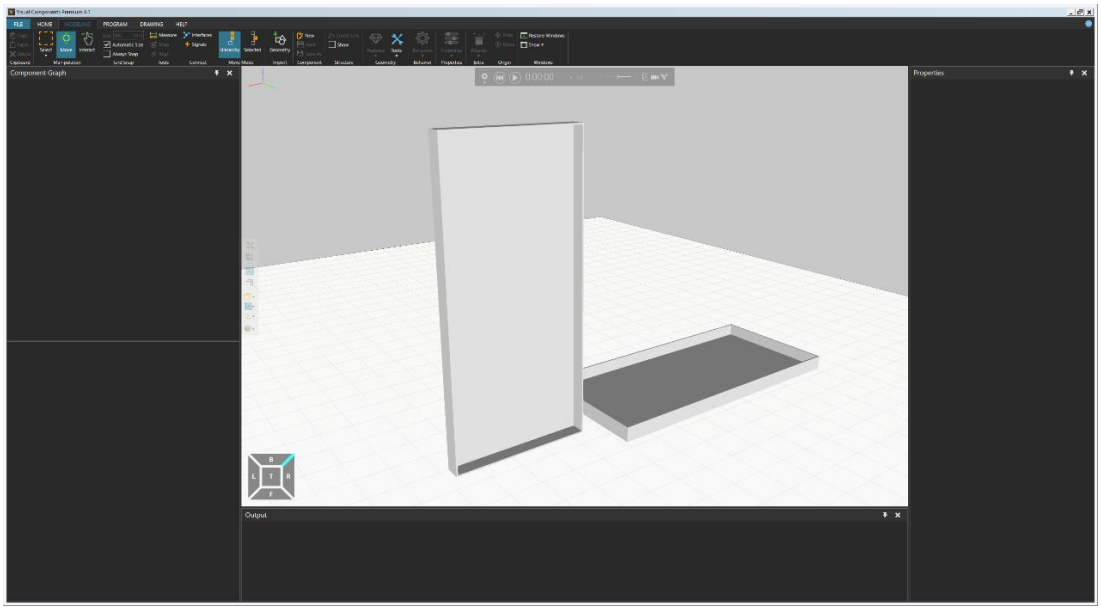


Kuva 11. Komponentin tuontiasetukset

Tesselöintiasetuksen avulla saadaan komponentin alkuperäistä kolmioprimitiivien lukemaa (polygoni) pienennettyä näytönohjainta ja prosessoria vähemmän kuormittavaksi. Sen avulla voidaan vaadittaessa myös nostaa polygonien määrää suuremmaksi.

Visual Componentsilla voidaan tukea myös tuotavien komponenttien alkuperäistä värimaailmaa käyttämällä Materials-valintaa tai se voidaan ottaa pois päältä, jolloin ohjelma luo komponentille Visual Componentsin oman oletusvärimaailmaan sopivat pelkistetyt sävyt. Väreistä ja tekstuureista pitää löytyä alkuperäistä hyvin vastaava versio ohjelman omasta kirjastosta tai ohjelma automaattisesti luo omat, ohjelmaa vähiten kuormittavat värit ja tekstuurit tuoduille komponenteille.

Visual Components saadaan myös luomaan tuotava komponentti tietyn akselin mukaan sekä positiivisella että negatiivisella astekulmalla akseliin nähden. Esimerkkinä toimii pöydän tuominen ohjelmaan Y-akselia noudattaen positiivisella kulmalla valitsemalla tämä vaihtoehto tuontiasetuksista ja painamalla Import-painiketta.



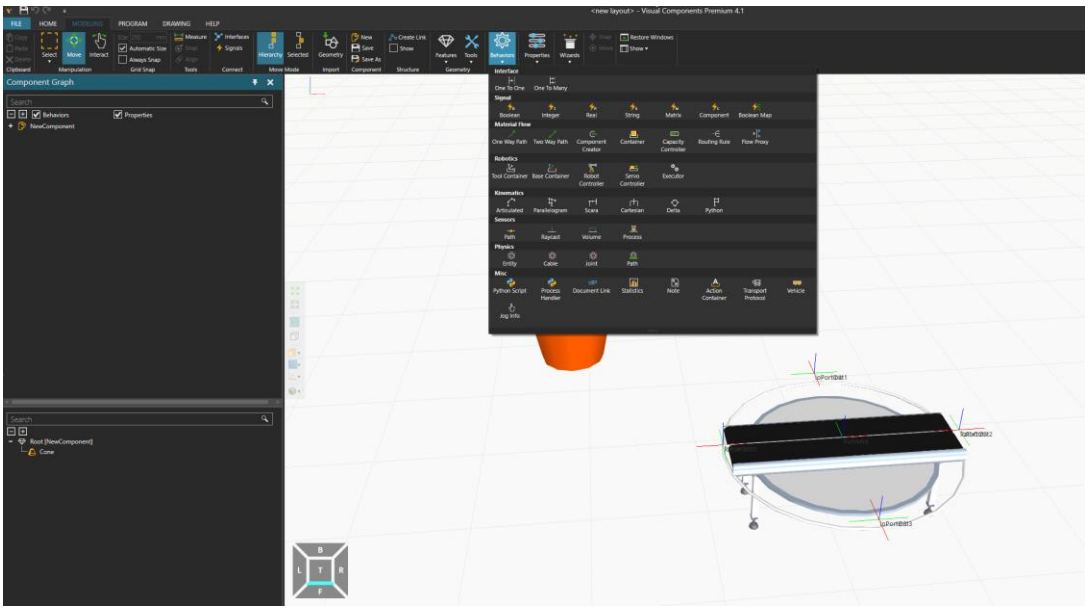
Kuva 12. Saman komponenttien tuonti eri akseleille

Pöydästä on tuotu kaksi instanssia Visual Componentsiin. Toinen positiiviselle Y-akselille ja toinen negatiiviselle X-akselille. Y-akseli on pystysuorassa ja X-akseli vaakasuorassa. Projektissa käytetään maailmassa valmiiksi esiintyvää pöytää alkuperäisenä mallina niille pöydille, joita feederit tulevat luomaan simulaatiossa. Tuodun pöydän mittasuhteet, orientaatio ja käytetty akseli tulevat olla oikeinmääriteltyjä jo pöydän ensimmäisestä ohjelmaan tuonnista lähtien.

Visual Components antaa Python-lisäosien avulla mahdollisuuksia tuoda komponentteja maailmaan myös ennalta määrittämättömällä tavalla, mikä ei ole tämän projektin luonteen kannalta suotavaa. Python-koodilla voidaan myös tuoda komponentteja maailmaan eri aikavälein ja resursseja ohjata niin, että ne käyttäytyvät mahdollisimman paljon kuin oikeat työntekijät. Esimerkkinä voi olla pöydän muotin valmistuminen nopeammin kuin edellisen, jolloin voidaan olettaa, että tämän pöydän ja sen kuljettajan pitää odottaa edellä olevan pöydän kuljetusta eteenpäin, ennen kuin se pääsee esimerkiksi raudoitusalueelle.

2.5 Käyttäytymismallit ja niiden hyödyt

Python-ohjelmakielen käyttö Visual Componentsilla on verrattavissa pitkälti muiden kolmiulotteisten maailmojen luomiseen kehitettyjen ohjelmien vastaaviin laajennuksiin. Oliokielenä Python toimii Visual Componentsin oman komponenttikirjaston ja siihen sisällytettyjen komponenttien logiikan rajapintana niin pitkälle, että kaikkia komponentteja pystytään referoimaan ja yksinkertaisilla lisäyksillä Visual Componentsin valmiiden lisäosien kanssa saadaan myös suurien tehtaiden ja kauppakeskusten logiikat simuloitua. Visual Componentsilla on Python-ohjelmakoodin ja ohjelman välisestä rajapinnasta kuvaus mahdollisista toiminnoista jokaiseen komponenttityyppiin ja käyttäytymismalliin liittyen.



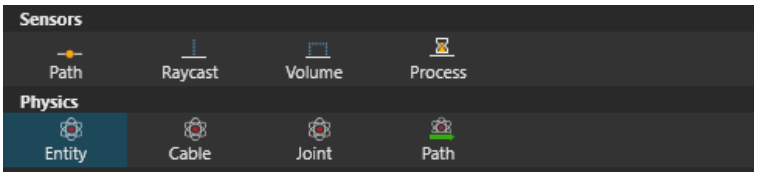
Kuva 13. Erilaisten käyttäytymismallien esittelyä

Käyttäytymismallityyppejä on tarjolla runsaasti, mutta projektiin sopivia tyyppejä näistä ovat Interface, Signal, Material Flow, Physics, Sensors ja muiden käyttäytymismallien alle lukeutuva oman koodin lisäämisen mahdollistava PythonScript. Käyttäytymismalleja luodaan komponenteille käyttäytymiset-valikon alta. Interface-käyttäytymistyyppiä sovelletaan kahden tai useamman komponentin välisessä logiikassa, kuten komponentin siirtoon toiselta komponentilta toiselle liukuhihnoja pitkin.

Signaalit toimivat tiedonvälittäjinä Python-ohjelmakoodin ja komponentin välillä. Niiden avulla voidaan antaa esimerkiksi signaali robotille tai ihmiselle tehtävän aloittamiseksi. Sensoreita käytetään, kun halutaan seurata komponentin liikettä maailmassa ja niitä hyödynnetään usein yhdessä signaalien kanssa. Material Flow -käyttäytymismallit ovat valmiita komponentin siirtoon määritettyjä logiikoita polkujen rakentamista varten.

2.6 Ohjelman fysiikanmallinnuksen mahdollisuudet

Fysiikkaa voidaan mallintaa ohjelmassa monin eri tavoin, komponenttien painosta ja liikkumissuunnasta lähtien. Siihen liittyviä käyttäytymisiä ovat painovoimaa ja liike-energiaa mallintava Entity-käyttäytyminen, kaapelin fysiikkaa simuloiva Cable-käyttäytyminen, komponenttien liitoskohtien fysiikkaa mallintava Joint-käyttäytyminen ja polkujen fysiikanmallinnuksen mahdollistava Path-käyttäytyminen.

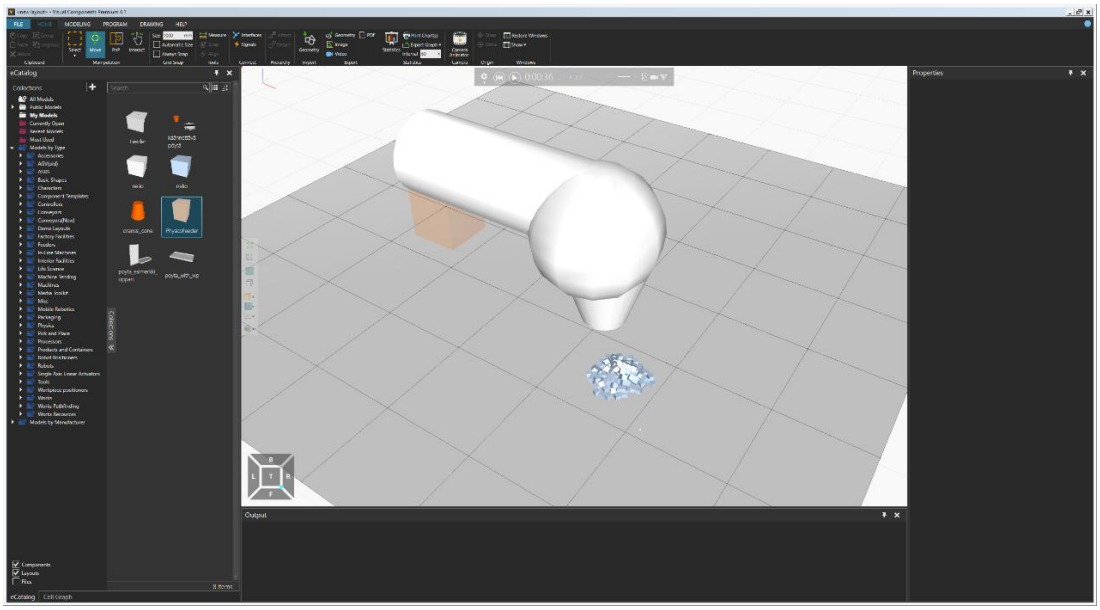


Kuva 14. Fysiikanmallinnus Visual Componentsilla

Betonielementin valmistusprosessin yksi vaihe on mallintaa betonivalukomponentti, joka täyttää muotin betonilla. Projektissa oli tarkoitus esittää tämä mahdollisimman yksinkertaisesti alussa ja kehittää se näyttävämmäksi myöhemmin. Yksinkertainen tapa on pysäyttää pöytä tiettyyn kohtaan polkua, siirtää ihmisiä ja betonivalukomponenttia pöydän päällä kierros tai kaksi ja lisätä maailmaan laatikko pöydän sisälle. Tällöin betonielementti olisi valmis uuniin.

Polkujen teko roboteille tapahtuu Path-käyttäytymisellä ja polun koon ja suuntien määrittämisellä Python-koodilla.

Fysiikanmallinnusta käytettäessä pitää olla varovainen simulaation ajettavuuden kanssa, sillä liian monta komponenttia maailmassa aiheuttaa ohjelman kaatumisen. Yksi mahdollisuus on käyttää liukuhihnaa, joka kuljettaa tarpeeksi pieniä neliöitä eteenpäin ja tiputtaa nämä neliöt suuttimesta muotille ja pysäyttää liukuhihnan, kun neliöitä on muotilla tarpeeksi.



Kuva 15. Betonivalupaikan fysiikanmallinnus Visual Componentsilla

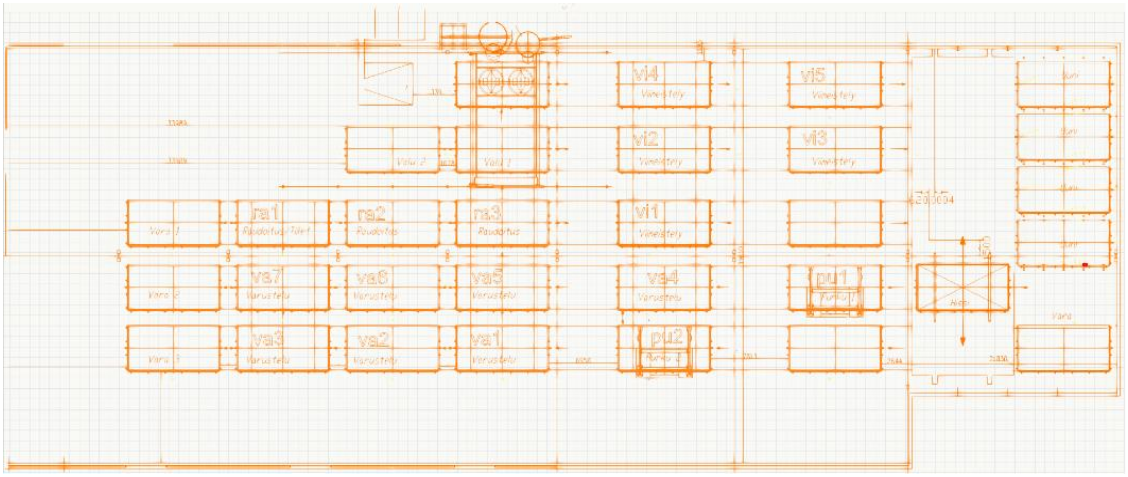
Neliöille annettaisiin fysiikka, joka mahdollistaa niiden tippumisen alaspäin, kun ne saavuttavat liukuhihnan pään. Neliöitä käytettäisiin pallojen sijaan, koska palloilla on huomattavasti enemmän polygoneja kuin neliöillä, joten ne kuormittaisivat ohjemaa tarpeettomasti.

Visual Components mallintaa fysiikkaa oletuksena käynnistäessäsi simulaation. Jos mihinkään komponenttiin maailmassa ei ole sidottu fysiikanmallinnus-käyttäytymistä, ei mikään maailmassa myöskään liiku. (Model a Physics Feeder 2016.) Komponenttien fysiikanmallinnusta voidaan rajoittaa monella tavalla. Entity-fysiikanmallinnus käyttäytymisen omaava komponentti saadaan toimimaan vain simulaation aikana, mikä tarkoittaa sitä, että et voi manipuloida komponenttia simulaation ollessa sammuksissa. Tämä tapahtuu valitsemalla In Physics –vaihtoehto.

Komponentin siirtäminen saadaan mahdolliseksi valitsemalla Out Of Physics -vaihtoehto. Komponentteihin saadaan liike-energiaa mallintava vaikutus simulaation aikana valitsemalla Kinematic -vaihtoehto. In Container -valinta on toimiva, kun komponentin halutaan käyttäytyvän maailmassa olevassa astiassa todenmukaisesti.

3 SIMULAATION RAKENTAMINEN

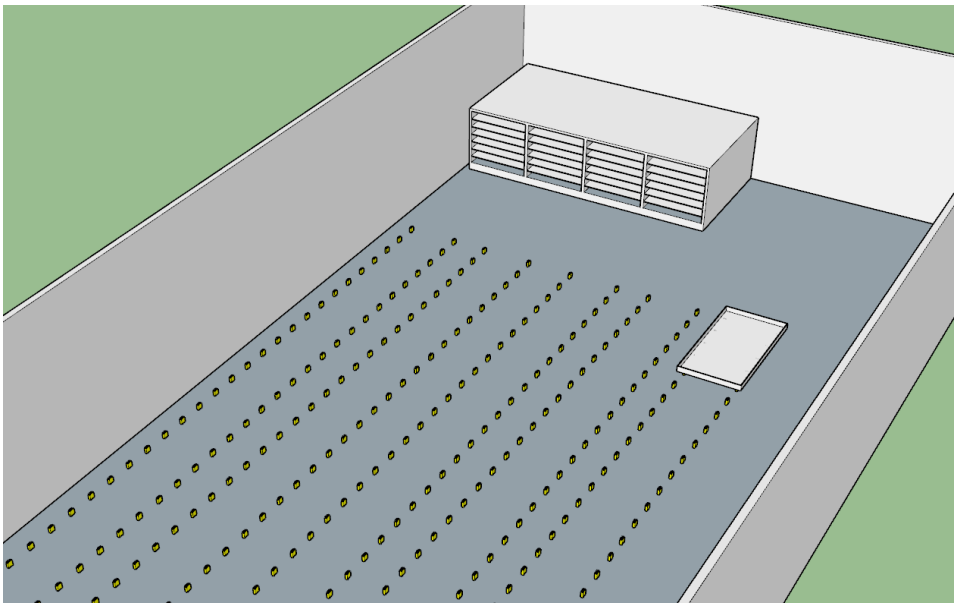
Projektissa käytettiin tehtaan pohjapiirrosta, joka saatiin toimeksiantajalta oikeiden mittasuhteiden saavuttamiseksi. Tutustuminen tehtaaseen tapahtui 11. kesäkuuta 2019. Tapaamisessa tuli selväksi mm. työntekijöiden lukumäärä, betonielementin valmistukseen liittyvä kiertokulku ja siihen kuluva aika, elementin rakennukseen liittyvät työkalut ja materiaalit, muottien valmistus tehtaalla sekä aliurakoitsijoiden käyttö rautavahvikkeiden valmistuksessa.



Kuva 16. Simuloitavan elementtitehtaan pohjapiirros

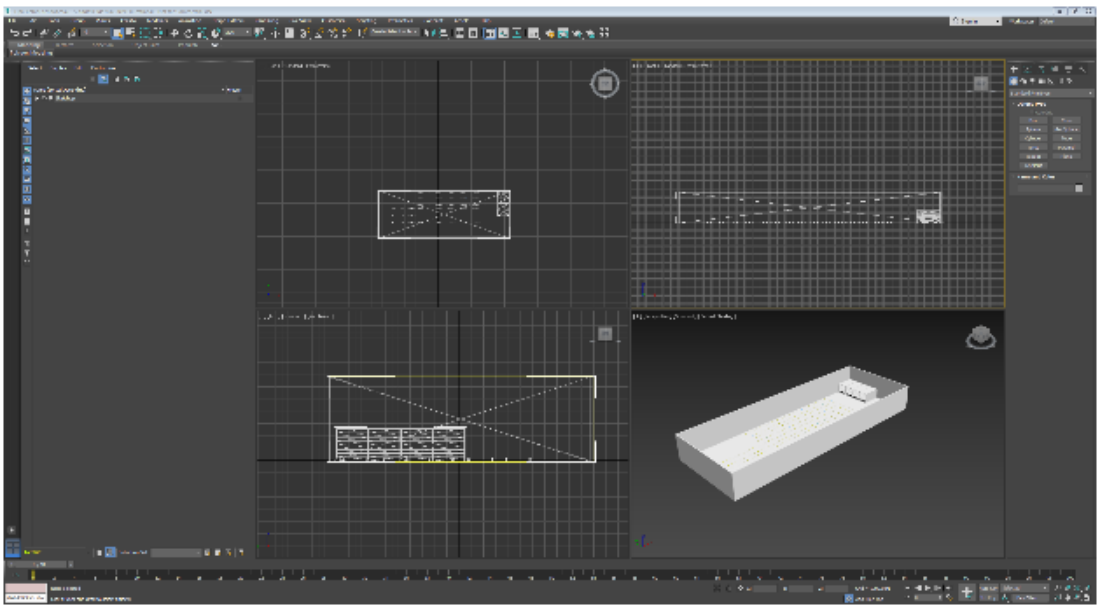
Tehdas ja simulaatiossa liikkuvat komponentit, kuten pöytä, jolle muotti asennetaan, mallinnettiin pohjapiirroksen mittojen avulla SketchUp-ohjelman 2017 versiolla.

3.1 Visual Componentsiin tuotavien mallien luominen



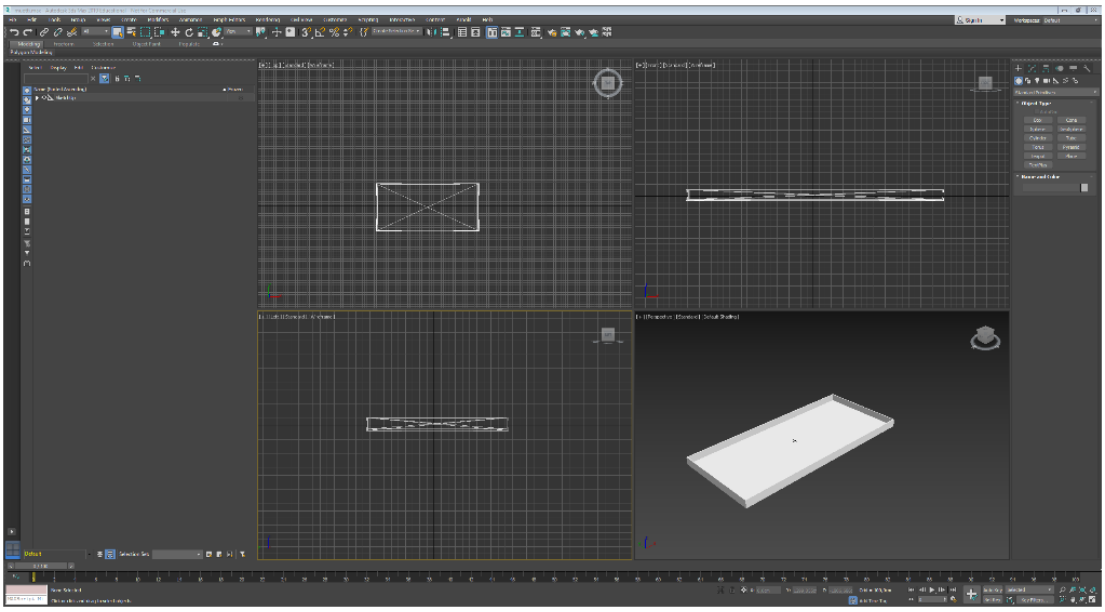
Kuva 17. Mallit luotiin SketchUp-ohjelmalla

Tehtaan malli ja siirrettävä pöytä vietiin erikseen 3ds Max -ohjelmaan siksi, että käytetyssä SketchUp-versiossa ei ollut mahdollisuutta tuottaa Visual Componentsin hyvin tukemia FBX-muotoisia tiedostoja. SketchUp-mallinnus-ohjelmalla luodut mallit vietiin DAE-muodossa (Digital Asset Exchange) 3ds Max -ohjelmaan. Tehtaan malli tuotiin Visual Componentsiin ensin siksi, että pystyttäisiin varmistamaan, säilyykö mittasuhteet tuonnissa oikeina.



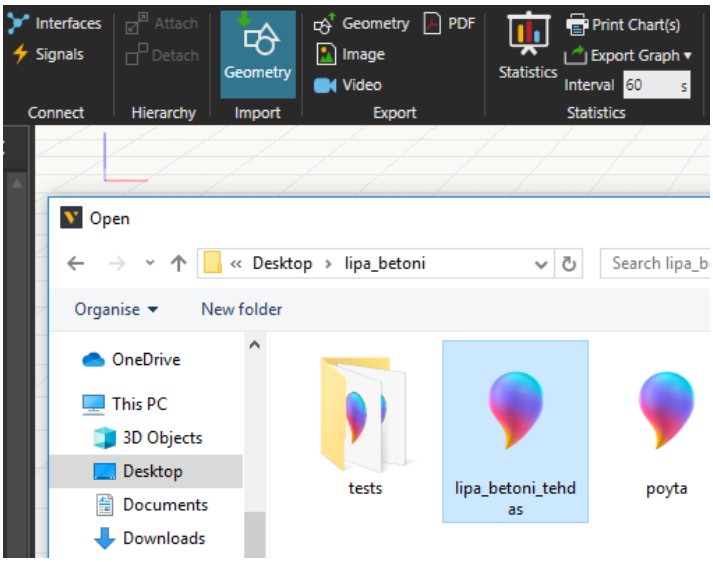
Kuva 18. Mallit muutettiin FBX-muotoon 3ds Max -ohjelmalla

3ds Max -ohjelmassa on tuki FBX-tiedostojen luonnille ja sillä voitiin myös hyvin tarkasti määritellä komponenteille oikea korkeus akseleiden leikkauspisteeseen (origoon) nähden. FBX-tiedostot ovat myös melko kevyitä. Näin saadaan Visual Componentsin simulaatioiden viiveet mahdollisimman pieniksi.



Kuva 19. Pöydän 3D-mallin muuntaminen FBX-muotoon 3ds Max -ohjelmassa

Visual Components käyttää komponentteja simulaatioon tuotaessa komponenttiin sidottua origoa. Origon ollessa väärässä paikassa komponenttiin nähden, myös luotu komponentti tulostuu väärään paikkaan. Komponentin tulostus väärään paikkaan estetään tuomalla komponentit 3ds Maxiin ja sijoittamalla se oikealle paikalleen, kuten paikalliseen origoon.



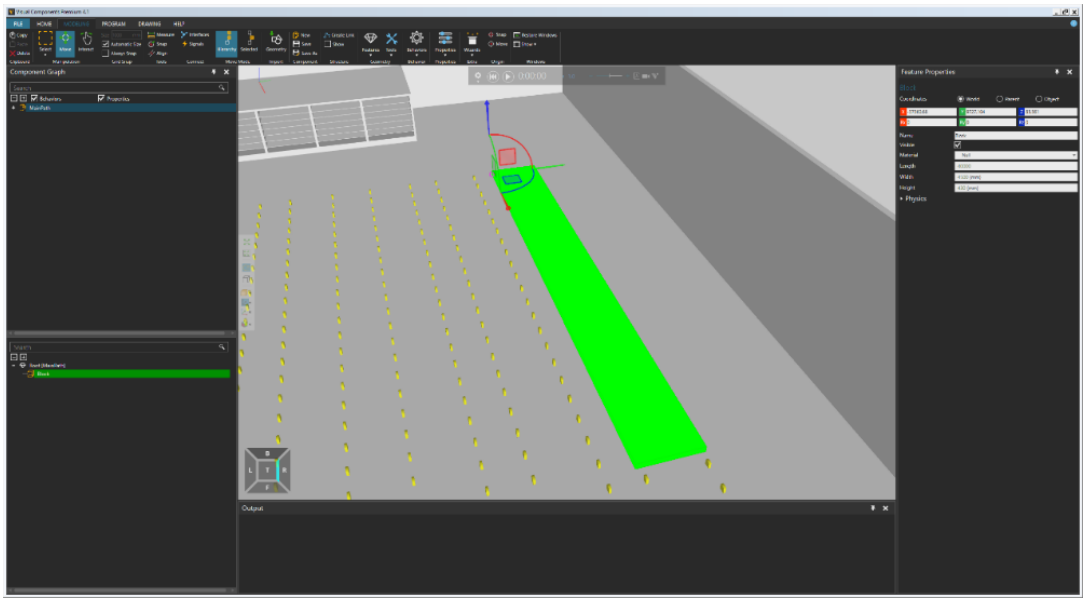
Kuva 20. FBX-muodossa olevat 3D-mallit tuodaan Visual Componentsiin.

Geometrian tuominen maailmaan tapahtuu Import-valikosta löytyvää Geometry-työkalua käyttämällä. Työkalu avaa resurssienhallinnan, josta voidaan navigoida tarvittavaan komponentin sijaintiin ja valita se. Valitsemisen jälkeen Visual Components avaa omat tuontiasetuksensa, josta päästään paremmin tutkimaan ja säätämään tuontiin liittyviä ominaisuuksia.

3.2 Oman polun rakentaminen pöydän siirtoon

Polkuja käyttämällä saadaan komponentteja kiinnitettyä määrättyihin ratoihin. Ratoihin voidaan kiinnittää sensoreita tunnistamaan komponenttien saapumiset tietyille paikoille rataa ja signaalien avulla viestittää linkitetylle Python-koodille, että toiminnon tai useampia toimintoja voidaan alkaa suorittaa. Sensoreina voidaan käyttää Visual Componentsin Frame-työkalua, jolla voidaan tulostaa niin monta kiintopistettä maailmaan kuin on milloinkin tarve.

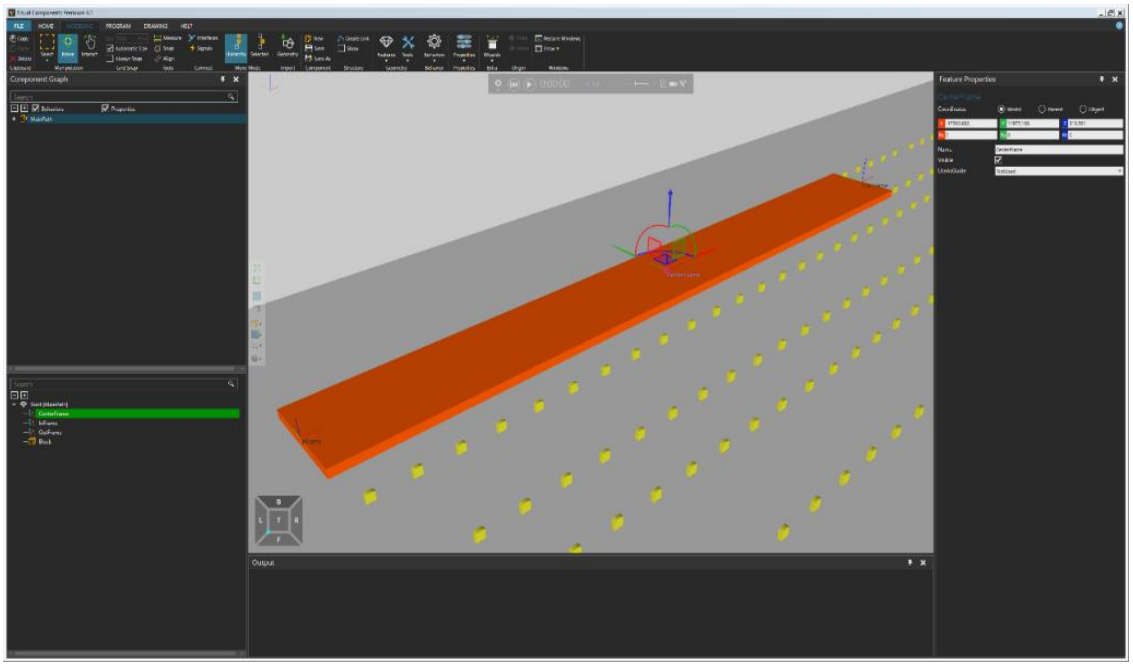
Luodut kiintopisteet olevat yhteydessä siihen komponenttiin, joka on ollut valittuna kiintopistettä tehtäessä. Polun rakentaminen aloitettiin luomalla ensin uusi komponentti Modeling-ympäristön ollessa aktiivisena painamalla ”New”-valintaa Component-valikon alta. Polun pituus, orientaatio ja sijainti maailmassa määritellään sen mukaan, mitä reittiä komponentin on tarkoitus kulkea tehtaassa. Uudelle komponentille annetaan nimi ja Features-valikosta ja sille annetaan sopivasti polkua muistuttava muoto. Tehtaan lattiaan kiinnitetyt kuljettimet toimivat mittapuuna polun korkeudelle.



Kuva 21. Komponentin sijainnin ja koon määrittelyä Visual Componentsissa

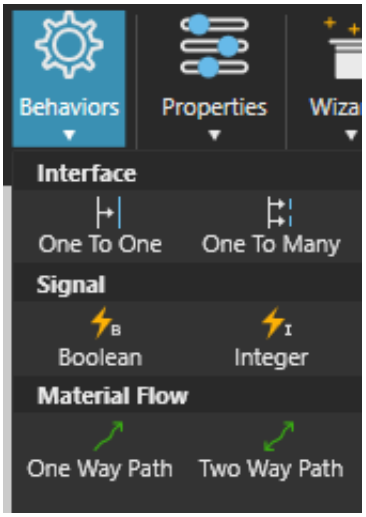
Kun sopivan kokoinen komponentti on luotu ja vietynä tehtaan siihen päätyyn, mistä betonielementin rakennus alkaa, voidaan komponentille antaa kiintopisteet kumpaankin päähän. Kiintopisteitä luodaan Features-valikosta löytyvällä Frames-työkalulla. Tämä työkalu tulostaa maailmaan kiintopisteen valittuna olevalle komponentille. Se voidaan siirtää haluttuun paikkaan käyttämällä Modeling-ympäristön valintatyökalua.

Kiintopisteet kannattaa nimetä niiden sijoituspaikkaan sopivalla nimellä, kuten esimerkiksi komponentin keskelle tuleva kiintopiste voidaan nimetä nimellä CenterFrame. Polulla kulkevan komponentin kulkusuuntaan liittyvät saapumis- ja lähtemispisteet voidaan nimetä esimerkiksi nimillä InFrame ja OutFrame.



Kuva 22. Frameja (kiintopisteitä) luotuna komponenttia varten

Framet on sijoitettu komponentin päihin ja keskelle sitä pitkin kulkevan pöydän kulkusuuntaan sopivalla tavalla. Komponentille pitää luoda kaksi eri käyttäytymismallityyppiä, Interface ja Material Flow. Polun tarvittavaan toimintaan vaaditaan vähintään kaksi yhden tehtävän käyttäytymismallia, joka kulkee nimellä One To One, sekä komponentin liikkumissuunnan antamiseksi vaadittavan Material Flow-tyypin alta löytyvä One Way Path -käyttäytyminen.

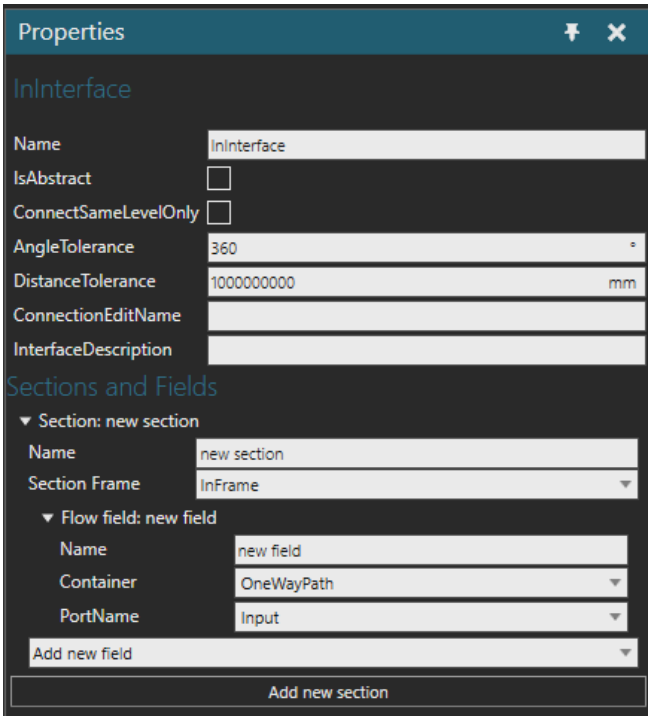


Kuva 23. Käytössä olevia käyttäytymisiä (Behaviors)

Pöytä ei tule siirtymään samaa polkua pitkin takaisin päin, eli polku ei tarvitse samasta valikosta löytyvää One To Many -käyttäytymismallia. One To One- ja One To Many -käyttäytymiset on tarkoitettu eri tehtäviin. (Model a Component Creator Part 1 2016.)

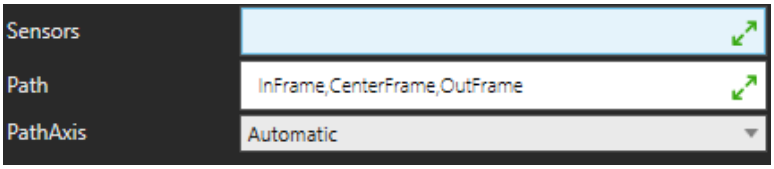
One To Many -käyttäytymismallit ovat sopiva toimintoalusta pienissä tiloissa, joissa robotit hakevat ja toimittavat komponentteja työstökoneille ja sieltä pois. Tämä ei sovi meidän simulaatioomme. One To One -käyttäytymismalleille annetaan samantapaiset nimet kuin jo nimetyille InFrame ja OutFrame-kiintopisteille. Tähän kävisivät esimerkiksi nimet InInterface ja OutInterface. Kummassakaan käyttäytymisessä oletuksena olevia määritteitä ei tarvitse vaihtaa. Section and Fields -kohdassa polulle annetaan uusi tehtävä.

Haluamme tehdä liukuhihnaa muistuttavan samalla nopeudella komponenttia siirtävän polun ja tähän sopiva tehtävä on Flow. Section-kohdassa valitaan alasvetovalikosta jo luomamme InFrame ja valitaan Fields-kohdan alasvetovalikosta Flow-tehtävä, joka antaa meille mahdollisuuden valita tehtävää suorittavan One Way Path -käyttäytyminen. Lisäksi voidaan määritellä, onko tehtävä komponentin vastaanottotyyppiä vai sen luovutustyyppiä. Tälle käyttäytymiselle annetaan vastaanotto-tyypin tehtävä valitsemalla se PortName-valikosta.



Kuva 24. Määritellään tehtäviä One To One -käyttäytymiselle

OutInterface-käyttäytyminen toteutetaan samalla lailla, mutta valitsemalla Out-Frame-vaihtoehto Section Frame -kohdassa ja muuttamalla käyttäytyminen luovutustyypin tehtäväksi. Material Flow -käyttäytymisessä ainoa muutettava asia on Path-valikosta nuolilla merkittyä valikonlaajennusvalintaa ja sen alta paljastuvaa +-merkkiä painamalla valittava InFrame, CenterFrame ja Out-Frame. Valintojen pitää olla tässä järjestyksessä, että komponentin kulku-suunta polulla on oikea.

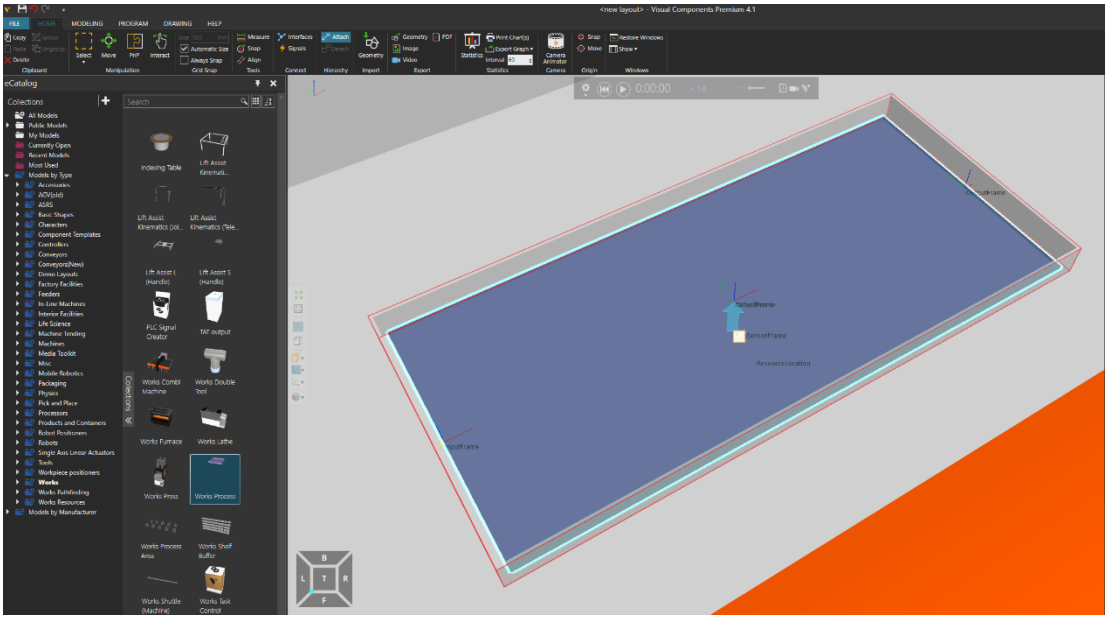


Kuva 25. Annetaan polku esimerkkikomponentille (tässä pöytä)

Projektia tehdessä komponenttien, polkujen ja tehtävien nimet olivat tärkeitä ohjelman käytön helpottamisen kannalta ja myös siksi, että toimeksiantajan edustajat pysyivät kärryillä projektin edistymistä esiteltäessä.

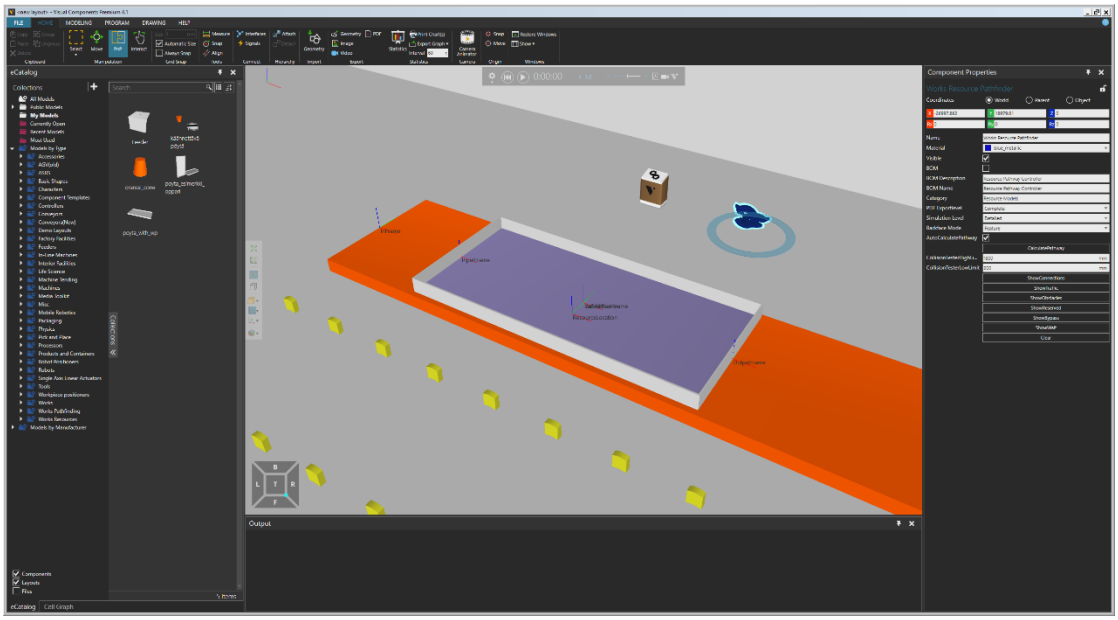
3.3 Muotin asennukseen valmiin pöydän luonti

SketchUp-ohjelmalla tehty pöytä on jo tuotu Visual Componentsiin, mutta sille ei ole annettu kaikkia vaadittavia lisäkomponentteja, jotta muotin siirtoon vaadittavat toiminnot voitaisiin koodata. Pöydästä tehdylle mallille sijoitetaan Visual Componentsin eCatalogista löytyvä Works Process -komponentti. Works Process -komponentit sisältävät vastaanottoon ja luovutukseen liittyvät käyttäytymiset sekä Frame-kiintopisteet, joilla on niiden käyttöön sopivat nimet. Works Process -komponentteja pystytään muuttamaan kuin mitä tahansa muita komponentteja niiden koosta lähtien ja tallentamaan ne omiksi komponenteikseen. Projektissa haluttiin simuloida tietyn kokoisia betonielementtejä, mutta monen kokoisten muottien tekemisen mahdollistaminen oli bonusta.



Kuva 26. Works Process -komponentin asettaminen pöydälle

Works Process -komponentti kiinnitettiin pöydän malliin käyttämällä Attach-työkalua, joka löytyy Home-ympäristön Hierarchy-valikosta ohjelman työkalupaneelista. Tämän valinnan ollessa aktiivinen valitusta komponentista lähtee sininen nuoli, joka osoittaa sitä komponenttia päin, jonka päällä hiiren osoitinta viet. Attach-työkalu sallii komponenttien liittämisen toisiin komponentteihin, jolloin niitä voidaan liikuttaa yhtä aikaa paikasta toiseen. (How to Attach Objects to Conveyor Paths 2018.)



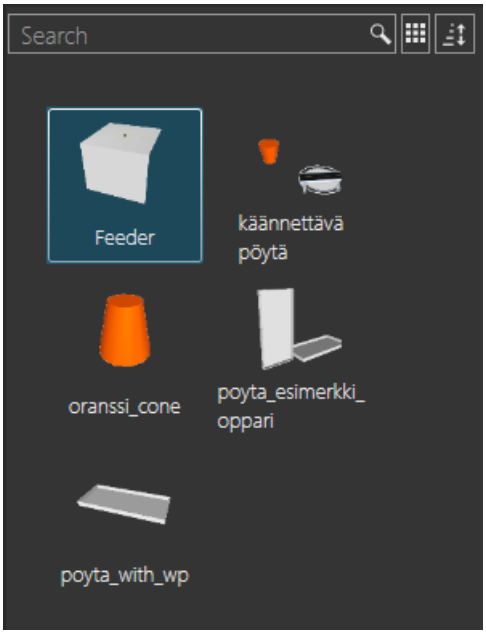
Kuva 27. Works TaskControl- ja Works Resource PathWay -komponenttien tuonti maailmaan

Tähän projektiin soveltuva pöytä, jossa on kiinnitettynä Works Process -komponentti, tallennetaan omana komponenttinaan omana tiedostonaan. Tätä komponenttia tullaan Feederien avulla tuottamaan simulaation aikana niin monta kappaletta kuin on tarpeen.

Works Process -komponentti tarvitsee projektiin liittyvää logiikkaansa varten Works TaskControl -komponentin, joka on väreiltään ruskea/valkoinen laatikko. Works TaskControl -komponentti vastaa kaikesta Works Process -komponentille annettavasta logiikasta. (Model a Component Creator Part 2 2019.)

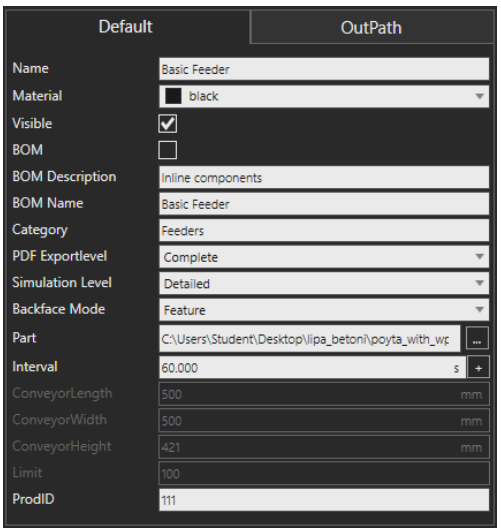
3.4 Uusien pöytien luominen simulaation aikana

Visual Componentsin Basic Feeder -komponenttia käytetään komponenttien luomiseen simulaation ollessa käynnissä ja se löytyy eCatalog-kirjastosta. Komponentille määrätään simulaation aikana tulostettava pöytä hakemalla se resurssienhallinnan kautta tai käyttämällä Visual Componentsin pöydälle antamaa VCID-tunnusta.



Kuva 28. Itse luotu Feeder-komponentti

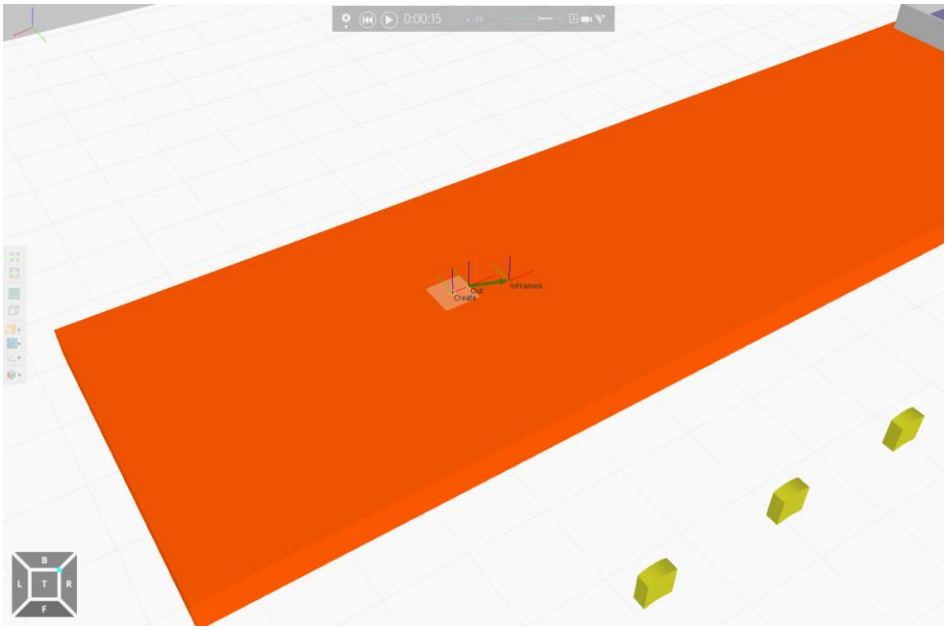
Basic Feeder -komponentille annetaan aikaväli, jonka kuluttua uusi komponentti luodaan ja annetaan nopeus, jolla komponentti liikkuu Basic Feeder -komponentin päällä. Simulaatiossa halutaan komponentin liikkuvan samaa nopeutta, kuin se tulee liikkumaan luodulla polulla (radalla).



Kuva 29. Aikavälin asettaminen Visual Componentsissa

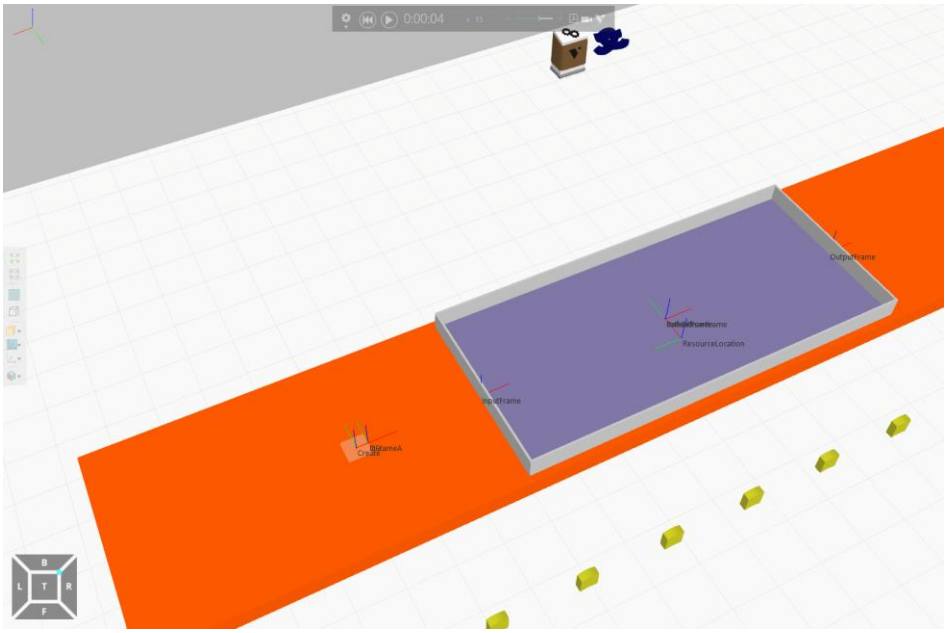
Visual Componentsin eCatalogin My Models -kansio toimii oletustallennuspaikkana omille komponenteille. Sieltä löytyy Feederiksi nimetty komponentti, jolle ollaan annettu tehtäväksi luoda uusia pöytiä 60 sekunnin välein.

Polulle annetut kiintopisteet toimivat Feederin liitoskohtina polkuun. Omiin malleihin tallennettu Feeder-komponentti viedään polun keskelle sopivalle etäisyydelle polun päästä lähelle aloituspään InFrame-kiintopistettä ja katsotaan, että se huomaa Feederin ja kiinnittyy siihen.



Kuva 30. Feeder-komponentti tuodaan lähelle InFrame-kiintopistettä

Vihreä nuoli kuvaa kiinnitysmahdollisuutta Feederin ja InFrame-kiintopisteen välillä Feederiä siirrettäessä. Nuolen ollessa näkyvillä hiiren painikkeen päästäminen irti liittää ne toisiinsa. Simulaation käynnistyessä ensimmäinen pöytä luodaan heti ja seuraava luodaan minuutin kuluttua.

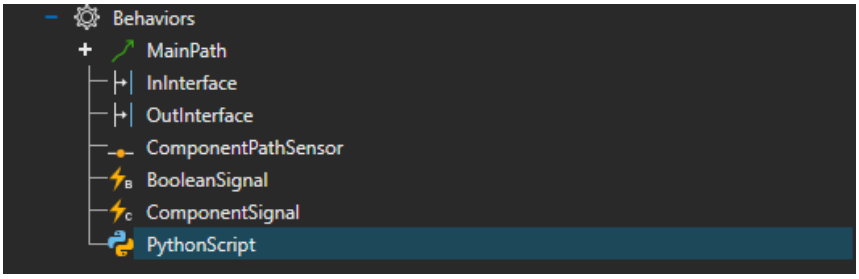


Kuva 31. Simulaatio toiminnassa Visual Componentsissa

Pöytä siirtyy polkua pitkin toisessa päässä odottavaa OutFrame-kiintopistettä kohti ja häviää sen saavutettuaan. Pöydän jääminen maailmaan simulaation suorittamisen jälkeen ei ole tarpeellista, mutta simulaatiossa on tarkoitus pysäyttää pöytä useaan kohtaan polkua muotin asennusta, raudoitusta, valua, uunissa käyntiä ja pesua varten.

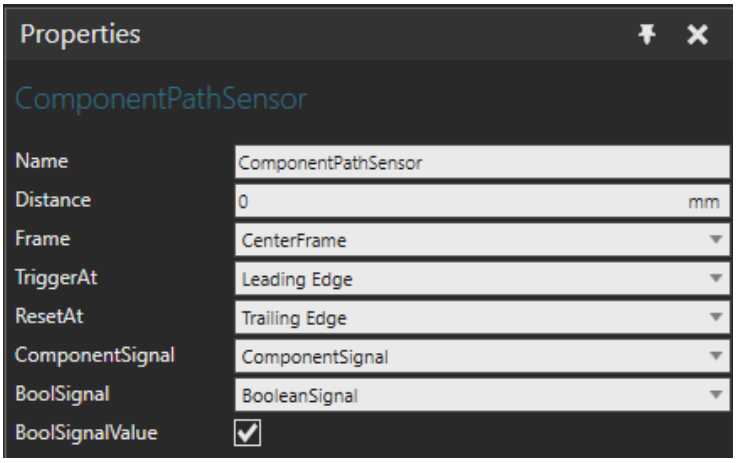
3.5 Komponenttien siirto pysäytetylle pöydälle

Komponentin siirtoon käytetään Feed- ja Need-tehtäviä. Works Process -komponenttien ominaisuuksista löytyy tehtävänantoon liittyvä valikko ja niiden luonnin jälkeen ilmestyy tehtävään liittyvät asetukset. Simulaation pysäytys tietyn komponentin osalta onnistuu käyttämällä signaaleita. ComponentPathSensor-käyttäytyminen mahdollistaa ComponentSignal-käyttäytymisen ja BooleanSignal-käyttäytymisen liittämisen toisiinsa Pythonilla. (Model a Raycast Sensor, 2017.)



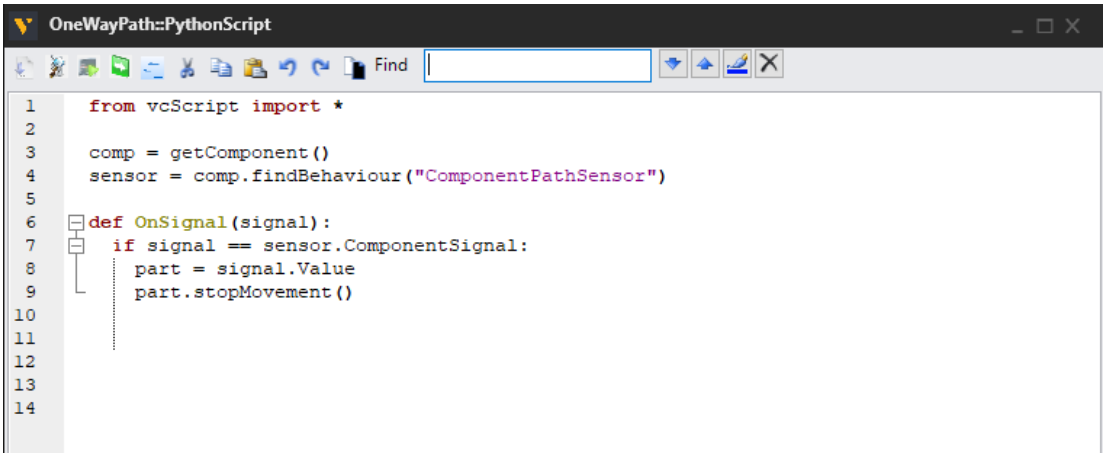
Kuva 32. Simulaation Behaviors-määrittelyt

Sensorit vaativat kiintopisteen polulla, että ne voidaan liittää signaaleihin. BooleanSignal ja ComponentSignal liitetään Python-ohjelmaan valitsemalla ne näiden kahden ominaisuuksista löytyvistä valikoista. MainPath-polun ominaisuuksista valitaan Sensors-valikosta edellä luotu ComponentPathSensor. ComponentPathSensor-käyttäytymisen ominaisuuksista löytyy mahdollisuus valita valittuun polkuun liittyvät kiintopisteet. Tähän valitaan polkuun kiinnitetty CenterFrame.



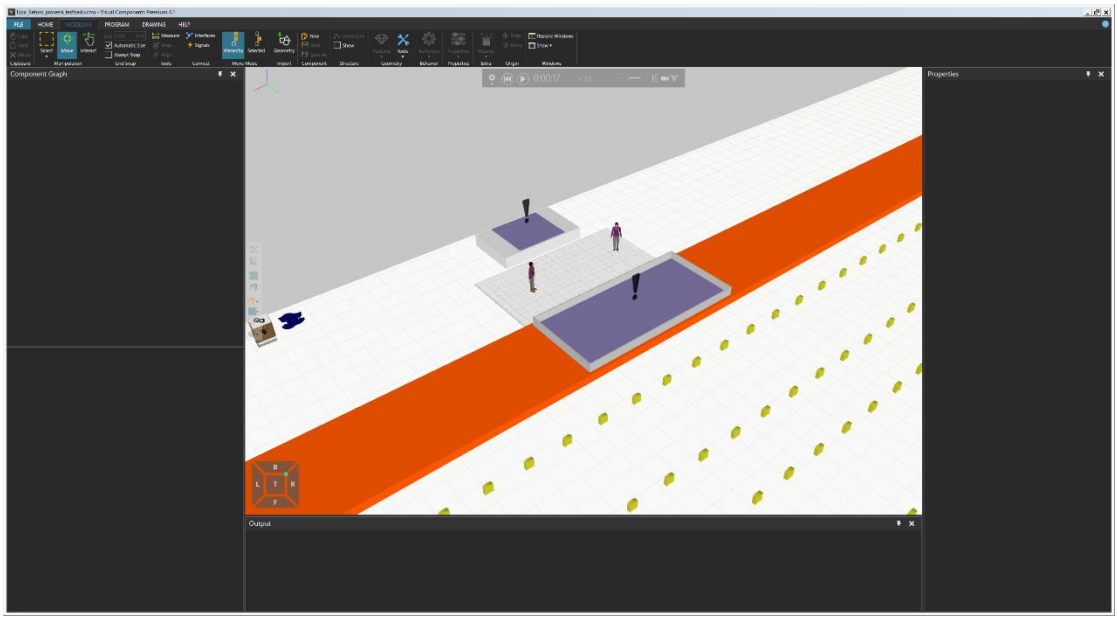
Kuva 33. ComponentPathSensor-käyttäytymisen ominaisuudet

Valittuna olevat signaalit ovat yhteydessä PythonScript-käyttäytymiseen, jolloin voidaan aloittaa Python-koodin rakentaminen. Pythonilla on haettu rakentamamme polku, jolle on annettu nimi comp. Simulaatiosta haetaan ComponentPathSensor-käyttäytyminen ja sille annetaan nimi sensor.



Kuva 34. Käyttäytymisen (komponentin pysäytys) määrittelyä Python-kielellä

OnSignal-funktiolle on annettu if-lause, joka seuraa sensoria. Kun se havaitsee komponenttisignaalin, se pysäyttää simulaation liikkuvan komponentin osalta. (Model a Volume Sensor 2017.) Komponentin pysähdyttyä CenterFrame-kiintopisteen kohdalle voidaan suorittaa laatikonhaku-simulaation mukainen tehtävä ihmisresurssien avulla.



Kuva 35. Pöydän simulaatio pysäytetty ja kaksi ihmisresurssia on tuotu simulaatioon.

Polun viereinen pöytä lisää muotin osia maailmaan. Ihmisresurssien määrä tulisi vaikuttamaan käytettyyn aikaan, mutta saattaa viedä myös resursseja muualta. Resurssien optimointi on yksi tämänkin simuloinnin päämotiiveista.

4 POHDINTAA JA JOHTOPÄÄTÖKSIÄ

Projektissa ollaan päästy tähän vaiheeseen ja työtä tullaan jatkamaan lisätyövoimin. Jatkossa simulaatiota pyritään kehittämään niin, että voidaan seurata komponenttien tietä polulla ja samalla voidaan ottaa tiedot ylös tehtävien kestoista yhden ja useamman resurssin (esim. ihmisen) osalta. Lopullinen simulaatio vaatii Visual Componentsin laajentamista Python-ohjelmoinnin avulla. Tämän opinnäytetyöprojektin puitteissa ohjelmoin jo pöydän pysäytyksen. Seuraava askel on toteuttaa Python-koodi, joka aloittaa muotin osien hakemisen ja asentamisen, jolloin tätä samaa toimintoa voidaan käyttää myös raudoitusalueella. Tämän jälkeen projekti on täysin toimeksiantajan vaatimusten mukainen.

Betonivalu tapahtuu pohjapiirroksessa selvänä näkyvässä pullonkaulassa, mikä rajoittaa työstettävän muotin määrän yhteen. Ratkaisu tämän pullonkaulan poistamisen kanssa ei ole helppo, koska tehtaassa on lattialämmitysratkaisuja, jotka ovat pullonkaulan osasyy. Pullonkaulan poisto tehtaasta mahdollistaisi useamman betonivalun teon samanaikaisesti ja kasvattaisi tarvetta

myös uusille työntekijöille. Simulaation rakentaminen tehtaasta mahdollistaa eri ratkaisuvaihtoehtojen testaamisen nopeasti ja edullisesti.

4.1 Ajatuksia Visual Componentsin käytöstä

Visual Components on käyttötarkoitukseensa nähden mielestäni melko vaativa ohjelma. Suurten tehdasratkaisujen simuloiminen ohjelmalla vaatii laajaa Python-ohjelmointikielen ja ohjelman omien komponenttien käytön tuntemusta. Vasta-alkajan saattaa olla vaikea päästä sinuiksi ohjelman kanssa. Visual Componentsin nettisivut ovat pitkälti ainoa lähde mihin voidaan nojata ohjelmaa opiskeltaessa ja käytettäessä. Visual Components -yrityksellä on harjoitusvideoita ja videokursseja ohjelmaan liittyen. Ne antavat mielestäni suppean kuvan ohjelmistosta ja sen käytöstä. Ohjelman opettelu on lähinnä kiinni siitä, että olet valmis tekemään paljon töitä opetellessasi jotain tiettyä asiaa ohjelmasta. Ohjelman opettelu tapahtuukin siis pääosin testaamalla, päättelemällä ja joissain kohdissa lähes arvaamalla, mikäli halutusta ratkaisusta ei ole suoria harjoituksia saatavilla.

Tiettyjen asioiden selvittämistä voi auttaa Visual Componentsin keskustelupalstat, mutta siellä olet sen varassa, että apu tulee muilta käyttäjiltä. Kysyttyjä kysymyksiä eri ongelmista on keskustelupalstalla satoja, jopa tuhansia. Ratkaisun löytäminen juuri sinun ongelmaasi näistä jo vastauksen saaneista kysymyksistä saattaa viedä paljon aikaa eikä ratkaisua välttämättä edes löydy. Visual Components tarvitsee mielestäni kipeästi oppimateriaalia sen perustyökalujen käytöstä. Tähän kuuluisi komponenttikirjaston sisällön kartoittaminen ja selitteet kaikesta mitä Visual Componentsin työkalupalkista löytyy. Käyttäytymismallien eroavaisuuksista ja niille annettavista tehtävistä pitäisi olla selvät ohjeet ja käytännön testausta.

4.2 Viimeinen analyysi

Visual Componentsin vertaaminen muihin samantapaisiin ohjelmiin on mielestäni jo yhden opinnäytetyön arvoinen tutkielma. Pelimoottorit, kuten Unity ja Unreal Engine sisältävät samaan tapaan omat toimintokirjastonsa ja niillä on valtava määrä oppimismateriaalia yksittäisten työkalujen käytöstä ja uusia työkaluja tulee jatkuvasti lisää. Visual Components tarvitsisikin taaksensa yhtä omistautuneen ja innokkaan käyttäjäkunnan, jotta Visual Components voisi

yhdessä käyttäjiensä kanssa kehittyä paremmaksi. Visual Componentsilla olisikin mielestäni mahdollisuus kehittyä suuremman käyttäjäkunnan tuotteeksi tarpeeksi suurella panostuksella sen käyttökokemukseen ja kunnolliseen ohjeistukseen, sekä ohjelman markkinointiin nykyistä yleiskäyttöisempänä simulointi- ja prosessien mallinnus -ohjelmana.

LÄHTEET

Discover what you can do with the Visual Components eCatalog. 2017. Visual Components. Blogi. Saatavissa: https://www.visualcomponents.com/insights/blog/discover-visual-components-ecatalog/ [viitattu 10.11.2019].

How to Attach Objects to Conveyor Paths. 2018. Visual Components. Saatavissa: https://academy.visualcomponents.com/lessons/how-to-attach-objects-to-conveyor-paths/ [viitattu: 13.11.2019].

Kasurinen, J. 2009. Python 3 -ohjelmointi. Jyväskylä: Docendo.

Model a Component Creator – Part 1. 2016. Visual Components. WWW-dokumentti. Saatavissa: https://academy.visualcomponents.com/lessons/model-a-component-creator-part-1/ [viitattu 12.11.2019].

Model a Component Creator – Part 2. 2019. Visual Components. WWW-dokumentti. Saatavissa: https://academy.visualcomponents.com/lessons/model-a-component-creator-part-2/ [viitattu 13.11.2019].

Model a Physics Feeder. 2016. Visual Components. WWW-dokumentti. Saatavissa: https://academy.visualcomponents.com/lessons/model-a-physics-feeder/?course=137 [viitattu 15.11.2019].

Model a Raycast Sensor. 2017. Visual Components. WWW-dokumentti. Saatavissa: https://academy.visualcomponents.com/lessons/model-a-raycast-sensor/ [viitattu 17.11.2019].

Model a Volume Sensor. 2017. Visual Components. WWW-dokumentti. Saatavissa: https://academy.visualcomponents.com/lessons/model-a-volume-sensor/ [viitattu 18.11.2019].

Simulate Process with Human, Machine and Robot. 2017. Visual Components. WWW-dokumentti. Saatavissa: https://academy.visualcomponents.com/lessons/simulate-process-with-human-machine-and-robot/ [viitattu 13.11.2019].

Works Library – Basics of Task Creation. 2017. Visual Components. WWW-dokumentti. Saatavissa: https://academy.visualcomponents.com/lessons/works-library-basics-of-task-creation/ [viitattu 11.11.2019].