Tampereen yliopisto

Jaakko Huotari

RENKAAN LASKENTAMALLIN LUOMISEN AUTOMATISOINTI

Tekniikan ja luonnontieteiden tiedekunta
Diplomityö
Kesäkuu 2020

TIIVISTELMÄ

Jaakko Huotari: Renkaan laskentamallin luomisen automatisointi
Diplomityö
Tampereen yliopisto
Konetekniikan diplomi-insinöörin tutkinto-ohjelma
Kesäkuu 2020

Tässä diplomityössä ohjelmoitiin elementtimenetelmälle soveltuva verkkogeneraattori, joka soveltuu henkilöauton renkaan pyörähdyssymmetriseen analysointiin. Tavallisesti renkaan pyörähdyssymmetrinen laskentamalli luodaan ja verkotetaan käsin. Tarkoituksena oli automatisoida tämä paljon aikaa vievä toimenpide.

FEM-analyysejä tehdään Abaqus-laskentaohjelmistolla. Laskennan nopeuttamiseksi käytetään ali-integroituja nelikulmioelementtejä, jotka ovat herkkiä epästabiilille tiimalasimuodolle. Muodonmuutosten ehkäisemiseksi elementtejä stabiloidaan tiimalasimuotoihin nähden ortogonaalisilla stabilointivoimilla. Työn ensimmäisessä vaiheessa tarkasteltiin, miten Abaqus käsittelee stabiloituja ja stabiloimattomia elementtejä. Elementeistä muodostettiin puhtaassa taivutuksessa oleva testikappale, jonka taipumaa verrattiin analyyttisiin tuloksiin. Laskennan tuloksista havaittiin, että tiimalasimuotojen heräämisen lisäksi stabilointi ehkäisee tehokkaasti myös leikkaus- ja tilavuuslukkiutumista. Stabiloidut elementit toimivat merkittävästi paremmin kuin stabiloimattomat elementit, riippumatta verkon tiheydestä.

Työn jälkimmäinen osa keskittyi verkottajan toiminnan kuvaamiseen. Renkaan geometriamallista tuodaan geometriaviivojen koordinaattipisteitä. Pisteet projisoidaan projektiokäyrään nähden kohtisuorille normaalisuorille, mikä maksimoi elementtien suorakulmaisuuden. Pisteet järjestetään koordinaattien mukaiseen järjestykseen ennen solmupisteiden luomista. Lähekkäin olevat solmupisteet yhdistetään numeeristen virheiden ehkäisemiseksi. Pisteet yhdistetään pystypuiksi normaalisuorien mukaisesti. Solmut yhdistetään vaakapuiksi pystypuiden solmumäärien mukaan. Lopulta pysty- ja vaakapuut kootaan elementeiksi.

Uusi verkottaja nopeuttaa merkittävästi mallin luomista ja verkottamista. Käsin tehtäessä tähän kuluu tunteja. Uusi verkottaja laskee tämän ajan minuutteihin, vaikka se ei vielä täysin poistakaan käsin tehtävää työtä. Mallien kasvanut systemaattisuus parantaa mallien vertailtavuutta.

Avainsanat: Rengas, laskentamalli, FEM, automatisointi

ABSTRACT

Jaakko Huotari: Automation of Creation of Tire Simulation Models
Master of Science Thesis
Tampere University
Master's Degree Programme in Machine Design
Kesäkuu 2020

A finite element mesh generation program was coded for a passenger car tire analysis. Traditionally, these axisymmetric models have been created and meshed by hand. The aim of this work was to automate this time-consuming task.

Finite element method (FEM) -simulations are done with simulation software Abaqus. Reduced integrated quadrilateral elements are used to increase simulation speed. This form is susceptible to hourglass-deformations. To prevent these deformations, the elements are stabilized using forces orthogonal to desired deformation modes. Handling of stabilized and unstabilized elements was studied in Abaqus. An ideal disc was loaded with pure bending numerically and analytically. The results showed that enhanced stabilization prevents hourglass deformations and limits shear- and volumetric locking regardless of mesh density.

A new mesh creation algorithm was described. Coordinate points are imported from tire geometry model. The points are projected onto normal lines orthogonal to a selected projection curve, which maximizes element orthogonality. The points are then sorted according to their coordinates. Points close together are merged before node creation. Nodes are connected as vertical rails based on their normal lines. Nodes are connected as horizontal rungs based on rail node counts. The nodes in rails and rungs are then combined into elements.

The new mesh creation algorithm improves model and mesh creation speed. Done by hand, this task takes hours. Done automatically, it takes only minutes, although manual corrections are still required. A systematic method for creating the models increases model comparability.

Keywords: Tyre, simulation model, FEM, automation

ALKUSANAT

Tämä diplomityö on tehty työsuhteessa Nokian Renkaat Oyj:lle. Aloitin päätoimisen työnteon 2011 tavoitteenani diplomityön tekeminen työn ohessa heti, kun se on mahdollista. Muutamia vuosia myöhemmin pääsin aloittamaan. Pitkän ponnistelun jälkeen tämä työ on lopulta valmis.

Haluan kiittää ohjaajaani DI Jani Räisästä opastuksesta sekä prof. Reijo Kouhiaa työn tarkastamisesta. Kiitän myös nykyisiä ja entisiä rakennekehitysläisiä DI Samu Lepistöä, DI Mikko Mäkelää ja DI Lotta Nortamoa tuesta.

Lisäksi haluan kiittää perhettäni jatkuvasta kannustuksesta.

Helenalle.

Tampereella 2.6.2020

Jaakko Huotari

SISÄLLYS

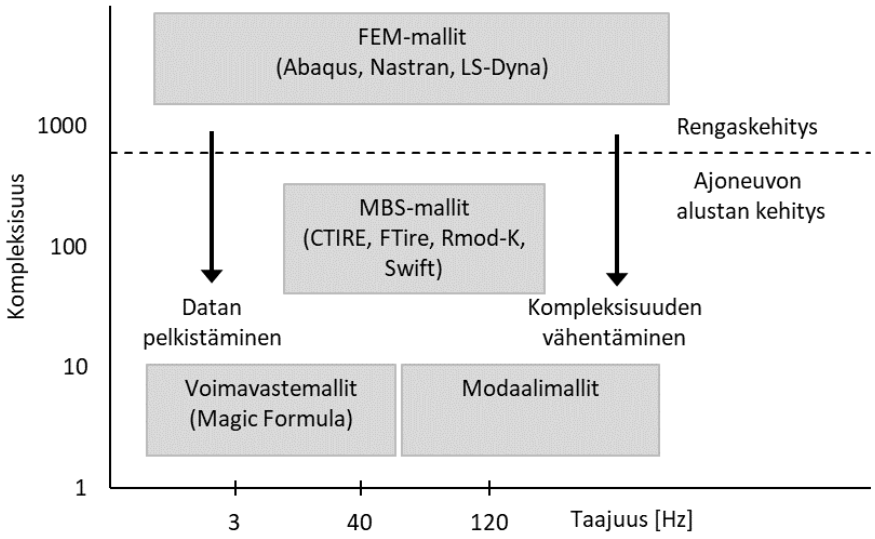
LYHENTEET JA MERKINNÄT

FEM	Elementtimenetelmä (Finite Element Method)
JLB-nauha	vahvikenauha (jointless band)
$\boldsymbol{n}_{i,j}$	vektorin i-komponentti j-koordinaatin suhteen
A	kaksiulotteisen elementin pinta-ala
$\boldsymbol{A}$	kinemaattisen matriisin koordinaattitekijä
a_{ji}	siirtymäkenttävakio
$\boldsymbol{B}$	kinemaattinen matriisi
$\widehat{\boldsymbol{B}}$	keskiarvomuotoinen kinemaattinen matriisi
$\widehat{\boldsymbol{B}}_c$	keskiarvomuotoisen kinemaattisen matriisin vakio-osa
$\widehat{\boldsymbol{B}}_{\boldsymbol{n}}$	keskiarvomuotoisen kinemaattisen matriisin muuttujaosa
$\widehat{\boldsymbol{b}}_i$	keskiarvomuotoisen kinemaattisen matriisin tekijä
c_{ji}	siirtymäkenttävakio
$\boldsymbol{D}$	materiaalimatriisi
E	materiaalin kimmomoduuli
$\boldsymbol{\epsilon}$	venymät
ε	vaimennuskerroin
ζ	isoparametrinen koordinaatti
$\boldsymbol{f}$	sisäiset voimat
$\boldsymbol{f}_i^{stab}$	tiimalasi-stabilointivoimavektori
$\boldsymbol{G}$	gradienttimatriisi
H_{ij}	jäykkyysmatriisin tekijöiden kertoimet
$\boldsymbol{h_i}$	tiimalasiperusvektori
h_i	isoparametrinen kerroin
η	isoparametrinen koordinaatti
I	solmunumero
$\boldsymbol{J}$	Jakobin matriisi
$\boldsymbol{K}$	jäykkyysmatriisi
K_i	kuormituksen reunaehtokerroin
$\boldsymbol{k}^c$	elementin jäykkyyden vakio-osa
$\boldsymbol{k}^e$	elementin jäykkyys
$\boldsymbol{k}^{stab}$	elementin jäykkyyden muuttujaosa
k_{ij}	jäykkyysmatriisin tekijät
λ	Lamén ensimmäinen parametri
μ	Lamén toinen parametri
n	pisteiden lukumäärä
ν	Poissonin vakio
ξ	isoparametrinen koordinaatti
$\boldsymbol{P}$	ulkoiset pistevoimat
$\boldsymbol{Q}$	kaikkien solmujen siirtymävektori
Q_i	vaimennusenergia
$\boldsymbol{q}$	solmujen siirtymävektori
ρ	materiaalin tiheys
S	elementin pinta
$\boldsymbol{s}$	vakiosiirtymän perusvektori
$\boldsymbol{\sigma}$	jännitykset
$\boldsymbol{T}$	ulkoiset pintavoimat

t_e	kaksiulotteisen elementin paksuus
U_e	elementin kimmoenergia
$\boldsymbol{u}$	siirtymävektori
$\boldsymbol{\Upsilon}$	tiimalasimuotovektori
$\widehat{\boldsymbol{\gamma}}$	keskiarvomuotoinen tiimalasimuotovektori
V	kolmiulotteisen elementin tilavuus
$\boldsymbol{\phi}$	perusvektori, muotofunktiovektori
w_i	elementin numeerinen painokerroin
Ω	elementin alue

1. JOHDANTO

Renkaan simulointimalleilla on useita käyttökohteita. Niitä käytetään etenkin rengas- sekä ajoneuvoteollisuudessa kehitystyökaluina. Rengasvalmistajan näkökulmasta rengasmallin tulee pystyä ennustamaan rengasominaisuuksia renkaan komponenttien ominaisuuksien perusteella (Selig ym., 2017, s. 37, 43). Kuvassa 1 on esitetty erilaisten olemassa olevien rengasmallien kompleksisuutta suhteessa niiden dynaamiseen taajuskäyttöalueeseen. Yksityiskohtaisilla malleilla on kaikista suurin käyttöalue. Niissä pyritään mallintamaan monimutkaista järjestelmää kuvaamalla sen komponentit tarkasti. Tähän käytetään yleensä elementtimenetelmää (Finite Element Method, FEM). Sitä käyttävien mallien luominen on erittäin työlästä niiden monimutkaisuudesta johtuen. Ajoneuvoteollisuuden ratkaisu mallien käytettävyyden lisäämiseen on kompleksisuuden vähentäminen. Tälläisissä yksinkertaistetuissa voimavastemalleissa rengasta kuvataan laatikkona, joka tuottaa tietyn voima- ja momenttiresultantin tietyillä käyttöparametreilla. Näistä malleista tunnetuin on Pacejkan luoma Magic Formula (Pacejka, 2012). Yksinkertaistus vähentää myös mallista saatavissa olevan datan määrää (Leister, 2015). Tässä diplomityössä pyritään helpottamaan elementtimenetelmän käyttöä.



Kuva 1. *Rengasmallit ja niiden käyttökohteet (perustuu lähteeseen Leister, s. 5)*

Laskentamallilla voidaan selvittää, miten renkaan komponentit ja ulkomitat vaikuttavat renkaan tiekontaktiin ja kuormituksiin. Yleensä mallista luodaan erilaisia variaatioita, joissa muutetaan komponenttien ominaisuuksia, renkaan mittoja tai muotoja. Jos muutokset variaation geometriassa ovat suuria, malli pitää muodostaa renkaan perusgeometriasta lähtien. Tavanomaisesti tämä tehdään piirtämällä CAD-ohjelmistossa renkaan ääriviivat ja rajapinnat ja siirtämällä ne FEM-ohjelmistoon. Mallin rakentaminen tähän tapaan vaatii

paljon käsityötä. Mallin luomiseen kuluu yhteensä 4-8 tuntia tekijän nopeudesta riippuen. Suurin osa tästä ajasta kuluu geometrian luomiseen ja mallin verkottamiseen.

Renkaan komponentit ovat ohuita, kaarevia levyjä. Niiden mallintaminen perinteisillä elementtimenetelmän keinoilla on vaikeaa. Verkon kaartuessa automaattinen verkottaja vääristää elementtien muotoa liikaa, jolloin niitä ei voida käyttää laskennassa. Perinteinen tapa korjata tämä tilanne on kasvattaa verkon tiheyttä, jolloin verkottaja osaa toimia paremmin. Seurauksena tihentämisestä mallin laskenta-aika kasvaa.

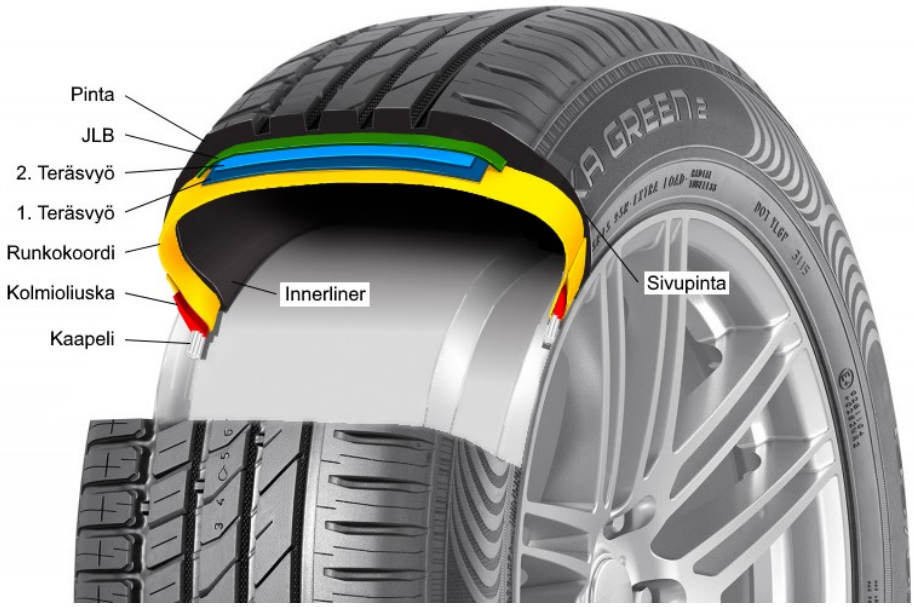
Myös mallien laadussa on eroa. Kokenut suunnittelija luo mallin nopeasti, mutta myös mallin laskentanopeus paranee. Hyvän simulointimallin laskenta-aika voi olla noin tunnin. Huonon mallin laskemiseen saattaa mennä kymmenenkin tuntia tai laskenta epäonnistuu. Hyvän ja huonon mallin eroa ei ole pystytty aukottomasti esittämään. Tämän työn tavoitteena on kehittää itsenäiseen toimintaan pystyvä ohjelmisto, joka tuottaa laadukkaita laskentamalleja. Laskentamallien tulee olla nopeita, rakenteeltaan ja verkoltaan systemaattisia sekä tarkkoja. Ohjelmistolle osoitetaan rengasgeometria, jonka pohjalta se luo itsenäisesti FEM-laskentamallin.

Työn rakenne jakautuu kahteen osaan. Ensin selvitetään, mitä renkaan rakenteen ja teorian perusteella pitäisi ottaa huomioon mallia muodostettaessa. Seuraavaksi esitetään, miten malli käytännössä verkotetaan. Lopuksi tarkastellaan, mitä valituista lähestymistavoista seuraa.

2. RENGAS JA RENKAAN LASKENTAMALLI

2.1 Renkaan rakenne

Henkilöauton rengas on komposiiteista valmistettu paineastia. Yleisimmät henkilöautonrenkaat ovat vyörenkaita, joten tässä työssä keskitytään vain niihin. Rengas koostuu erilaisista kumikomponenteista ja niiden sisään asetetuista, lujittavista vahvikkeista. Nämä komponentit on yleensä ladottu päällekkäin toisiinsa limittyneinä, ohuina kerroksina. Renkaan komponenttien tarkoitus on välittää voimia ajoneuvon ja tien välillä.



Kuva 2. *Renkaan komponentit (perustuu lähteeseen Nokian Renkaat Oyj)*

Kuvassa 2 on lueteltu renkaan tärkeimmät osat. Renkaan päällimmäisin osa on pinta tai pintaprofiili. Se siirtää voimia rungon komponenttien ja tien välillä. Pinta voi sisältää useita ohuita kerroksia erilaisia kumiseoksia. Pinnan päällimmäisin kumikerros synnyttää kitkavoimia ja on renkaan kuluva osa. Sen alla olevat kumiseokset auttavat päällimmäisen kerroksen toimintaa. Renkaan vahvikekuidut on ladottu kumitettuihin langoista muodostuviin koordeihin.

Heti pinnan alapuolella on ylimmäinen koordi nimeltä päällikerros (JLB). Se kääritään saumattomasti renkaan vierinkehän suuntaiseksi. JLB:n tarkoitus on hillitä renkaan halkaisijan kasvua suurilla nopeuksilla. JLB:n alapuolella on henkilöautonrenkaassa kaksi vastakkaisiin kulmiin ladottua teräsvyötä, jotka saavat aikaan pääosan renkaan pinta-alueen jäykkyydestä. Vahvikkeiden muodostama vyöpaketti kiinnitetään renkaan runkoon.

Runkokoordi on renkaan pyörimissuuntaan nähden kohtisuoraan ladottu kumitettu vahvikekuitu. Runkokoordeja voi olla renkaassa useita kerroksia. Runkokoordi kiertää renkaan

kummallakin jalalla kaapelin ja kolmioliuskan ja kääntyy takaisin kohti renkaan ulkohalkaisijaa.

Kaapeli painaa renkaan vannetta vasten, jolloin niiden liitoskohdasta tulee ilmatiivis. Kolmioliuska tukee kaapelia ja jäykistää rengasta. Rungon ulkopuolella on sivupinta. Se suojaa koordia kulumiselta ja myöskin jäykistää rengasta. Rungon sisäpuolella on sisäkumikerros innerliner, joka estää ilmaa suotamasta rungon läpi.

2.2 Renkaan laskentamalli

Renkaan laskentamallin käsittelyyn käytetään Abaqus 6.14 -ohjelmistoa. Laskentamallin luomiseen ja laskemiseen käytetään ohjelmiston ohjeita *Abaqus 6.14 Documentation* (2014) ja Ojalan diplomityössä kehitettyä lähestymistapaa (Ojala, 2003). Rengas on komponenttien saumoja lukuun ottamatta pyörähdyssymmetrinen kappale. Jos komponenttien saumoja ei oteta huomioon, renkaan laskentamallia voidaan käsitellä kaksiulotteisena.

Renkaasta muodostetaan ensin kaksiulotteinen pyörähdyssymmetrinen malli. Pyörähdyssymmetrinen malli koostuu neliö-, kolmio- ja viivaelementeistä. Renkaan kumikomponentit esitetään kaksiulotteisilla nelikulmio- ja kolmioelementeillä ja vahvikekuidut yksiulotteisilla viivaelementeillä. Viivaelementtien latominen paikoilleen on yksinkertaista, kun neliö- ja kolmioelementit on saatu paikoilleen. Mallia luotaessa vaikeinta onkin kartoittaa kaksiulotteiset kumielementit vastaamaan renkaan geometriaa.

2.3 Pyörähdyssymmetrisen mallin luominen

Jotta laskentamallin luominen voitaisiin automatisoida, on tärkeä ymmärtää, miten malli rakennetaan käsin. Todellinen rengas koostuu kumista ja vahvikkeista. Myös rengasmalli koostuu kiinteän kumin perusmateriaaliosasta ja vahvikekuitujen osasta. Kummallekin osalle tehdään pääpiirteissään samat toimenpiteet. Osan luominen voidaan jakaa geometrian luomiseen, verkottamiseen ja apumääreiden luomiseen.

2.3.1 Manuaalisen mallin geometria

Geometrian luominen alkaa geometrian siirrolla. Se tehdään tallentamalla rengasmallin rakennepiirustus IGES-formaattiin ja lukemalla tämä tiedosto Abaqus-ohjelmaan. Seuraavaksi rakennepiirustuksesta siivotaan tekstit ja viivat, joita ei käytetä. Rakennepiirustuksen käyrät jaetaan mallissa seuraaviin piirustuksiin:

1. pohjapiirustus
2. rajapintapiirustus
3. vahvikekuitupiirustus

Pohjapiirustukseen kuuluvat käyrät muodostavat renkaan ulko- ja sisäpinnat. Pohjapiirustuksen avulla luodaan pyörähdyssymmetrinen, tasomainen perusmateriaaliosa. Tämän

vuoksi käyrien tulee muodostaa suljettu ketju, joka ei leikkaa itseään. Rajapintapiirustukseen valitaan käyrät, jotka esittävät materiaalirajapintoja. Rajapintakäyrillä pohjaosa osioidaan, eli leikataan osiin, ja osat jaetaan joukkoihin. Näille joukoille asetetaan myöhemmin eri kumikomponenttien materiaaliarvot. Vahvikekuitupiirustuksen avulla luodaan vahvikekuituosa. Rakennemallissa vahvikekuitujen käyrät esittävät vahvikekuidun rajapintoja. Laskentamalliin vahvikekuitu halutaan näiden rajapintojen puoliväliin. Siksi vahvikekuitupiirustuksessa rajapinnasta otetaan vakiomitan pituinen siirtymä (offset), jonne uusi käyrä luodaan. Myös vahvikeosa jaetaan materiaalien mukaisiin joukkoihin.

Lopuksi siivotaan geometrian virheelliset osat. Jos viivoissa on ylimääräisiä pisteitä, ne pakottavat verkottajan luomaan solmupisteen näihin kohtiin. Tämä pakottaa elementtejä vääristymään. Myös lyhyet viivat muodostavat vääristyneitä elementtejä. Muut geometriavirheet tuodussa mallissa voivat aiheuttaa katkoksia tai aukkoja alueiden ja viivojen jatkuvuuteen. Yleensä mallissa on myös joitakin päällekkäisiä viivoja, jotka aiheuttavat satunnaista käytöstä. Nämä viivat täytyy poistaa. Viivat, jotka risteävät hyvin terävässä kulmassa, on hyvä katkaista jo kauempaa ja piirtää uudestaan tylpemmällä kulmalla. Muutoin ne aiheuttavat verkotuksessa teräviä kolmioita.

2.3.2 Manuaalinen verkottaminen

Verkottamisessa osioidut tasoalueet jaetaan laskennan käyttämiin elementteihin. Verkottamisen nopeuttamiseksi renkaasta mallinnetaan vain keskiosa sekä toinen jalka. Mallille annetaan yleinen verkon kokotavoite, jonka perusteella Abaqus-ohjlmena automaattinen verkottaja muodostaa ensimmäisen verkon. Tämä verkko on aina täynnä karkeita virheitä ja huonon muotoisia elementtejä. Verkon parantamiseksi Abaqus-ohjelman verkottajalle annetaan lisäohjeita.

Yleinen tapa parantaa verkkoa on pilkkoa verkko helpommin käsiteltäviin osiin. Malliin lisätään uusia, runkokoordin linjaan nähden kohtisuoria osiointeja. Nämä osioinnit pakottavat verkottajaa kääntämään ainakin yhden elementinsivun kulkemaan tällä kohdalla osioinnin suuntaisesti. Parhaita paikkoja näiden osiointien tekemiselle ovat epäjatkuvuuskohdat, joissa rajapintaviivat yhtyvät. Käyttäjä lisää alueille uusia osiointeja ja kokeilee solmulukumääriä ja eri verkotusalgoritmejä. Algoritmejä esitellään myöhemmässä osiossa. Kun yksi verkon alue näyttää tyydyttävältä, siirrytään seuraavaan alueeseen. Koska verkotusalgoritmejä, reunaviivoja ja mahdollisia solmumääriä on lukemattomasti, kokeiltavia vaihtoehtoja on paljon. Joskus yhden alueen viimeistely pakottaa korjaamaan myös valmiita alueita, jotta niiden reunaviivojen solmumäärät olisivat yhtenevät.

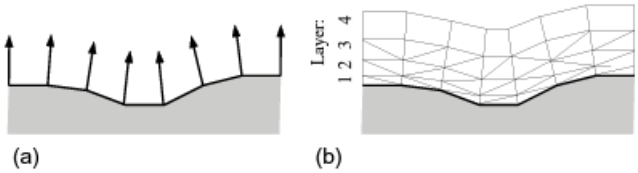
Lopuksi solids- ja fibers-verkoista luodaan orpoverkot. Orpoverkossa solmujen paikka irrotetaan geometriassa. Orpoverkosta luodaan peilattu kopio, joka liitetään alkuperäiseen. Näin saadaan kokonaisen renkaan poikkileikkaus. Fibers-verkolle verkottaja ei osaa päätellä elementtien oikeaa suuntaa. Sen elementtien normaalit käännetään samaan suuntaan käsin.

2.4 Verkotusalgoritmit

Alueen verkotusalgoritmi määrittää, miten alueen sisällä olevat solmut ja elementit jaetaan. Abaqus tuntee joukon erilaisia algoritmejä, mutta yleisesti verkotusalgoritmit voidaan jakaa säännöllisen verkon (structured) ja epäsäännöllisen verkon (unstructured) menetelmiin (Blazek, s. 360). Säännöllisen verkon menetelmissä verkon topologia esitetään matemaattisilla yhtälöillä. Näiden avulla verkko voidaan muodostaa systemaattisesti. Lisäksi solmujen ja elementtien luontainen paikkaan perustuva indeksointi nopeuttaa niiden laskemista. Epäsäännöllisen verkon menetelmissä solmujen ja elementtien muoto ja sijainti voidaan valita vapaasti, mikä tekee niistä joustavampia muuttuvan geometrian kuvaamiseen. Myös Abaqus tukee näitä molempia verkotustapoja, mutta kummankaan käyttö ei takaa toimivaa verkkoa.

Säännöllisen verkon menetelmillä muodostettu verkko toimii hyvin, mikäli verkotusalue on konveksi monikulmio. Rengasmallissa enemmistö verkon alueista on kuitenkin konkaaveja polygoneja. Näissä alueissa reunasolmut muodostavat C:n muotoisia kaaria. Yksinkertainen säännöllisen menetelmän verkottaja muodostaa elementtejä alueen onton keskialueen yli. Tällainen verkko ei voi toimia oikein. Tästä syystä säännöllisiä menetelmiä ei yleensä voi käyttää suurille alueille.

Epäsäännöllisen verkon menetelmissä verkottaja muodostaa alueen sisään kolmioverkon. Yleisesti käytetyssä Delayneyn algoritmissä verkottaja pyrkii maksimoimaan muodostuvien kolmioiden pienintä kulmaa (Blazek, s. 374). Toisessa yleisessä advancing front -algoritmissä verkottaja etenee reunasolmuista kohti alueen keskustaa pyrkien välttämään solmujen syntymistä elementtien sisään. Jos verkosta halutaan pääasiassa neliöverkko, vierekkäisiä kolmioita yhdistetään neliöiksi, missä se on mahdollista. Aiemmassa osiossa kuitenkin huomattiin, että neliöelementtien kannalta tärkein ominaisuus on elementtien sivujen kohtisuoruus. Yleiset epäsäännölliset menetelmät eivät kuitenkaan takaa tätä kohtisuoruutta. Tähän tarkoitukseen paremmin soveltuu advancing layers -algoritmi (Blazek, s. 381). Siinä verkottaja muodostaa kuvan 12a tapaan rajapinnan solmukohtiin normaalivektoreita. Kuvassa 12b näille vektoreille luodaan uusia uusia solmuja ja solmujen välille muodostetaan elementtejä, kunnes riittävä elementtikerrosten paksuus saavutetaan.



Kuva 3. *Advancing layers -algoritmin toimintaa (a) reunasolmujen normaalien määritys, (b) elementtien muodostaminen normaaleja pitkin. (Blazek, s. 381)*

Koska advancing layers -algoritmi käyttää normaalivektoreita, syntyvät elementit ovat luonnostaan kohtisuorassa rajapintaan nähden. Rengasmallin tapauksessa elementtejä pitää muodostaa kahden rajapinnan väliin. Tämä algoritmi ei kuulu Abaqus-ohjelman sisäänrakennettuihin verkotusalgoritmeihin. Myöhemmässä osiossa esitetään, miten tätä algoritmia voidaan soveltaa rengasmallille.

2.5 Mallin laskenta

Aksisymmetrisen mallin laskennassa mallia kuormitetaan todellista rengasta vastaavasti. Rengasmalli paineistetaan ja se asennetaan vanteelle. Vannetta esittää kaksi analyyttistä geometriaviivaa. Vanteelle asennus tehdään tuomalla erilliset kartiomaiset vanteen puolikkaat yhteen, jolloin renkaan jalkaosa puristuu vaadittavaan esikiristykseen. Aksisymmetrisen mallin laskenta on erittäin nopeaa. Tavallisesti sen laskenta-aika on alle minuutin.

Seuraavaksi renkaasta muodostetaan kolmiulotteinen malli. Aksisymmetrinen malli pyöräytetään keskiakselinsa ympäri. Pyöräytettäessä Abaqus monistaa kaksiulotteisista elementeistä kolmiulotteisia. Kolmiulotteisten elementtien sektorien tiheyttä voidaan tarvittaessa kasvattaa kohdissa, joihin halutaan lisää tarkkuutta. Tavanomaisesti tarkkuutta kasvatetaan etenkin renkaan footprintissä eli renkaan tiekontaktialueella. Tämän jälkeen pyöräytetystä mallista voidaan poistaa pinnasta elementtejä. Näin pintaan saadaan muodostettua renkaan poikittaisurat.

Ojalan lähestymistavassa kolmiulotteisen mallin laskennassa on kolme askelta (Ojala, 2003). Ensin malli saatetaan elementtimuutoksen jälkeen tasapainoon. Tämän jälkeen malli painetaan kontaktiin analyyttisen tien pinnan kanssa. Lopuksi renkaalle annetaan tavoitekuorma. Nämä askeleet ovat verrattain raskaita. Laskennassa kestää yleensä 1-8 tuntia nykyaikaisella laskentakoneella. Ojalan esityksessä laskenta-aika oli 10 tuntia. Elementtien ja solmujen määrän kasvaessa laskenta-aika pitenee, mutta suurin osa laskenta-ajasta kuluu aika-askeleiden ratkaisun iterointiin. Huonolaatuisen elementtiverkon laskenta ei välttämättä suppene. Tällöin Abaqus-ohjelman ratkaisija pienentää laskennan aika-askelta, jolloin laskenta-aika kasvaa. Jos aika-askel pienenee alle asetetun raja-arvon, ratkaisija pysäyttää laskennan ja simulointi epäonnistuu.

3. LASKENNAN ELEMENTTIEN MÄÄRITTELY

3.1 Elementtimenetelmän perusteoria

Tässä esityksessä pohjana käytetään teosta *Introduction to Finite Elements in Engineering* (Chandrupatla ja Belegundu, 2002). Abaqus käyttää laskennassaan mm. Flanaganin ja Belytchkon kehittämiä elementtejä. Merkkejä on täydennetty vastaamaan etenkin Belytschkon käyttämää notaatiota (Flanagan ja Belytchko, 1981; Belytchko ja Bindeman, 1993).

Elementtimenetelmässä keskitytään ratkaisemaan elementeiksi jaetun kappaleen voimien tasapainoyhtälöä. Kolmiulotteisessa avaruudessa tämä virtuaalisen työn yhtälö on (Chandrupatla ja Belegundu, s. 15):

$$\int_V \boldsymbol{\sigma}^T \, \delta\boldsymbol{\epsilon} dV - \int_V \delta\boldsymbol{u}^T \, \boldsymbol{f} dV - \int_S \delta\boldsymbol{u}^T \, \boldsymbol{T} dS - \sum_i \delta\boldsymbol{u}^T \boldsymbol{P} = 0, \qquad \forall \delta\boldsymbol{u} \in K \tag{1}$$

jossa $\delta\boldsymbol{u}$ on virtuaalinen siirtymä, $\delta\boldsymbol{\epsilon}$ virtuaalinen venymä ja K kinemaattisesti luvallisten virtuaalisten siirtymien joukko.

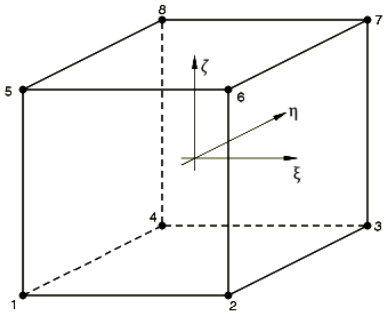
Ongelman tarkempaa ratkaisua on käsitelty ansiokkaasti muissa töissä (Ojala, 2003, s. 1; Lehtoranta, 2008, s. 13-28; Laitila, 2018, s. 7-9), joten tässä esityksessä keskitytään tarkastelemaan teorian vaikutuksia verkon muodostukseen.

3.1.1 Isoparametrinen elementti

Käytettyjen elementtien integrointi on määritetty isoparametristen elementtikoordinaattien ξ, η ja ζ funktioina. Tällaisen elementin siirtymäkenttä interpoloidaan (Chandrupatla ja Belegundu, 2002, s. 50)

$$\boldsymbol{u} = \boldsymbol{\phi q} \tag{2}$$

jossa **q** on elementin solmujen siirtymävektori ja $\boldsymbol{\phi}$ elementin solmun muotofunktiovektori.



Kuva 4. *Isoparametrinen, kolmiulotteinen nelikulmioelementti (perustuu lähteeseen Abaqus 6.14 Documentation, 2014)*

Kuvassa 4 näytetään kahdeksansolmuisen nelikulmioelementin isoparametrinen koordinaatisto, jossa käytetään isoparametrisiä koordinaatteja ξ, η ja ζ. Tällaisen elementin siirtymiksi saadaan u, v ja w. Näistä esimerkkinä x-suunnassa siirtymäksi saadaan (s.285)

$$u = N_1u_1 + N_2u_2 + N_3u_3 + N_4u_4 + N_5u_5 + N_6u_6 + N_7u_7 + N_8u_8, \tag{3}$$

jossa u_i ovat solmujen x-suuntaisia siirtymiä ja muotofunktiot ovat

$$N_1 = \frac{1}{8}(1-\xi)(1-\eta)(1-\zeta), \qquad N_2 = \frac{1}{8}(1+\xi)(1-\eta)(1-\zeta),$$

$$N_3 = \frac{1}{8}(1+\xi)(1+\eta)(1-\zeta), \qquad N_4 = \frac{1}{8}(1-\xi)(1+\eta)(1-\zeta),$$

$$N_5 = \frac{1}{8}(1-\xi)(1-\eta)(1+\zeta), \qquad N_6 = \frac{1}{8}(1+\xi)(1-\eta)(1+\zeta),$$

$$N_7 = \frac{1}{8}(1+\xi)(1+\eta)(1+\zeta), \qquad N_8 = \frac{1}{8}(1-\xi)(1+\eta)(1-\zeta).$$

Kaavan (3) tapaan saataisiin myös v ja w vaihtamalla solmusiirtymät. Elementin sisäisen pisteen koordinaatit voidaan esittää samojen muotofunktioiden avulla solmupisteiden koordinaattien summana. Tällöin esimerkiksi x-koordinaatiksi saadaan (s.285)

$$\mathrm{x} = N_1x_1 + N_2x_2 + N_3x_3 + N_4x_4 + N_5x_5 + N_6x_6 + N_7x_7 + N_8x_8, \tag{4}$$

jossa x_i ovat solmujen x-koordinaatteja. Kaavan 4 tapaan saataisiin myös y- ja z-koordinaatit vaihtamalla solmukoordinaatit.

3.1.2 Elementin venymät ja jännitykset

Jakobin matriisi välittää muunnoksen emoelementin koordinaattien ξ, η ja ζ ja rakennekoordinaattien x, y ja z välillä ja on muotoa (s. 278)

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix}. \tag{5}$$

Kaavasta (5) saadaan kääntämällä

$$\boldsymbol{A} = \boldsymbol{J}^{-1} = \frac{1}{\det \boldsymbol{J}} \begin{bmatrix} J_{22}J_{33} - J_{32}J_{23} & J_{32}J_{13} - J_{12}J_{33} & J_{12}J_{23} - J_{22}J_{13} \\ J_{23}J_{31} - J_{33}J_{21} & J_{33}J_{11} - J_{13}J_{31} & J_{13}J_{21} - J_{23}J_{11} \\ J_{21}J_{32} - J_{31}J_{22} & J_{31}J_{12} - J_{11}J_{32} & J_{11}J_{22} - J_{21}J_{12} \end{bmatrix} \tag{6}$$

Elementin venymäkenttä saadaan kaavalla (s. 275)

$$\boldsymbol{\epsilon} = \boldsymbol{Bq} = [\epsilon_x, \quad \epsilon_y, \quad \epsilon_z, \quad \gamma_{yz}, \quad \gamma_{xz}, \quad \gamma_{xy}]^T \tag{7}$$

jossa $\boldsymbol{B}$ on 6x12 gradienttimatriisi (s.279)

$$\boldsymbol{B} = \begin{bmatrix} A_{11} & 0 & 0 & A_{12} & 0 & 0 & A_{13} & 0 & 0 & -\widetilde{A_1} & 0 & 0 \\ 0 & A_{21} & 0 & 0 & A_{22} & 0 & 0 & A_{23} & 0 & 0 & -\widetilde{A_2} & 0 \\ 0 & 0 & A_{31} & 0 & 0 & A_{32} & 0 & 0 & A_{33} & 0 & 0 & -\widetilde{A_3} \\ 0 & A_{31} & A_{21} & 0 & A_{32} & A_{22} & 0 & A_{33} & A_{23} & 0 & -\widetilde{A_3} & -\widetilde{A_2} \\ A_{31} & 0 & A_{11} & A_{32} & 0 & A_{12} & A_{33} & 0 & A_{13} & -\widetilde{A_3} & 0 & -\widetilde{A_1} \\ A_{21} & A_{11} & 0 & A_{22} & A_{12} & 0 & A_{23} & A_{13} & 0 & -\widetilde{A_2} & -\widetilde{A_1} & 0 \end{bmatrix} \tag{8}$$

jossa A_{ij} on kaavan (6) matriisin tekijä, $\widetilde{A_1} = A_{11} + A_{12} + A_{13}$, $\widetilde{A_2} = A_{21} + A_{22} + A_{23}$ ja $\widetilde{A_3} = A_{31} + A_{32} + A_{33}$.

Elementin jännitykselle on voimassa kaavan (7) mukaan

$$\boldsymbol{\sigma} = \boldsymbol{D\epsilon} = \boldsymbol{DBq} \tag{9}$$

jossa $\boldsymbol{D}$ on symmetrinen materiaalimatriisi. Jos materiaalimalli on isotrooppinen ja noudattaa Hooken lakia (s.6)

$$\boldsymbol{D} = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5-\nu & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5-\nu & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5-\nu \end{bmatrix} \tag{10}$$

jossa E on kimmokerroin ja ν Poissonin vakio.

3.1.3 Muodonmuutosenergia

Elementin muodonmuutokseen vaadittu kimmoenergia saadaan integroimalla elementin tilavuuden yli (s. 279)

$$U_e = \frac{1}{2}\int_e \boldsymbol{\epsilon}^T \boldsymbol{D}\boldsymbol{\epsilon} dV. \tag{11}$$

Sijoittamalla kaavaan (11) kaava (7) ja siirtämällä vakiotermit integraalista ulos saadaan (s.213)

$$U_e = \frac{1}{2}\boldsymbol{q}^T\boldsymbol{B}^T\boldsymbol{D}\boldsymbol{B}\boldsymbol{q}\int_e dV = \frac{1}{2}\boldsymbol{q}^T V_e \boldsymbol{B}^T\boldsymbol{D}\boldsymbol{B}\boldsymbol{q} = \frac{1}{2}\boldsymbol{q}^T\boldsymbol{k}_e\boldsymbol{q} \tag{12}$$

jossa V_e on elementin tilavuus ja $\boldsymbol{k}_e$ elementin jäykkyysmatriisi.

$$\boldsymbol{k}^e = V_e\boldsymbol{B}^T\boldsymbol{D}\boldsymbol{B} \tag{13}$$

Kaava (13) voidaan yleistää koko kappaleelle laskemalla kappaleen elementtien jäykkyyksien summa

$$U = \sum_e \frac{1}{2}\boldsymbol{q}^T\boldsymbol{k}^e\boldsymbol{q} = \frac{1}{2}\boldsymbol{Q}^T\boldsymbol{K}\boldsymbol{Q} \tag{14}$$

jossa **Q** on koko rakenteen kaikkien vapausasteiden siirtymävektori ja **K** on jäykkyysmatriisi.

3.1.4 Aksisymmetrinen elementti

Rengasmallin laskennassa käytetään usein kolmiulotteista elementtiä laskennallisesti kevyempää aksisymmetristä esitystapaa. Siinä elementti kuvataan sylinterikoordinaatistossa, jolloin kaavojen (7) ja (9) mukaiset venymät ja jännitykset ovat (s.179)

$$\boldsymbol{\epsilon} = [\epsilon_r, \quad \epsilon_z, \quad \gamma_{rz}, \quad \epsilon_\theta]^T, \qquad \boldsymbol{\sigma} = [\sigma_r, \quad \sigma_z, \quad \tau_{rz}, \quad \sigma_\theta]^T \tag{15}$$

jossa r on elementin säteen, z pystykoodinaatin ja θ pyörähdyskulman suunta. Kaavasta (15) saadaan johdettua kaavaa (10) muistuttava materiaalimatriisi (s.180)

$$\boldsymbol{D} = \frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix} 1-\nu & \nu & 0 & \nu \\ \nu & 1-\nu & 0 & \nu \\ 0 & 0 & 0.5-\nu & 0 \\ \nu & \nu & 0 & 1-\nu \end{bmatrix} \tag{16}$$

Kaavan (13) tapaan elementin jäykkyydeksi saadaan

$$\boldsymbol{k}^e = 2\pi \int_e \boldsymbol{B}^T \boldsymbol{D}\boldsymbol{B} r \, dA \tag{17}$$

Kaavojen (15), (16) ja (17) tapaan voitaisiin aksisymmetrisille elementeillä johtaa muutkin aiemmin esitetyt kaavat. Aksisymmetrisiä elementtejä käytetään myöhemmin laskennassa, mutta niiden kuvaukseen ei tässä työssä syvennytä enempää.

3.1.5 Numeerinen integrointi

Edellä kaavoissa muotofunktiot ja koordinaatit esitetään jatkuvina koko elementin alueella. Käytännössä funktioiden integraalit lasketaan numeerisesti vain rajatuissa pisteissä. Kaksiulotteisella, ensimmäisen asteen nelikulmioelementillä näitä integrointipisteitä on neljä. Pisteinä käytetään kahta Gaussin pistettä, jolloin niiden avulla saadaan laskettavan funktion integraalin numeerinen likiarvo (Chandrupatla ja Belegundu, 2002, s. 216)

$$\int_{-1}^{1} f(g)\, d\xi \approx \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} w_i f(\xi_i) \tag{18}$$

jossa f on integroitava funktio, ξ esimerkkinä käytettävä ensimmäinen isoparametrinen koordinaatti, n Gaussin pisteiden lukumäärä, w_i i:nnen pisteen painokerroin ja ξ_i i:nnen pisteen isoparametrinen koordinaatti. Täysin integroidulla neliöelementillä painokertoimet ovat $w_1 = w_2 = 1.0$ ja isoparametriset koordinaatit ovat $\xi_i = \eta_i = \pm\frac{1}{\sqrt{3}}$.

Elementtien esitystapa aiheuttaa laskentaan virheitä (Abaqus 6.14 Documentation, 2014). Integrointipisteiden sijainnista ja muotofunktioiden tyypistä johtuen ensimmäisen asteen neliöelementti ei pysty esittämään tarkasti kaikkia kuormitustiloja. Jos elementti on puhtaassa taivutuksessa, integrointipisteisiin muodostuu keinotekoinen leikkausjännitys. Tästä jännityksestä johtuen elementti on liian jäykkä todelliseen kuormitukseen nähden ja elementin sanotaan leikkauslukittuvan. Elementin lukittuminen johtuu Hueckin mukaan siitä, että elementin ominaisvektorit eivät ole toisistaan riippumattomia (Hueck, ym., 1994, s.7). Kun elementin pituussuhde kasvaa, elementin molemmat taivutusmuodot kasvattavat jäykkyyttään.

Täysin integroitu elementti ei myöskään toimi kaikilla materiaalimalleilla. Kun materiaalin Poissonin vakio lähestyy kokoonpuristumattoman materiaalin rajaa 0.5, elementin jäykkyys kasvaa äärettömäksi. Tässä työssä kumin Poissonin vakiona käytetään arvoa 0.48, jolloin täysin integroidulla elementillä tilavuuslukittuminen alkaa jo nostaa elementin jäykkyyttä. Tämän ongelman ratkaisemiseksi Abaqus käyttää ensimmäisen asteen nelikulmioelementeillä ja tiilielementeillä tilavuustekijöiden laskennassa aina osittain ali-integroitua muotoa, joka estää tilavuuslukittumisen. Tätä kutsutaan valikoivasti ali-integroiduksi (Abaqus 6.14 Documentation, 2014). Sen lisäksi elementit voidaan laskea täysin ali-integroituna, jota käsitellään seuraavana.

3.1.6 Elementin ali-integroitu muoto

Lukittumisen estämiseksi nelikulmioelementeistä voidaan käyttää ali-integroitua muotoa. Ali-integroidussa muodossa elementille käytetään kaavassa (18) astetta liian pientä Gaussin pisteiden määrää. Ensimmäisen asteen tasoneliöelementille käytetään ali-integroidussa muodossa yhtä integrointipistettä. Abaqus-ohjelmassa integrointipisteenä voidaan käyttää elementin keskipistettä tai keskiarvoa elementin yli. Elementin keskipistemuodossa käytetään Gaussin integraalissa painokerrointa $w_1 = 2$ ja isoparametrisiä koordinaatteja $\xi_i = \eta_i = \zeta_i = 0$. Tällöin kaavassa (8) esiintyvä gradienttimatriisi saadaan helposti Chandrupatla ja Belegundu, 2001, S.219)

$$\boldsymbol{B}^o = \frac{1}{4}\begin{bmatrix} -1 & 0 & 1 & 0 & 1 & 0 & -1 & 0 \\ 0 & -2 & 0 & -2 & 0 & 2 & 0 & 2 \\ -2 & -1 & -2 & 1 & 2 & 1 & 2 & -1 \end{bmatrix}. \tag{19}$$

Yleensä käytetään kuitenkin laskennallisesti raskaampaa keskiarvoa elementin yli, jolloin muotofunktioista otetaan integraali elementin yli (Flanagan ja Belytchko, 1981, s.682)

$$\boldsymbol{B_{iI}} = \int_V \frac{\partial \boldsymbol{\phi}^I}{\partial x_i} dV \tag{20}$$

Keskiarvomuoto on laskennallisesti työläämpi kuin keskipistemuoto, mutta kuitenkin kevyempi kuin täysin integroitu muoto. Keskiarvomuotoa käytetään, koska se tarjoaa sopivan tasapainon laskennallisen tarkkuuden ja kustannusten välillä. (Flanagan ja Belytchko, 1981).

Vaikka ali-integroitu muoto vähentääkin elementin leikkaus- ja tilavuuslukittumista, se synnyttää elementeille uusia, kinemaattisia deformaatiomuotoja. Näiden deformaatiomuotojen syntymiseen ei tarvita energiaa tai ulkoisia voimia, minkä vuoksi ne voivat nopeasti aiheuttaa suuria, virheellisiä solmusiirtymiä, jotka voivat pilata laskennan tulokset. Neliöelementeillä nämä deformaatiomuodot saavat elementit näyttämään tiimalasilta, minkä vuoksi niitä kutsutaan tiimalasimuodoiksi. Tämän ongelman hallitsemiseksi Abaqus-ohjelmassa on käytössä useita erilaisia tiimalasikontrollitapoja. Niistä seuraavana esitellään Flanaganin ja Belytchkon ehdottama ortogonaalinen stabilointi ja myöhemmin Flanaganin ja Bindemanin ehdottama parannettu tiimalasikontrolli.

3.2 Ortogonaalinen tiimalasistabilointi

Flanaganin ja Belytchkon mukaan ortogonaalisessa tiimalasikontrollissa ali-integroidun suorakulmaisen särmiöelementin kaavan (2) mukainen muotofunktio pilkotaan toisistaan riippumattomiin osiin. Tässä esitetään muotofunktio x-koordinaatin suuntaan, mutta se voitaisiin esittää samoin myös y- ja z-suuntiin:

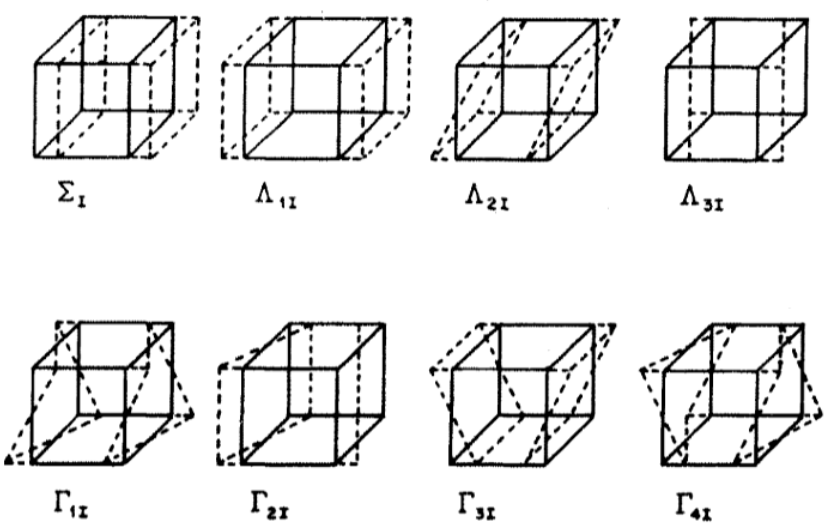
$$\boldsymbol{\phi}^I = \frac{1}{8}\boldsymbol{\Sigma}^I + \frac{1}{4}\xi\boldsymbol{\Lambda}_1^I + \frac{1}{4}\eta\boldsymbol{\Lambda}_2^I + \frac{1}{4}\zeta\boldsymbol{\Lambda}_3^I + \frac{1}{4}\eta\zeta\boldsymbol{\Gamma}_1^I + \frac{1}{4}\zeta\xi\boldsymbol{\Gamma}_2^I + \frac{1}{4}\xi\eta\boldsymbol{\Gamma}_3^I + \frac{1}{4}\xi\eta\zeta\boldsymbol{\Gamma}_4^I \qquad (21)$$

Kaavassa (21) muotofunktiovektori on auki kirjoitettuna. Tästä muodosta saadaan solmujen perusvektorit, jotka on lueteltu taulukossa 1.

Taulukko 1. *Elementin muotofunktion perusvektorit*

Solmu	ξ	η	ζ	$\boldsymbol{\Sigma}^I$	$\boldsymbol{\Lambda}_1^I$	$\boldsymbol{\Lambda}_2^I$	$\boldsymbol{\Lambda}_3^I$	$\boldsymbol{\Gamma}_1^I$	$\boldsymbol{\Gamma}_2^I$	$\boldsymbol{\Gamma}_3^I$	$\boldsymbol{\Gamma}_4^I$
1	-½	-½	-½	1	-1	-1	-1	1	1	1	-1
2	½	-½	-½	1	1	-1	-1	1	-1	-1	1
3	½	½	-½	1	1	1	-1	-1	-1	1	-1
4	-½	½	-½	1	-1	1	-1	-1	1	-1	1
5	-½	-½	½	1	-1	-1	1	-1	-1	1	1
6	½	-½	½	1	1	-1	1	-1	1	-1	-1
7	½	½	½	1	1	1	1	1	1	1	1
8	-½	½	½	1	-1	1	1	1	-1	-1	-1

Taulukossa 1 esitetyt perusvektorit näytetään visuaalisesti kuvassa 5. Vektoreista $\boldsymbol{\Sigma}^I$ kuvaa elementin vakiosiirtymää, $\boldsymbol{\Lambda}_1^I$ elementin venymää, $\boldsymbol{\Lambda}_2^I$ ja $\boldsymbol{\Lambda}_3^I$ elementin taipumia ja $\boldsymbol{\Gamma}_\alpha^I$ tiimalasimuodonmuutoksia.



Kuva 5. *Suorakulmaisen särmiöelementin deformaatiomuodot x-suuntaiselle siirtymälle (Flanagan ja Belytchko, 1981, s. 682)*

Kontrolloinnissa interpolointifunktion tiimalasiperusvektori (21) korvataan tiimalasimuotovektorilla:

$$\boldsymbol{\Upsilon}_\alpha^I = \boldsymbol{\Gamma}_\alpha^I - \frac{1}{V}\boldsymbol{B}_i^I \boldsymbol{x}_i^J \boldsymbol{\Gamma}_\alpha^J \qquad (22)$$

jossa *I* ja *J* viittaavat tarkasteltavan solmun indeksiin. Flanaganin ja Belytchkon mukaan muotovektori on ortogonaalinen kaikkiin muihin perusvektoreihin nähden, jos elementti on suuntaissärmiö, eli kaikki sen sivut ovat suunnikkaita. Tällöin tiimalasikontrolli ei vaikuta elementin tarkkuuteen. Vastaavasti kaksiulotteisen elementin tulee olla suunnikas. Elementin jäykkyys voidaan laskea elastisena kaavalla:

$$\dot{Q}_i = k_e t \frac{\lambda + 2\mu}{2} \frac{\boldsymbol{B}_i^I \boldsymbol{B}_i^I}{A} \dot{q}_i \tag{23}$$

jossa λ ja μ ovat materiaalin Lamén parametrit. Kimmoenergia saadaan kaavasta (23) integroimalla. Lopulta tiimalasitiloja vastustavat solmuvoimat saadaan laskettua kaavojen (22) ja (23) tulona:

$$\boldsymbol{f}_{i,\alpha}^{stab} = \frac{1}{\sqrt{8}} Q_i \boldsymbol{\Upsilon}_\alpha^I \tag{24}$$

Kaavassa (24) esitetyt ortogonaaliset solmuvoimat saavat mallin yleensä stabiloitumaan. Vaimennus- ja kimmokertoimien valinta on kuitenkin vaikeaa etenkin karkealla verkolla puhtaan taivutuksen tapauksissa (Belytchko ja Bindeman, 1993). Lisäksi kokoonpuristumattomilla materiaaleilla elementit voivat lukittautua stabilointiparametrien kasvaessa. Näiden ongelmien ratkaisemiseksi Belytchko ja Bindeman esittivät elementtien oletetun venymäkentän stabilointia, jota kutsutaan Abaqus-ohjelmassa parannetuksi tiimalasikontrolliksi (enhanced hourglass control).

3.3 Parannettu tiimalasikontrolli

Belythckon ja Bindemanin lähestymistavassa kolmiulotteisen solmun siirtymäkenttä määritetään mielivaltaisten a- ja c-vakioiden avulla (Belytchko ja Bindeman, 1993, s.229)

$$u_{\boldsymbol{i}} = a_{0i} + a_{1i}x_1 + a_{2i}x_2 + a_{3i}x_3 + c_{1i}h_1 + c_{2i}h_2 + c_{3i}h_3 + c_{4i}h_4 \tag{25}$$

jossa h-kertoimet muodostetaan isoparametrisista koordinaateista ξ, η ja ζ

$$h_1 = \eta\zeta,\ h_2 = \zeta\xi, h_3 = \xi\eta, h_4 = \xi\eta\zeta. \tag{26}$$

Tarkastelemalla kaavaa (25) elementin solmuissa saadaan:

$$\boldsymbol{d}_{\boldsymbol{i}} = a_{0i}\boldsymbol{s} + a_{1i}\boldsymbol{x}_1 + a_{2i}\boldsymbol{x}_2 + a_{3i}\boldsymbol{x}_3 + c_{1i}\boldsymbol{h}_1 + c_{2i}\boldsymbol{h}_2 + c_{3i}\boldsymbol{h}_3 + c_{4i}\boldsymbol{h}_4 \tag{27}$$

Kaavassa (27) vektorit $\boldsymbol{x}_i$ ja $\boldsymbol{d}_{\boldsymbol{i}}$ ovat elementin i:nnen solmun koordinaatit ja solmusiirtymät. Vektorit $\boldsymbol{h}_i$ ovat kolmiulotteisen elementin tiimalasivektoreita (s.229)

$$\begin{aligned} \boldsymbol{h}_1^t &= (+1, +1, -1, -1, -1 - 1, +1, +1) \\ \boldsymbol{h}_2^t &= (+1, -1, -1, +1, -1 + 1, +1, -1) \end{aligned} \tag{28}$$

$$\boldsymbol{h}_3^t = (+1,-1,+1,-1,+1-1,+1,-1)$$

$$\boldsymbol{h}_4^t = (-1,+1,-1,+1,+1-1,+1,-1)$$

Kinemaattisen matriisin tekijät saadaan keskiarvomuotoiselle elementille kaavan (20) tapaan (s.231)

$$\widehat{\boldsymbol{b}}_i^t = \frac{1}{V}\int_{\Omega_e} \boldsymbol{\phi}_i d\Omega \tag{29}$$

jossa Ω on elementin alue. Kaavojen (28) ja (29) tekijöitä yhdistelemällä saadaan kaavaa (22) muistuttava keskiarvomuotoinen tiimalasimuotovektori (s.231)

$$\widehat{\boldsymbol{\gamma}}_\alpha = \frac{1}{8}\left[\boldsymbol{h}_\alpha - (\boldsymbol{h}_\alpha^t \boldsymbol{x}_h)\widehat{\boldsymbol{b}}_j\right] \tag{30}$$

jossa α esittää tiimalasivektorien indeksiä. Kohtisuoruusehdoista johtuen kaava (25) voidaan derivoida kaavojen (29) ja (30) avulla muotoon (s.231)

$$\boldsymbol{u}_{i,j} = \left(\widehat{\boldsymbol{b}}_j^t + \boldsymbol{h}_{\alpha,j}\widehat{\boldsymbol{\gamma}}_\alpha^t\right)\boldsymbol{d}_i \tag{31}$$

jossa $\boldsymbol{u}_{i,j}$ on solmujen i-koordinaatin siirtymäkenttä ja $\boldsymbol{h}_{\alpha,j}$ indeksin mukainen tiimalasivektori, kumpikin osittaisderivoituna j-koordinaatin suhteen. Kaavan (31) avulla voidaan muodostaa suorakulmaisen särmiön keskiarvomuotoinen siirtymägradientti, joka on 6 x 24-matriisimuodossa (s.231)

$$\boldsymbol{\nabla}_s\boldsymbol{u} = \widehat{\boldsymbol{B}}\boldsymbol{d} = \begin{bmatrix} \widehat{\boldsymbol{b}}_x^t + \boldsymbol{h}_{\alpha,x}\widehat{\boldsymbol{\gamma}}_\alpha^t & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \widehat{\boldsymbol{b}}_y^t + \boldsymbol{h}_{\alpha,y}\widehat{\boldsymbol{\gamma}}_\alpha^t & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \widehat{\boldsymbol{b}}_z^t + \boldsymbol{h}_{\alpha,z}\widehat{\boldsymbol{\gamma}}_\alpha^t \\ \widehat{\boldsymbol{b}}_y^t\boldsymbol{h}_{\alpha,y}\widehat{\boldsymbol{\gamma}}_\alpha^t & \widehat{\boldsymbol{b}}_x^t + \boldsymbol{h}_{\alpha,x}\widehat{\boldsymbol{\gamma}}_\alpha^t & \boldsymbol{0} \\ \widehat{\boldsymbol{b}}_z^t\boldsymbol{h}_{\alpha,z}\widehat{\boldsymbol{\gamma}}_\alpha^t & \boldsymbol{0} & \widehat{\boldsymbol{b}}_x^t + \boldsymbol{h}_{\alpha,x}\widehat{\boldsymbol{\gamma}}_\alpha^t \\ \boldsymbol{0} & \widehat{\boldsymbol{b}}_z^t + \boldsymbol{h}_{\alpha,z}\widehat{\boldsymbol{\gamma}}_\alpha^t & \widehat{\boldsymbol{b}}_y^t + \boldsymbol{h}_{\alpha,y}\widehat{\boldsymbol{\gamma}}_\alpha^t \end{bmatrix}\begin{bmatrix} \boldsymbol{d}_x \\ \boldsymbol{d}_y \\ \boldsymbol{d}_z \end{bmatrix} \tag{32}$$

Kaavassa (32) esiintyvä $\widehat{\boldsymbol{B}}$-matriisi voidaan tehokkuuden vuoksi jakaa vakio- ja muuttujaosiin (s.233)

$$\widehat{\boldsymbol{B}} = \widehat{\boldsymbol{B}}_c + \widehat{\boldsymbol{B}}_n = \begin{bmatrix} \widehat{\boldsymbol{b}}_x^t & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \widehat{\boldsymbol{b}}_y^t & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \widehat{\boldsymbol{b}}_z^t \\ \boldsymbol{0} & \widehat{\boldsymbol{b}}_z^t & \widehat{\boldsymbol{b}}_y^t \\ \widehat{\boldsymbol{b}}_z^t & \boldsymbol{0} & \widehat{\boldsymbol{b}}_x^t \\ \widehat{\boldsymbol{b}}_y^t & \widehat{\boldsymbol{b}}_x^t & \boldsymbol{0} \end{bmatrix} + \begin{bmatrix} \widehat{\boldsymbol{X}}_{1234}^t & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \widehat{\boldsymbol{Y}}_{1234}^t & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \widehat{\boldsymbol{Z}}_{1234}^t \\ \widehat{\boldsymbol{Y}}_{1234}^t & \widehat{\boldsymbol{X}}_{1234}^t & \boldsymbol{0} \\ \widehat{\boldsymbol{Z}}_{1234}^t & \boldsymbol{0} & \widehat{\boldsymbol{X}}_{1234}^t \\ \boldsymbol{0} & \widehat{\boldsymbol{Z}}_{1234}^t & \widehat{\boldsymbol{Y}}_{1234}^t \end{bmatrix} \tag{33}$$

jossa

$$\widehat{\boldsymbol{X}}_{1234} \equiv \sum_{\boldsymbol{\alpha=1}}^{\boldsymbol{4}} \boldsymbol{h}_{\alpha,x}\widehat{\boldsymbol{\gamma}}_{\alpha}\,, \qquad \widehat{\boldsymbol{Y}}_{1234} \equiv \sum_{\boldsymbol{\alpha=1}}^{\boldsymbol{4}} \boldsymbol{h}_{\alpha,y}\widehat{\boldsymbol{\gamma}}_{\alpha}\,, \qquad \widehat{\boldsymbol{Z}}_{1234} \equiv \sum_{\boldsymbol{\alpha=1}}^{\boldsymbol{4}} \boldsymbol{h}_{\alpha,z}\widehat{\boldsymbol{\gamma}}_{\alpha} \tag{34}$$

kaavan (34) notaatiota jatketaan siten, että

$$\widehat{\boldsymbol{X}}_{24} \equiv \boldsymbol{h}_{2,x}\widehat{\boldsymbol{\gamma}}_{2} + \boldsymbol{h}_{4,x}\widehat{\boldsymbol{\gamma}}_{4}, \qquad \widehat{\boldsymbol{X}}_{2} \equiv \boldsymbol{h}_{2,x}\widehat{\boldsymbol{\gamma}}_{2} \tag{35}$$

Belytchkon ja Bindemanin lähestymistavassa kaavan (33) muuttujaosaa parannellaan, jotta saadaan muodostettua ASQBI-venymäkenttä (assumed strain quintessential bending incompressible) (s.233)

$$\overline{\boldsymbol{B}}_{n} = \begin{bmatrix} \widehat{\boldsymbol{X}}^{t}_{1234} & -\bar{\nu}\widehat{\boldsymbol{Y}}^{t}_{3} - \nu\widehat{\boldsymbol{Y}}^{t}_{24} & -\bar{\nu}\widehat{\boldsymbol{Z}}^{t}_{2} - \nu\widehat{\boldsymbol{Z}}^{t}_{34} \\ -\bar{\nu}\widehat{\boldsymbol{X}}^{t}_{3} - \nu\widehat{\boldsymbol{X}}^{t}_{14} & \widehat{\boldsymbol{Y}}^{t}_{1234} & -\bar{\nu}\widehat{\boldsymbol{Z}}^{t}_{1} - \nu\widehat{\boldsymbol{Z}}^{t}_{34} \\ -\bar{\nu}\widehat{\boldsymbol{X}}^{t}_{2} - \nu\widehat{\boldsymbol{X}}^{t}_{14} & -\bar{\nu}\widehat{\boldsymbol{Y}}^{t}_{1} - \nu\widehat{\boldsymbol{Y}}^{t}_{24} & \widehat{\boldsymbol{Z}}^{t}_{1234} \\ \widehat{\boldsymbol{Y}}^{t}_{12} & \widehat{\boldsymbol{X}}^{t}_{12} & \boldsymbol{0} \\ \widehat{\boldsymbol{Z}}^{t}_{13} & \boldsymbol{0} & \widehat{\boldsymbol{X}}^{t}_{13} \\ \boldsymbol{0} & \widehat{\boldsymbol{Z}}^{t}_{23} & \widehat{\boldsymbol{Y}}^{t}_{23} \end{bmatrix} \tag{36}$$

jossa $\bar{\nu} \equiv \frac{\nu}{1-\nu}$. Kaavan (33) muuttujaosan matriisin toisen ja kolmannen rivin ensimmäiseen sarakkeeseen on lisätty $\widehat{\boldsymbol{X}}$-tekijöitä, jotta muodostuva venymäkenttä olisi isokoorinen. Tämä estää elementin volumetrisen lukkiutumisen, kun Poissonin vakio lähestyy kokoonpuristumatonta. Samalla $\widehat{\boldsymbol{X}}_{\boldsymbol{2}}$- ja $\widehat{\boldsymbol{X}}_{\boldsymbol{3}}$-tekijät ottavat vastaan poikittaisia venymiä, mikä ehkäisee leikkauslukkiutumista taivutuksessa. Vastaavasti $\widehat{\boldsymbol{Y}}_{\boldsymbol{34}}$- ja $\widehat{\boldsymbol{Z}}_{\boldsymbol{24}}$-tekijät poistetaan rivien neljä ja viisi ensimmäisestä sarakkeesta ylimääräisen leikkausjännityksen poistamiseksi puhtaassa taivutuksessa. Tekijöitä muokataan, kunnes lopulta kaavan (36) paranneltu matriisi pystyy esittämään 18 toisistaan riippumatonta deformaatiomuotoa. Oletettavasti näiden parannusten vuoksi Abaqus-ohjelmassa tätä stabilointitapaa kutsutaan lyhyesti enhanced-stabiloinniksi. Matriisin avulla voidaan muodostaa kaavassa (13) esitetty elementin jäykkyys (s.235)

$$\boldsymbol{k}^{e} = \boldsymbol{k}^{c} + \boldsymbol{k}^{stab} = \mathrm{V}\widehat{\boldsymbol{B}}^{t}_{c}\boldsymbol{C}\widehat{\boldsymbol{B}}_{\boldsymbol{c}} + \int_{\boldsymbol{\Omega_e}} \overline{\boldsymbol{B}}^{t}_{\boldsymbol{n}}\boldsymbol{C}\overline{\boldsymbol{B}}_{\boldsymbol{n}}d\Omega \tag{37}$$

jossa $\boldsymbol{k}^{c}$ on jäykkyyden vakio-osa ja $\boldsymbol{k}^{stab}$ stabilointiosa. Kaavassa (37) esiintyvä stabilointimatriisi voidaan antaa myös muodossa (s.236)

$$\boldsymbol{k}^{stab} = 2\mathrm{G}\begin{bmatrix} \boldsymbol{k}_{11} & \boldsymbol{k}_{12} & \boldsymbol{k}_{13} \\ \boldsymbol{k}_{21} & \boldsymbol{k}_{22} & \boldsymbol{k}_{23} \\ \boldsymbol{k}_{31} & \boldsymbol{k}_{32} & \boldsymbol{k}_{33} \end{bmatrix}, \tag{38}$$

jossa G on leikkausmoduuli ja $\boldsymbol{k}_{ii}$ ja $\boldsymbol{k}_{ij}$ jäykkyystekijöiden vektoreita. Jäykkyysvektorit saadaan taulukon 2 mukaisesti permutoimalla indeksejä i, j ja k. Indeksi k esiintyy myöhemmin kaavoissa (41)-(44) ja (46).

Taulukko 2. *Stabilointimatriisin indeksien permutaatiot*

i	1	1	2	2	3	3
j	2	3	3	1	1	2
k	3	2	1	3	2	1

Koska avaruuden koordinaatisto kääntyy elementin mukana, koordinaatistojen välille voidaan olettaa yhteys

$$\frac{\partial x_i}{\partial \xi_i} = \frac{1}{\partial \xi_i / \partial x_i} = \frac{1}{8}\widehat{\boldsymbol{\Lambda}}_i^t \boldsymbol{x}_i, \qquad \frac{\partial x_i}{\partial \xi_j} = \frac{\partial \xi_j}{\partial x_i} = 0, \qquad jos\ i \neq j \tag{39}$$

jossa

$$\begin{aligned} \widehat{\boldsymbol{\Lambda}}_1^t &= (-1,+1,+1,-1,-1,+1,+1,-1), \\ \widehat{\boldsymbol{\Lambda}}_2^t &= (-1,-1,+1,+1,-1,-1,+1,+1), \\ \widehat{\boldsymbol{\Lambda}}_3^t &= (-1,-1,-1,-1,+1,+1,+1,+1). \end{aligned} \tag{40}$$

Kaavassa (40) esitettyjen vektorien, kaavan (30) ja taulukon 2 permutaatioiden avulla saadaan lopulta kaavan (38) matriisin tekijät (s.236)

$$\boldsymbol{k_{ii}} = \boldsymbol{H}_{ii}\left[\frac{1}{1-\nu}\left(\widehat{\boldsymbol{\gamma}}_j\widehat{\boldsymbol{\gamma}}_j^t + \widehat{\boldsymbol{\gamma}}_k\widehat{\boldsymbol{\gamma}}_k^t\right) + \frac{1+\nu}{3}\widehat{\boldsymbol{\gamma}}_4\widehat{\boldsymbol{\gamma}}_4^t + \frac{1}{2}\left(\boldsymbol{H}_{ii} + \boldsymbol{H}_{jj}\right)\widehat{\boldsymbol{\gamma}}_i\widehat{\boldsymbol{\gamma}}_i^t\right], \tag{41}$$

$$\boldsymbol{k}_{ij} = \boldsymbol{H}_{ij}\left(\nu\widehat{\boldsymbol{\gamma}}_j\widehat{\boldsymbol{\gamma}}_i^t + \frac{1}{2}\widehat{\boldsymbol{\gamma}}_i\widehat{\boldsymbol{\gamma}}_{kj}^t\right), \tag{42}$$

jossa

$$\boldsymbol{H}_{ii} = \frac{1}{3}\frac{\left(\widehat{\boldsymbol{\Lambda}}_j^t\boldsymbol{x}_j\right)\left(\widehat{\boldsymbol{\Lambda}}_k^t\boldsymbol{x}_k\right)}{\left(\widehat{\boldsymbol{\Lambda}}_i^t\boldsymbol{x}_i\right)}, \tag{43}$$

$$\boldsymbol{H}_{ij} = \frac{1}{3}\left(\widehat{\boldsymbol{\Lambda}}_k^t\boldsymbol{x}_k\right). \tag{44}$$

Lopulta elementin stabilointivoima saadaan kaavasta (s.237)

$$\boldsymbol{f}_i^{stab} = \sum_{\alpha=1}^{4}\boldsymbol{Q}_{i\alpha}\widehat{\boldsymbol{\gamma}}_\alpha^t \tag{45}$$

jossa i on voimavektorin i:s tekijä ja Q_{ij} saadaan integroimalla kaavoista (s.237)

$$\dot{\boldsymbol{Q}}_{ii} = \mathrm{G}\left[\left(\boldsymbol{H}_{jj} + \boldsymbol{H}_{kk}\right)\dot{\boldsymbol{q}}_{ii} + \boldsymbol{H}_{ij}\dot{\boldsymbol{q}}_{jj} + \boldsymbol{H}_{ik}\dot{\boldsymbol{q}}_{kk}\right], \tag{46}$$

$$\dot{\boldsymbol{Q}}_{ij} = 2\mu\left[\frac{1}{1-\nu}\boldsymbol{H}_{ii}\dot{\boldsymbol{q}}_{ij} + \bar{\nu}\boldsymbol{H}_{ij}\dot{\boldsymbol{q}}_{ji}\right], \tag{47}$$

$$\dot{\boldsymbol{Q}}_{i4} = 2\mu\frac{1+\nu}{3}\boldsymbol{H}_{ii}\dot{\boldsymbol{q}}_{i4} \tag{48}$$

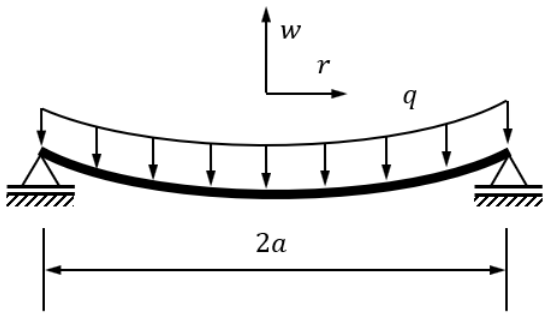
jossa $\boldsymbol{H}_{ii}$ ja $\boldsymbol{H}_{ij}$ ovat kaavoista (43) ja (44) saatavia tekijöitä ja $\dot{\boldsymbol{q}}_{ij}$ solmun j:nnen solmun i:n koordinaatin siirtymänopeus.

4. STABILOINNIN VAIKUTUS

Niin kuin edellä esitetystä yhteenvedosta huomataan, tiimalasikontrollin tarkan toiminnan kuvaus on vaikeaa. Kontrollin toiminta riippuu käytetystä elementistä, kuormituksesta, materiaalista ja jopa ratkaisijan numeerisista pyöristyksistä. Todellisen toiminnan selvittämiseksi numeerista ratkaisua verrataan tässä analyyttiseen ratkaisuun.

Laskennassa käytettävät elementit ovat aksisymmetrisiä, joten tarkasteltavan analyyttisen tilanteen pitää myös olla pyörähdyssymmetrinen. Yllä esitetyn perusteella parannetulle tiimalasikontollille pahin mahdollinen tapaus on (lähes) kokoonpuristumattoman kappaleen taivutus. Tarkastelemalla ohuen kumilevyn taipumaa voidaan arvioida elementtien toimintaa rengasmallin ohuissa kerroksissa.

4.1 Analyyttisen testikappaleen käsittely



Kuva 6. *Pyörähdyssymetrisen laatan taipuma*

Eräs tällainen kappale on esitetty kuvassa 6. Se on pyörähdyssymetrinen, yksinkertaisesti reunalta tuettu laatta, jonka reuna pääsee vapaasti liikkumaan säteen suunnassa. Laattaa kuormitetaan tasaisella painekuormalla q. Laatan säde on a, paksuus h ja materiaalin kimmokerroin E. Laatan taipuma w on reunalla 0 ja levyn keskipisteessä w_0. Timoshenkon ja Woinowsky-Kriegerin mukaan $\nu = 0.30$ ja taipuman ollessa suuri suhteessa levyn paksuuteen tällaiselle kappaleelle pätee likimäärin (Timoshenko ja Woinowsky-Krieger, 1987, s.412)

$$\frac{w_0}{h} + 0{,}262\left(\frac{w_0}{h}\right)^3 = 0{,}696\frac{qa^4}{Eh^4} \tag{49}$$

Kaavasta (49) voidaan ratkaista w_0, joka on esitetty taulukossa 3 kuormilla 0,6 MPa ja 2,0 MPa. Näin saadut tulokset vastaavat kuitenkin kokoonpuristuvaa materiaalia. Yleisessä tapauksessa pitää Timoshenkon ja Woinowsky-Kriegerin mukaan ratkaista yhtälöt (s.409)

$$S_r = \frac{h^2}{12(1-\nu^2)a^2\rho}(B_0\rho + B_3\rho^3 + B_5\rho^5 + \cdots) \tag{50}$$

$$\frac{dw}{dr} = -\frac{h}{2a\sqrt{3}}(C_1\rho + C_3\rho^3 + C_5\rho^5 + \cdots) \tag{51}$$

missä S_r on säteen suuntainen voima, $\rho = r/a$ ja kertoimet B_k ja C_k saadaan yhtälöistä (s.409)

$$B_k = -\frac{1-\nu^2}{2(k^2-1)} \sum_{m=1,3,5,\ldots}^{k-2} C_m C_{k-m-1} \qquad k = 3,5,\ldots \tag{52}$$

$$C_k = -\frac{1}{k^2-1} \sum_{m=1,3,5,\ldots}^{k-2} C_m B_{k-m-1} \qquad k = 5,7,\ldots \tag{53}$$

$$8C_3 - B_1C_1 + 12\sqrt{3}(1-\nu^2)\frac{qa^4}{Eh^4} = 0 \tag{54}$$

Symmetrisen laatan keskipisteessä taipuma on $dw/dr = 0$. Koska reuna on vapaa liikkumaan säteen suunnassa, $S_a = 0$ ja voimassa on ehdot (s.409)

$$\sum_{k=1,3,5,\ldots} B_k = 0 \qquad\qquad \sum_{k=1,3,5,\ldots} C_k(k+\nu) = 0 \tag{55}$$

Kaavojen (52), (53), (54) ja (55) perusteella kertoimet voidaan ratkaista numeerisesti eri kuormitustapauksissa. Taulukossa 2 on koottu ratkaistuja sarjakehitelmän kertoimien likiarvoja kokoonpuristuvan ($\nu = 0.30$) ja lähes kokoonpuristumattoman ($\nu = 0.48$) materiaalin tapauksissa kuormilla 0,6 MPa ja 2,0 MPa. Laskennassa on käytetty arvoja h=1mm, a=10mm ja E=10GPa.

Taulukko 3. *Laatan taipuman sarjakehitelmän vakiokertoimia*

$\boldsymbol{\nu}$	$\boldsymbol{q}$ [MPa]	$\boldsymbol{B_1}$	$\boldsymbol{B_3}$	$\boldsymbol{B_5}$	$\boldsymbol{B_7}$	$\boldsymbol{C_1}$	$\boldsymbol{C_3}$	$\boldsymbol{C_5}$	$\boldsymbol{C_7}$
0,30	0,6	0,505	-0,654	0,155	-0,006	-3,391	1,204	0,118	-0,026
0,30	2,0	3,705	-4,207	0,243	0,259	-8,601	0,746	1,623	0,016
0,48	0,6	0,316	-0,421	0,113	-0,008	-2,719	1,092	0,062	-0,016
0,48	2,0	2,575	-3,166	0,452	0,139	-7,461	1,597	1,155	-0,113

Taulukon 3 arvojen perusteella kaavasta (51) voidaan laskea w_0. Esitettyjen kuormitustapauksien tulokset on koottu taulukkoon 4.

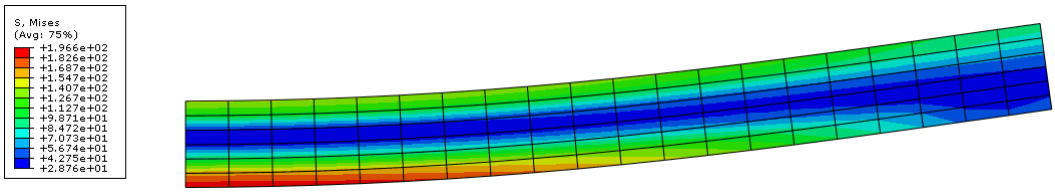
Taulukko 4. *Laatan keskipisteen taipuma*

ν	q [MPa]	w_o [mm] (49)	w_o [mm] (51)
0,30	0,6	-0,401	-0,398
0,30	2,0	-1,071	-1,109
0,48	0,6		-0,311
0,48	2,0		-0,910

Taulukossa 4 on esitetty keskipisteen taipuma w_o. Kokoonpuristuvalle materiaalille voidaan laskea taipuma kaavalla (49) ja (51). Kokoonpuristumattomalle materiaalille saadaan taipumat vain kaavasta (51). 0,6 MPa kuormalla kaava (51) antaa 0,7% pienemmän arvon kuin kaava (49). 2,0 MPa kuormalla kaava (51) antaa 3,6% suuremman arvon kuin kaava (49). Tässä tarkastelussa ollaan kiinnostuneita suurista taipumista kokoonpuristumattomalla materiaalilla. Siksi seuraavassa osiossa vertailuarvoina käytetään kaavalla (51) saatuja taipumia. Näitä taipuman arvoja verrataan Abaqus-ohjelmalla laskettuihin arvoihin.

4.2 Numeerisen testikappaleen käsittely

Numeeriset ratkaisut saatiin mallintamalla analyyttistä tapausta vastaava testikappale Abaqus-ohjelmassa erilaisilla verkon tiheyksillä. Kaikki verkon tiheydet laskettiin taulukon 3 esittämissä kuormitustapauksissa.

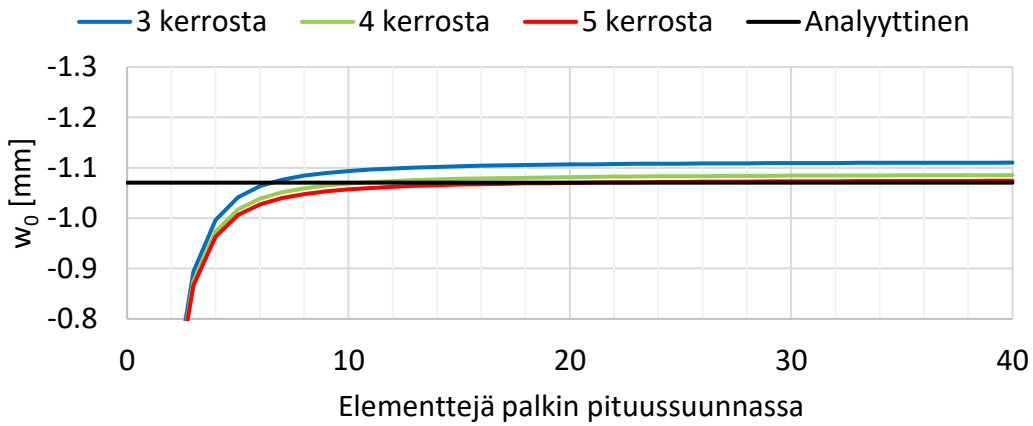


Kuva 7. *Laatan jännitykset 20x6 CAX4RE-elementeillä*

Kuva 7 esittää levyn puolikkaan pyörähdyssymmetristä mallia, jossa elementtien lukumäärä on 20x6 (20 elementtiä vierekkäin, 6 elementtiä päällekkäin). Elementtityyppinä on kuvan tapauksessa CAX4RE eli ali-integroitu pyörähdyssymmetrinen neliöelementti parannetulla tiimalasikontrollilla. Vertailussa käytetään myös pyörähdyssymmetrisiä neliöelementtejä CAX4 ja CAX4R, jotka ovat täysin integroitu neliöelementti ja ali-integroitu, kontrolloimaton neliöelementti. Laskennassa käytetään suurten siirtymien epälineaarisen geometrian teoriaa. Suurimmat Von Mises -jännitykset syntyvät levyn keskikohdan alapinnalle (kuvassa 7 vasempaan alanurkkaan). Alkuperäisenä oletuksena oli, että kaikilla elementtityypeillä elementtien määrän kasvattaminen pituus- ja poikittaissuunnissa parantaisi tulosten tarkkuutta. Oletettavasti paras tarkkuus saavutettaisiin elementeillä, joiden sivujen pituus olisi lähellä toisiaan. Nämä oletukset osoittautuivat vääriksi. Riittävän tiheyden saavuttamisen jälkeen merkittävin tekijä oli elementtityyppi.

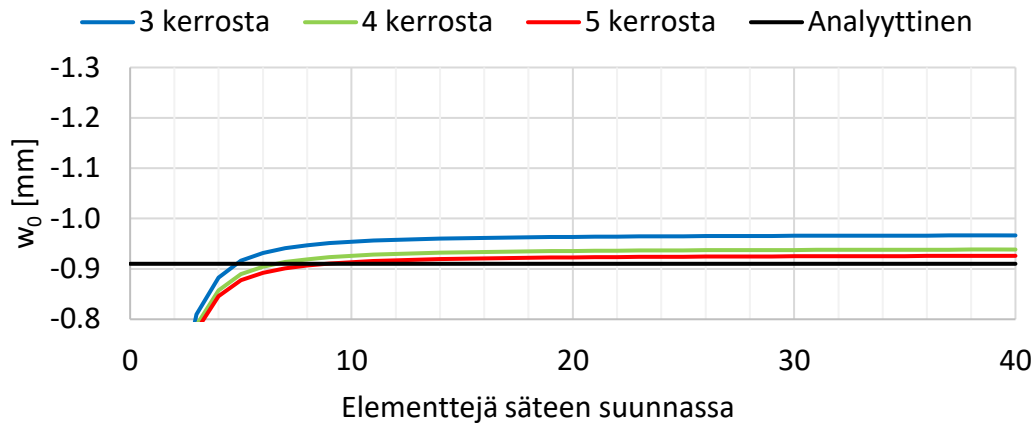
Kuvasta 8 nähdään miten CAX4-elementeillä elementtien lukumäärä vaikuttaa taipuman suuruuteen. Elementtien lukumäärän kasvaessa säteen suunnassa elementtien jäykkyys putoaa ja tulos lähestyy ja lopulta ylittää analyyttisen taipuman. Noin 20 elementin jälkeen elementtien määrän lisääminen säteen suunnassa ei enää merkittävästi paranna tarkkuutta. Elementtikerrosten määrän kasvaessa tulos on lähempänä analyyttistä tulosta. Paras tulos saadaan viidellä elementtikerroksella.

Elementtien muoto ei vaikuta tuloksiin tiheydestä eroavalla tavalla. Jos elementin neliömäisyydestä olisi merkittävää etua, neliömäisemmät elementit erottautuisivat edukseen verrattuna niitä tiheämpiin elementtiladontoihin. Esimerkiksi kolmella elementtikerroksella neliömäisimmät elementit saadaan säteen suunnassa 30 elementin kohdalla. Tässä kohdassa kuvaajaa ei näy huomattavaa eroa muihin kohtiin nähden.



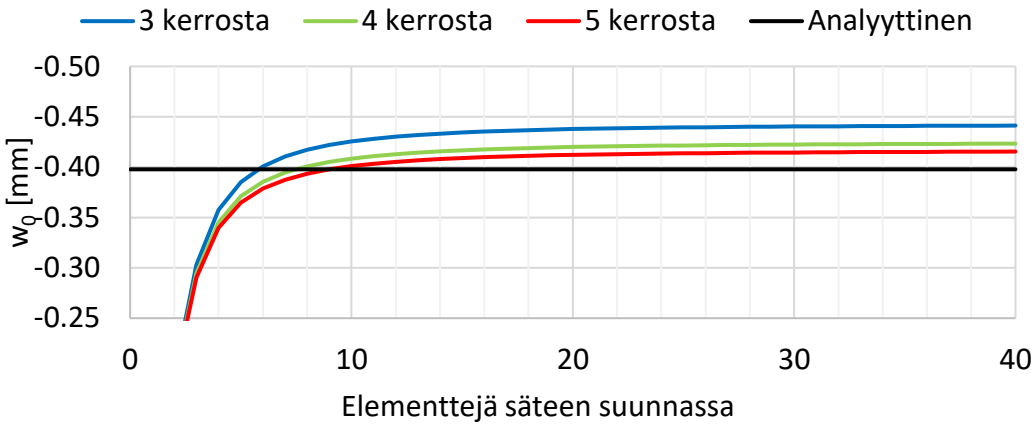
Kuva 8. *CAX4-elementtien lukumäärän vaikutus keskipisteen taipumaan, v=0,30, q=2,0 MPa*

Kuvassa 9 esitetään kuvan 8 tapaus käytettäessä lähes kokoonpuristuvaa materiaalia. Analyyttinen taipuman arvo on laskenut kuvaan 8 verrattuna alemmalle tasolle. CAX4-elementtien virhe ei poikkea kokoonpuristuvan tapauksen tuloksesta. Kuvaajien muoto on lähes identtinen kuvaan 8 verrattuna. Kuvaajien keskinäinen järjestys pysyy samana.



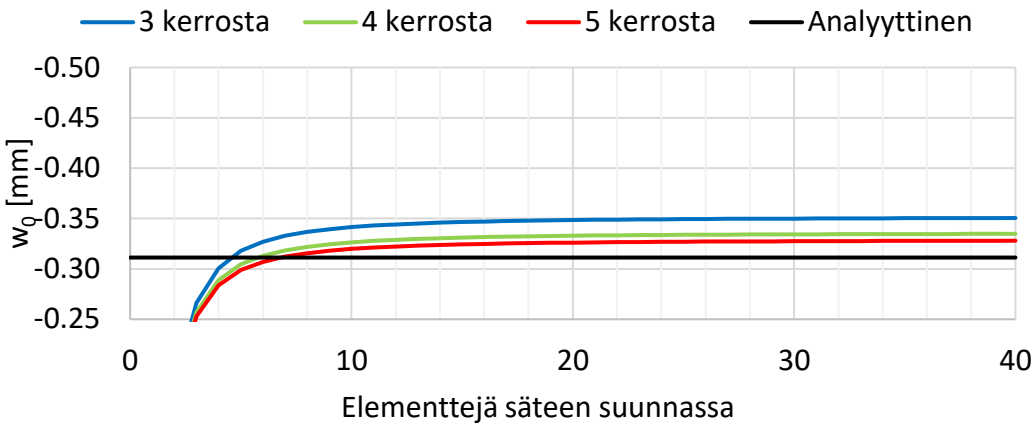
Kuva 9. *CAX4-elementtien lukumäärän vaikutus keskipisteen taipumaan, v=0,48, q=2,0 MPa*

Kuvassa 10 näytetään kuvan 8 tapaus pienellä kuormalla. Tasaantunut taipuma on pienempi kuin kuvassa 8. Kuorman muutos ei vaikuta CAX4-elementeillä kokoonpuristuvan materiaalin tarkkuuteen.



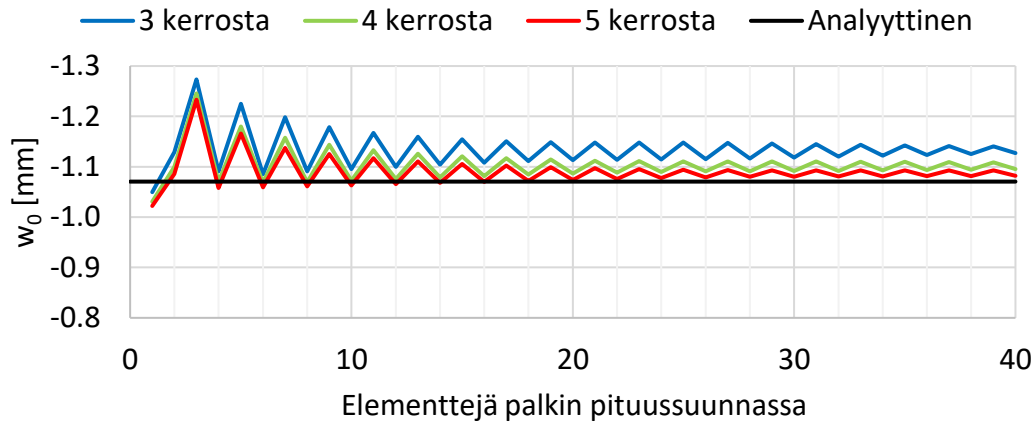
Kuva 10. *CAX4-elementtien lukumäärän vaikutus keskipisteen taipumaan, v=0,30, q=0,6 MPa*

Kuvassa 11 näytetään kuvan 10 tapaus lähes kokoonpuristumattomalla materiaalilla. Tasaantunut kuorma jää suhteessa kauemmas analyyttisestä vertailuarvosta.



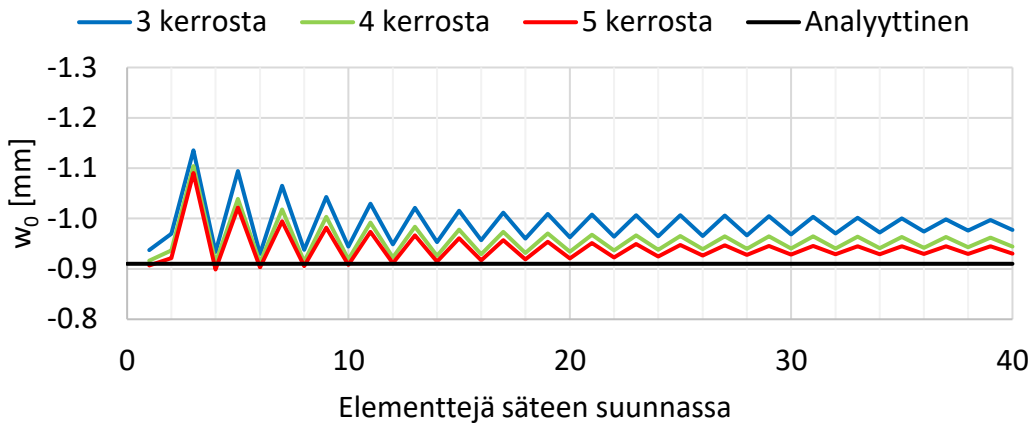
Kuva 11. *CAX4-elementtien lukumäärän vaikutus keskipisteen taipumaan, v=0,48, q=0,6 MPa*

Tulosten perusteella CAX4-elementeillä tehokkain keino parantaa laskennan tarkkuutta on kasvattaa elementtien lukumäärää. Säteen suunnassa elementtien lukumäärän optimaalinen tiheys on noin 20 elementtiä. Tätä suuremmilla elementtimäärillä laskenta-aika pitenee ilman merkittävää hyötyä tarkkuuteen. Paksuussuunnassa parhaat tulokset saatiin 5 elementtikerroksella. Näillä arvoilla elementin kooksi tulee 0,5 mm x 0,2 mm.



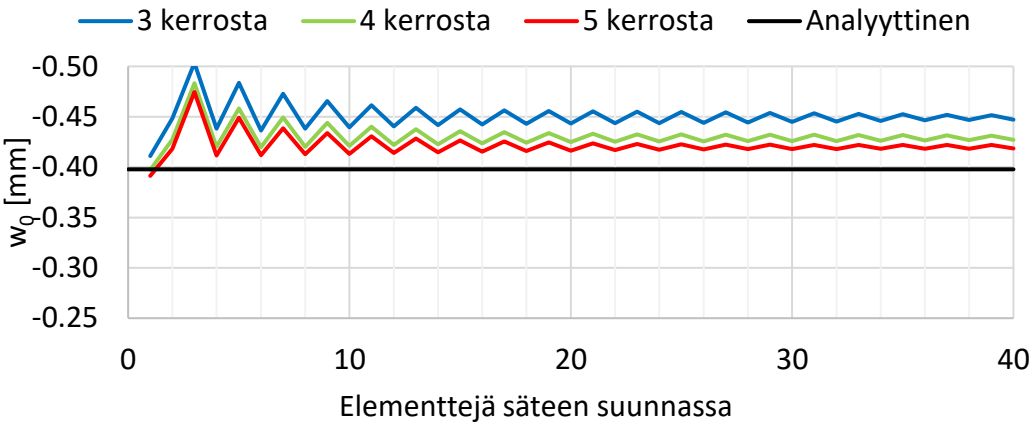
Kuva 12. *CAX4R-elementtien lukumäärän vaikutus keskipisteen taipumaan, v=0,30, q=2,0 MPa*

Kuvassa 12 on kuvan 8 tapaus käytettäessä ali-integroituja CAX4R-elementtejä. CAX4-elementtien tapaan CAX4R-elementeillä taipuman tarkkuus paranee elementtien määrän kasvaessa säteen suunnassa. Aiemmasta poiketen parittomilla elementtimäärillä taipuman arvo kasvaa suuremmaksi kuin parillisilla elementtimäärillä. Tämä johtuu todennäköisesti elementtien tiimalasimuodonmuutoksista. Elementtikerrosten määrän kasvaessa tulos lähestyy analyyttistä tulosta.



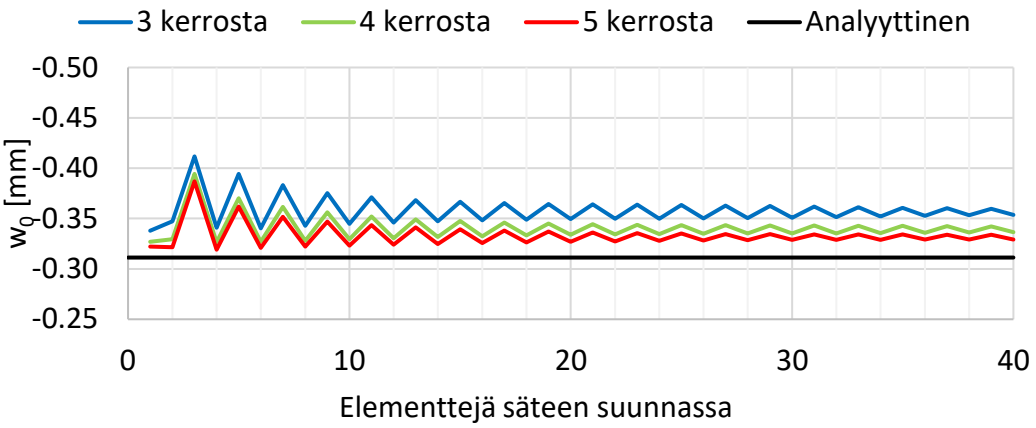
Kuva 13. *CAX4R-elementtien lukumäärän vaikutus keskipisteen taipumaan, v=0,48, q=2,0 MPa*

Kuvassa 13 on kuvan 12 tapaus lähes kokoonpuristumattomalla materiaalilla. Kokoonpuristumattoman materiaalin käyttö ei vaikuta merkittävästi CAX4R-elementtien tarkkuuteen.



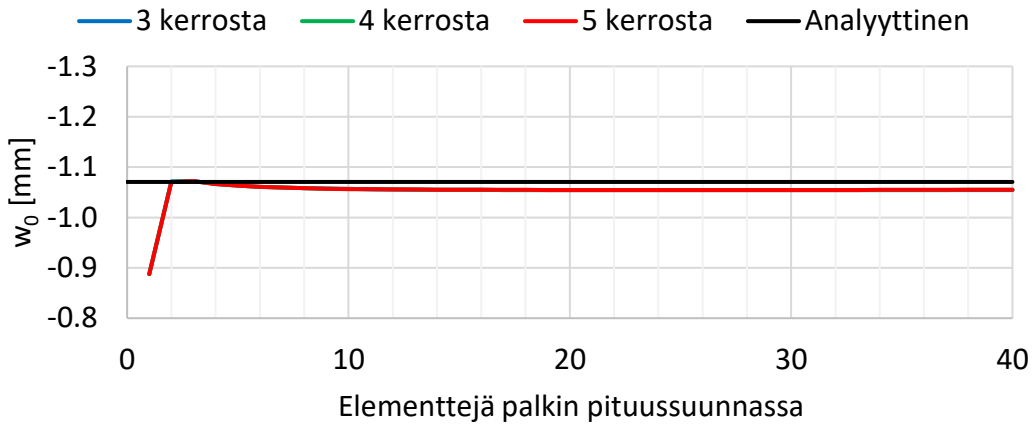
Kuva 14. *CAX4R-elementtien lukumäärän vaikutus keskipisteen taipumaan, v=0,30, q=0,6 MPa*

Kuvassa 14 näytetään kuvan 12 tapaus pienemmällä kuormalla. CAX4-elementtien tapaan tulos jää suhteessa kauemmas vertailuarvosta kuin suuremmilla kuormilla.



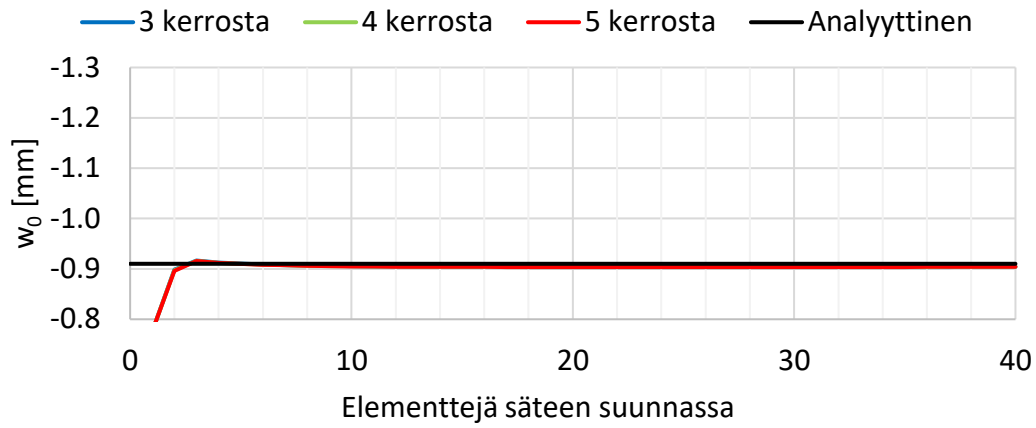
Kuva 15. *CAX4R-elementtien lukumäärän vaikutus keskipisteen taipumaan, v=0,48, q=0,6 MPa*

Kuvassa 15 esitetään kuvan 14 tapaus lähes kokoonpuristumattomalla materiaalilla. Materiaalimalli vaikuttaa taipuman suuruuteen, mutta kokonaisuutena elementtien käytös muistuttaa kuvan 14 tapausta.



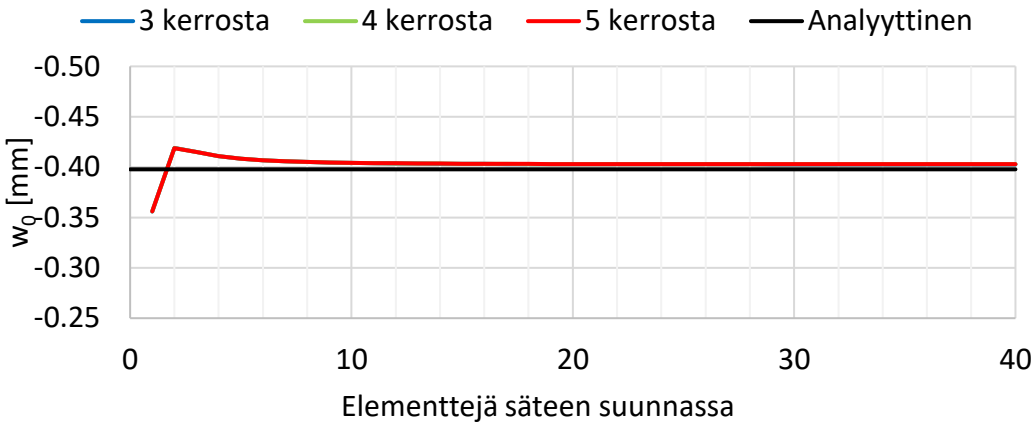
Kuva 16. *CAX4RE-elementtien lukumäärän vaikutus keskipisteen taipumaan, v=0,30, q=2,0 MPa*

Kuvassa 16 näytetään kuvan 12 tapaus käytettäessä CAX4RE-elementtejä. Näille elementeillä parannettu tiimalasikontrolli poistaa tiimalasimuodonmuutoksien aiheuttaman sahalaitakuvion. Samalla kontrolli tasaa kerroksien elementtivahvuuksista johtuvan vaihtelun. Taipuman suuruus ei näillä elementeillä riipu elementtikerrosten lukumäärästä. Taipuman absoluuttinen suuruus tasaantuu lähelle analyyttistä vertailuarvoa, mutta laskettu tulos jää kuitenkin noin 1,5% pienemmäksi kuin analyyttinen tulos.



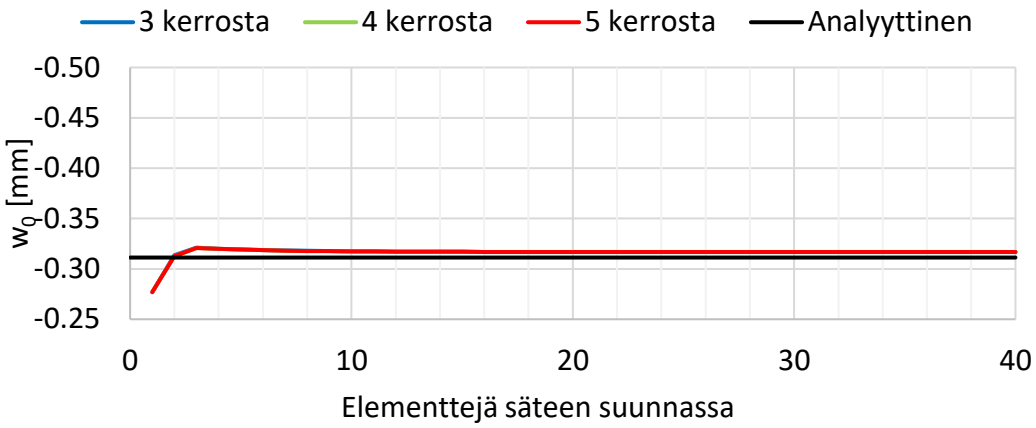
Kuva 17. *CAX4RE-elementtien lukumäärän vaikutus keskipisteen taipumaan, v=0,48, q=2,0 MPa*

Kuvassa 17 on kuvan 16 tapaus lähes kokoonpuristumattomalla materiaalilla. Kokoonpuristumattomuus ei tässäkään tapauksessa muuta merkittävästi elementtien käytöstä. Taipuman laskettu arvo jää noin 0,7% pienemmäksi kuin analyyttinen vertailuarvo.



Kuva 18. *CAX4RE-elementtien lukumäärän vaikutus keskipisteen taipumaan, v=0,30, q=0,6 MPa*

Kuvassa 18 esitetään kuvan 16 tapaus pienemmällä kuormalla. Pienempi kuorma muuttaa taipumien suhteellista eroa. Tässä tapauksessa laskennallinen absoluuttinen taipuma on noin 1,3% suurempi kuin analyyttinen vertailuarvo.



Kuva 19. *CAX4RE-elementtien lukumäärän vaikutus keskipisteen taipumaan, v=0,48, q=0,6 MPa*

Lopulta kuvassa 19 näytetään kuvan 18 tapaus kokoonpuristumattomalla materiaalilla. Tässä tapauksessa laskennallinen taipuma on noin 1,8% suurempi kuin analyyttinen vertailuarvo.

4.3 Stabiloinnin merkitys

Yllä saatujen tulosten perusteella voidaan väittää, että käytettäessä säännöllistä nelikulmioelementtiverkkoa, CAX4R-elementtejä ja parannettua tiimalasikontrollia, mallin yksittäisen kohdan elementtien tiheydellä ei ole merkitystä globaalin mallin ratkaisun tarkkuuteen. Esimerkiksi rengasmallin sivupinnan elementtien tiheys voidaan vapaasti valita ilman, että se vaikuttaa koko renkaan painumaan. Tämä on vastoin kokemusperäistä ohjetta, jonka mukaan sivupinnan elementtirivien paksuuden pitää olla riittävä. Ohje on todennäköisesti ajalta ennen enhanced-stabilointia, eikä se siten enää ole pätevä.

Epäsäännöllisen verkon tapauksessa elementtien käytös voi tietenkin muuttua. Vaihtuvan elementtitiheyden käsittely vaatisi kolmioiden lisäämistä tiheyden muutoskohtiin. Tämä puolestaan vaatisi kolmioelementtien teorian ja kolmioiden lukumäärän ja sijainnin tarkempaa käsittelyä. Siihen ei aiheen laajuuden vuoksi tässä työssä syvennytä.

Flanaganin ja Bindemanin mukaan tässä nelikulmioelementeille saadut tulokset pätevät vain, jos elementit ovat suorakulmaisia. Jos elementin muoto poikkeaa neliöstä, elementin tarkkuutta ja laskennan suppenemista ei voi taata. Tämä muotovirhe on todennäköinen syy laskenta-ajan kasvamiselle aiemmissa analyyseissä. Kolmioiden vaikutuksesta ei saatu merkittävää teoreettista tai kokeellista näyttöä, joten niitä ei erikseen käsitellä tässä työssä.

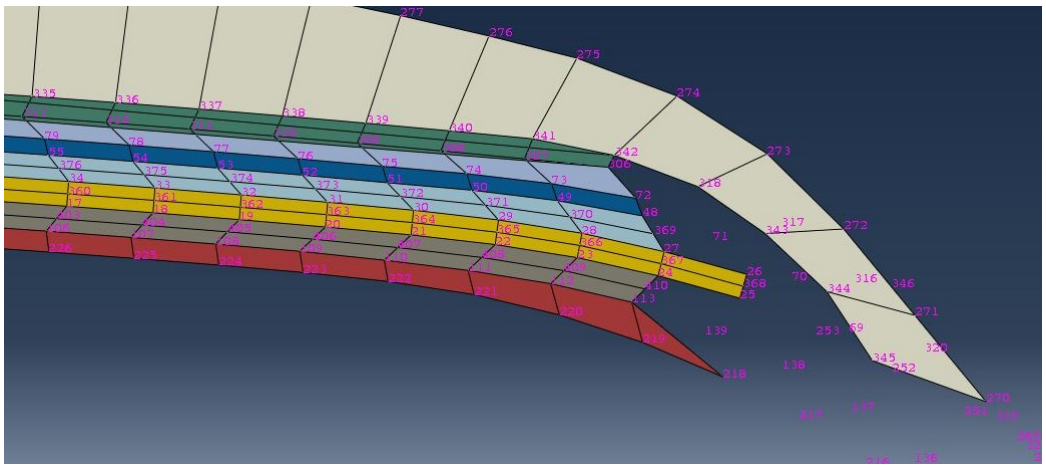
5. OHJELMISTON TOTEUTUS

Ohjelmisto koostuu verkottajasta, Abaqus-rajapinnasta ja CAD-rajapinnasta. Verkottaja ja Abaqus-rajapinta on toteutettu Python-ohjelmointikielellä, koska ne käyttävät Abaqus-ohjelman omaa Python-kääntäjää. Kääntäjän avulla verkottaja pääsee käsiksi mm. Abaqus-ohjelman solmuihin ja elementteihin. Abaqus-rajapinta yhdistää ohjelmiston muut osat toisiinsa. Se kutsuu ensin CAD-rajapintaa, joka on toteutettu C#-ohjelmointikielellä. CAD-rajapinta avaa yhteyden Creo 4.0 -ohjelmistoon geometrian lukemista varten. Tämän jälkeen Abaqus-rajapinta lähettää tiedot verkottajalle. Verkottaja luo tietojen perusteella rengasmallin. Abaqus- ja CAD-rajapintoja ei käsitellä tämän syvällisemmin, koska ne kuuluvat yrityssalaisuuden piiriin. Tämä osio kuvaa pääasiassa verkottajan toimintaa.

Automaattinen laskentamalli luodaan eri tavalla kuin käsintehty malli. Käsintehdyssä mallissa käyttäjä erottelee rengasmallin piirteitä toisistaan ulkonäön perusteella. Tämän erottelun automatisointi olisi erittäin vaikeaa. Sen sijaan malli voidaan jakaa osiin taustalla olevan rengasmallin piirteiden mukaan. Rengasmallin geometria haetaan viiva kerrallaan. Tällöin tiedetään jo haettaessa, mitä mallin osaa viivan janat ja käyrät esittävät. Samalla vältytään ylimääräisiltä viivoilta, jotka ovat mallin luonnin kannalta turhia ja aiheuttavat mallissa verkotusvirheitä.

5.1 Geometrian esikäsittely

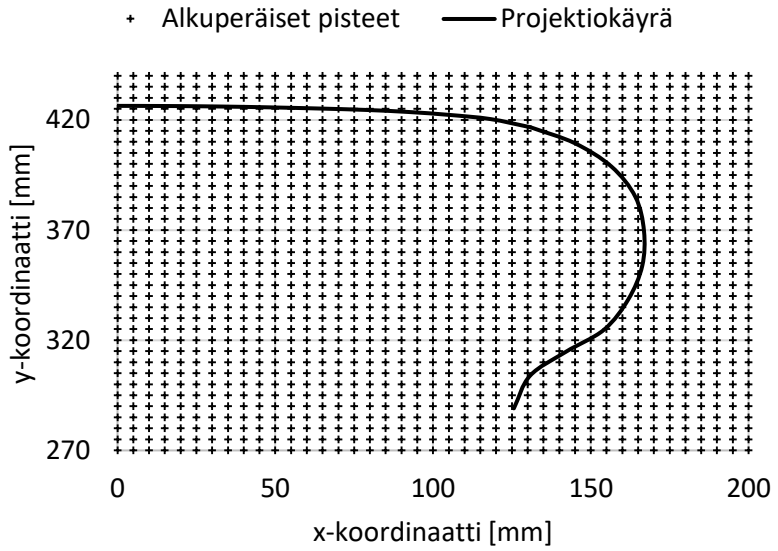
Tuodussa geometriassa on puutteita. Piirteiden viivat ja niistä luodut pistekoordinaatit eivät sovi sellaisenaan yhteen. Pistekoordinaatit täytyy normalisoida, jotta elementit olisivat mahdollisimman suorakulmaisia. Tämä normalisointi vastaa periaatteessa advancing layers -algoritmin ensimmäistä vaihetta. Ilman normalisointia elementtien suorakulmaisuus riippuu solmujaosta, mitä on vaikea hallita rengasmallista ja alueesta toiseen. Kuvassa 20 näkyy miltä verkotusalgoritmin tulos näyttää ilman normalisointia, kun kaikilla alueilla on sama solmujako. Keltaisella näkyvissä teräsvöissä suorakulmaisuus on onnistunut hyvin, mutta käytännössä kaikissa muissa alueissa elementit ovat vääristyneet.



Kuva 20. *Ilman normalisointia luotu epäsäännöllinen verkko.*

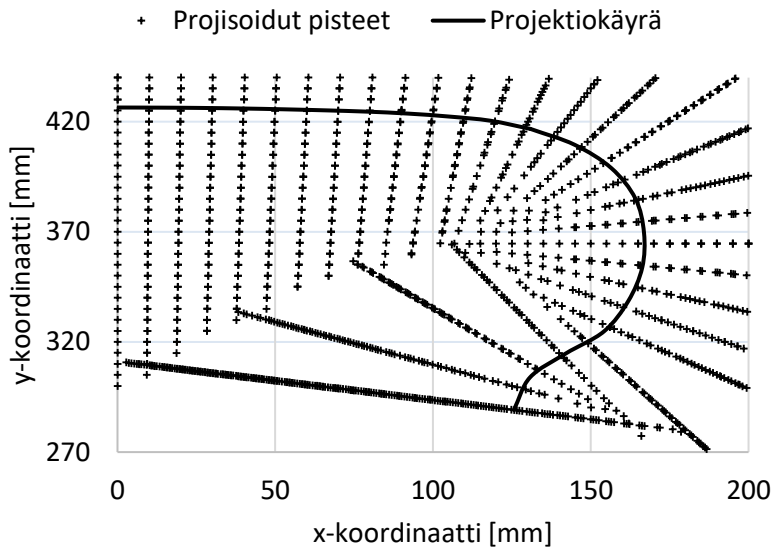
5.1.1 Koordinaattien normalisointi

Normalisoinnissa rengasmallista valitaan projektiokäyrä, esimerkiksi runkokoordin linja, jolle asetetaan tasaisesti jaettuna referenssipisteitä. Referenssipisteiden jako vaikuttaa mallin geometriseen tarkkuuteen, mikä nähdään tuloksia tarkasteltaessa. Jokaisen referenssipisteen kohdalla lasketaan projektiokäyrään nähden kohtisuora normaaliviiva. Seuraavaksi jokaisen malliin tuotavan piirteen pistekoordinaatit projisoidaan pistettä lähimmälle normaaliviivalle. Kuvassa 21 esitetään erään renkaan projektiokäyrä ja esimerkkinä käytettävä 5 mm tasajaolla tehty pisteruudukko.



Kuva 21. *Normalisoinnin projektiokäyrä ja tasan jaettu pisteruudukko*

Kuvassa 22 näytetään sama pisteruudukko projisoinnin jälkeen. Projisoidut pisteet siirtyvät projektiokäyrään nähden kohtisuorille normaalisuorille. Pisteiden etäisyys toisistaan riippuu projektiokäyrän kaarevuudesta ja pisteen etäisyydestä projektiokäyrästä.



Kuva 22. *Normalisoinnin projektiokäyrä ja projisoidut pisteet*

Joissakin tapauksissa projisoinnissa syntyy aukkoja, jolloin kahden pisteen välissä olevalle normaalisuoralle ei projisoida yhtään pistettä. Nämä aukkokohdat täytetään ylimääräisellä pisteellä, jonka koordinaatit saadaan käänteisesti projisoimalla normaalin referenssipiste piirteen geometriaviivalle. Myöhemmin normaalisuoria voidaan käyttää verkotuksen apuna, joten myös normaalin indeksi poimitaan talteen. Käyttämällä normaalin indeksiä pisteiden etsimiseen säästetään verkottamisessa aikaa, koska lähimmän pisteen etsintää ei tarvitse tehdä enää uudestaan. Normalisoinnin jälkeen geometria on valmis verkottamiseen.

5.2 Verkotusalgoritmi

Verkottamisessa osioidut tasoalueet jaetaan laskennan käyttämiin elementteihin. Verkotettavan alueen solmupisteistä osa sijaitsee alueen reunaviivoilla ja osa alueen sisällä. Jotta yksittäisen alueen verkko olisi yhteensopiva muun mallin kanssa, vierekkäisten alueiden solmujakojen tulisi olla yhteneviä. Solmujen lukumäärän ja paikkojen pitää täsmätä alueiden reunoilla. Alueiden yhdistäminen onnistuu parhaiten, jos verkottamisessa käytetään samoja sääntöjä ja toleransseja alueiden kesken.

Alueen sisäisten solmujen verkottamisessa tavoitteena on muodostaa solmuketjuja vastakkaisten reunasolmujen välille. Jos vastakkaisissa solmuketjuissa on yhtä monta solmua, ketjujen välinen alue voidaan verkottaa täysin neliöelementeillä. Jos taas solmujen määrässä on eroa, alueen sisään pitää luoda yksi tai useampi kolmioelementti.

5.2.1 Vahvikekuitujen solmut

Vahvikekuidut on helpointa verkottaa kumielementtien perusteella. Vahvike-elementit mallinnetaan kaksiulotteisilla, kaksisolmuisilla SFMAX-viivaelementeillä. Perinteisesti

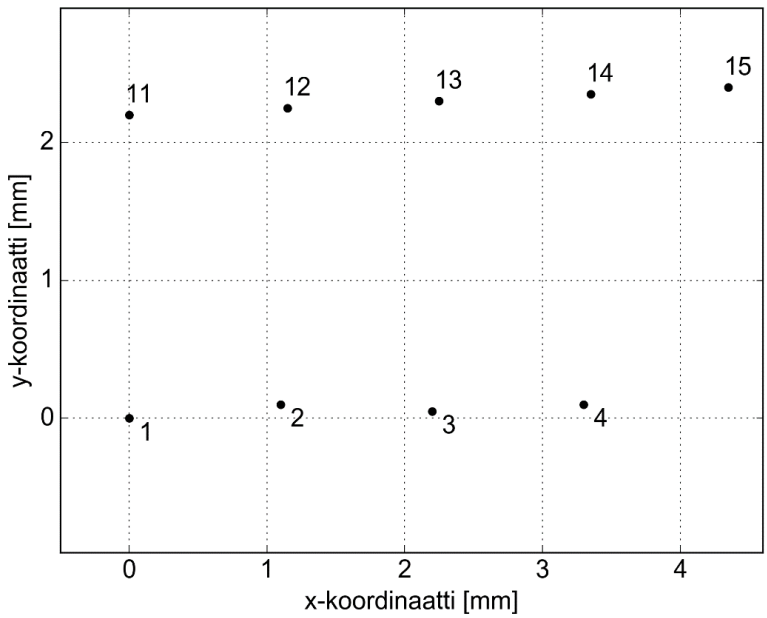
vahvikeverkko ja kumiverkko tehdään erillään toisistaan, mutta elementtien koko pyritään pitämään vakiona. Tämä voi olla vaikeaa. Täydellisessä verkossa jokaista vahvike-elementtiä vastaamassa on yksi kumielementti. Jos vahvikeverkko tehdään kumiverkon pohjalta, vahvikeverkon solmut voidaan panna haluttuun paikkaan ja verkoista saadaan varmasti yhteensopivia. Vahvike-elementin solmupisteillä on kaksi järkevää paikkaa:

1. kumielementtien geometrisissä painopisteissä
2. kumielementtien reunoilla

Solmupisteiden paikka määrää, miten Abaqus integroi vahvikekuidun vaikutuksen lujitettavaan elementtiin. Elementin keskipisteeksi valittiin monikulmion geometrinen painopiste, koska se on helppo laskea solmukoordinaateista. Tämän seurauksena myöhemmin huomattiin, että jos vahvikesolmut sijoitetaan kumielementtien painopisteisiin, vahvike-elementti voi deformoituna kääntyä sitä ympäröivän kumielementin ulkopuolelle. Tämä saattaa aiheuttaa laskennan kaatumisen. Jos vahvikesolmut sijoitetaan kumielementtien reunoille, vahvikekuitujen elementit pysyvät aina täysin niitä rajaavien kumielementtien sisällä. Tätä ajatusta viemällä pidemmälle huomattiin, että jos vahvikesolmut sijoitetaan kumielementtien nurkkasolmuihin, tämä ehto täyttyy ja verkon solmupisteiden määrä pysyy pienenä. Tämä lyhentää laskenta-aikaa.

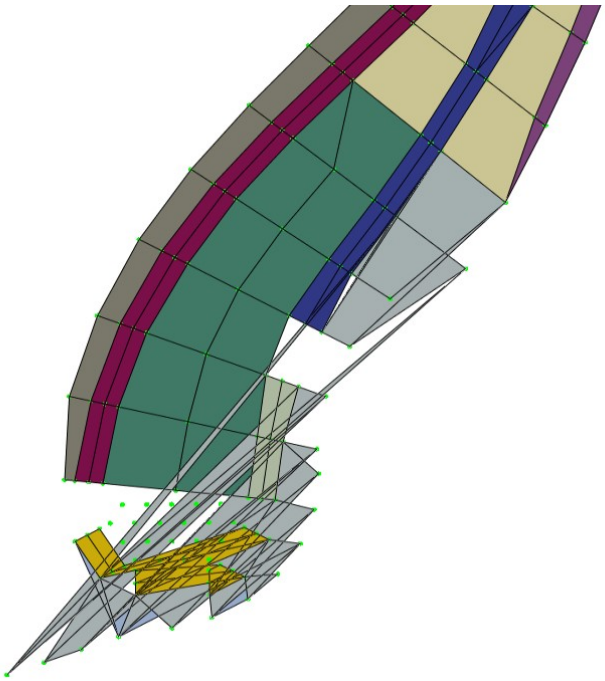
5.2.2 Solmujen luonti ja järjestys

Verkottajan toiminta on esitetty vuokaaviona liitteessä A. Verkotusalgoritmi aloitetaan hakemalla esikäsitellyt, normalisoidut solmupisteiden koordinaatit renkaan geometriamallista. Abaqus ja geometriamalli käsittelevät pisteitä kolmiulotteisina, mutta aksisymmetrisessä mallissa pisteen kolmas koordinaatti voidaan ohittaa.



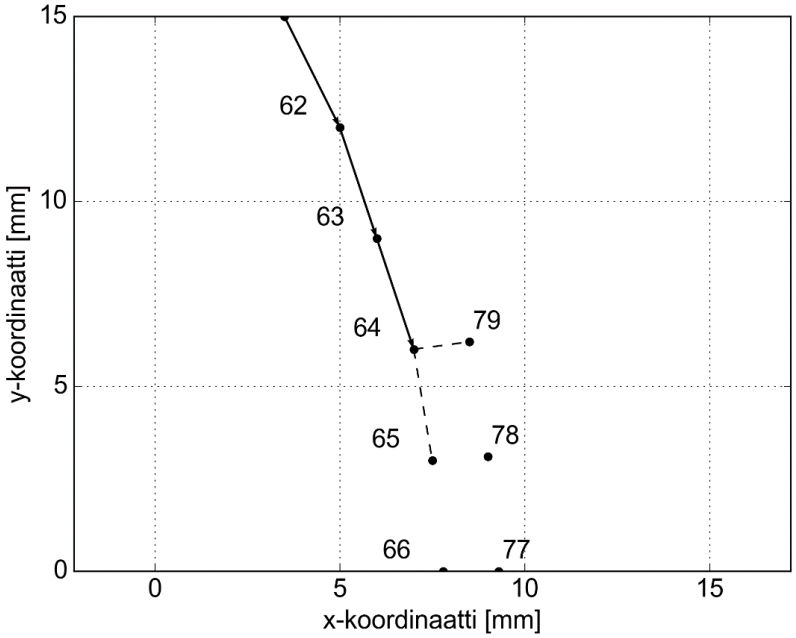
Kuva 23. *Koordinaattien perusteella luotuja solmuja*

Kuvassa 23 esitetään kaksi solmujoukkoa, jotka on luotu geometrian pistekoordinaattien pohjalta laskentamalliin. Kun solmujen luonti aloitetaan alemmasta joukosta, sen ensimmäisen solmun numeroksi tulee yksi, toiseksi kaksi jne. Rengasmallissa kuvioiden suunta ja siitä johtuen koordinaattien järjestys voi olla mielivaltainen. Koordinaattien järjestys määrää solmujen järjestyksen ja solmujen järjestys määrää elementtien järjestyksen. Ennen solmujen luontia koordinaatit pitää asettaa järjestykseen, mikä tehdään funktiossa *SortCoordinates.* Tässä esimerkissä solmunumerointi kasvaa siirryttäessä kauemmaksi y-akselista (x=0). Rengasmallissa yleisenä järjestyssuuntana käytetään keskiviivaa kallistettuna 15 astetta renkaan keskiaselista jalkaa kohti. Tämä referenssisuunta painottaa pisteen x-koordinaattia kertoimella 0,259 ja y-koordinaattia kertoimella 0,966. Tämä referenssisuunta toimii myös renkaan sivupinnan alueella, vaikka solmut kääntyvätkin sisäänpäin.



Kuva 24. *Virhe solmujen järjestyksessä luo vääristyneitä elementtejä*

Samaa referenssisuuntaa ei voida käyttää koko renkaalle. Kuvassa 24 näkyy virheellisen referenssisuunnan vaikutus verkotukseen. Verkottaja luo yhteyksiä solmujen välille niiden järjestyksen perusteella. Jos solmujärjestys on valittu väärin, elementeistä tulee vääristyneitä. Jalka-alueella solmujoukot kiertävät kaapelia, joten on luonnollista, että solmujärjestyksen tulee käyttää kaapelia referenssinä. Koordinaattien järjestämiseen käytetään myöhemmin esitettävää *NodeSlope*-funktiota, jonka avulla koordinaatit kierretään vastapäivään. Kierroksen keskipisteenä käytetään kaapelin keskipistettä.



Kuva 25. *Lähimmän solmupisteen hakeminen erikoistapauksessa*

Kuvassa 25 esitetään erikoistapaus, jossa solmujoukko kääntyy itsensä ympäri. *SortCoordinates* palauttaisi suunnaltaan vaihtelevan solmujärjestyksen (62, 63, 64, 79, 65, 78). Järjestyksen haluttaisiin kuitenkin jatkuvan sisempää pistejoukkoa pitkin (62, 63, 64, 65, 66). Tämän ongelman välttämiseksi solmujoukot pitää jakaa sellaisiin osiin, jotka eivät käänny itseään kohti. Ulommat solmut (79, 78, 77) pitää siirtää jo tuontivaiheessa toiseen solmujoukkoon.

Solmut luodaan funktiossa *CreateNodes*. Jos uuden solmun koordinaatit ovat liian lähellä olemassaolevaa solmua, uutta solmua ei luoda vaan palautetaan olemassaoleva solmu. Näin estetään pienten, virheellisten elementtien syntyminen. Lähin solmu etsitään ohjelmassa 1 esitetyllä funktiolla *GetClosestNode*.

```
def GetClosestNode(node, nodes):
  """Etsi solmujoukosta nodes solmua node lähimpänä oleva"""
  distances = []
  for currentNode in nodes:
    # Kootaan solmujen välinen vektori
    delta = tuple(t-n for t,n in zip(node.coordinates, \
                  currentNode.coordinates))
    # Laske solmujen välinen etäisyyden neliö
    distanceToNode = delta[0]**2+delta[1]**2
    # Kerätään solmut, hakuavaimena niiden etäisyydet
    distances.append((distanceToNode,currentNode))
  # Funktio palauttaa solmun jolla on pienin etäisyys
  return min(distances)[1]
```

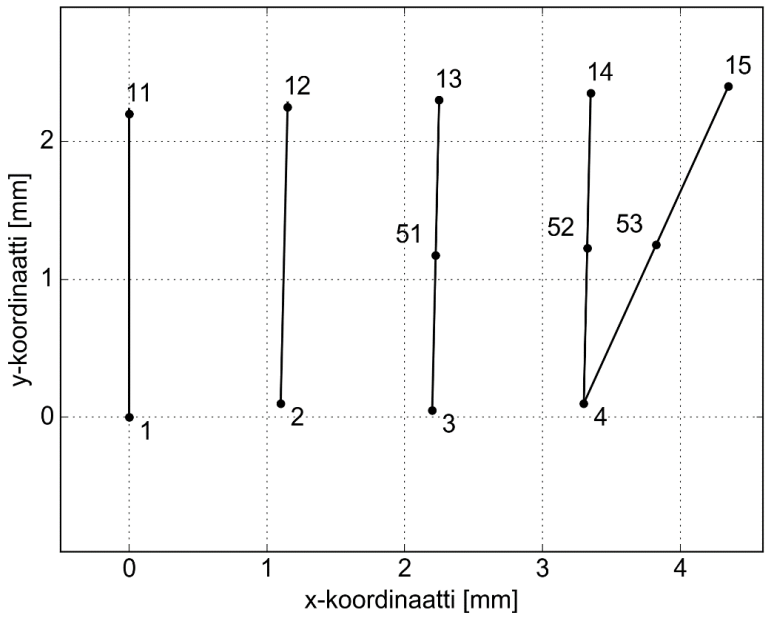
Ohjelma 1. Lähimmän solmun etsiminen solmujoukosta

Funktiota *GetClosestNode* käytetään funktioissa *CreateNodes*, *CreateRails* ja *CreateRungs*. Sille syötetään solmu *node*, jolle halutaan löytää pari ja solmujoukko *nodes*, mistä

paria etsitään. Ohjelma käy vuorollaan verrattavat solmut läpi ja muodostaa kahden solmun koordinaattien perusteella niiden välille vektorin (rivi 6). Vektorista lasketaan sen pituuden neliö (rivi 8) ja se lisätään dictionary-objektiin *distances* (rivi 9). Ylimääräistä neliöjuuren laskentaa ei tarvita, koska tuloksena ei haluta välimatkaa vaan lähin solmu. Lopulta funktio poimii *distances:*stä sen pienimmän arvon ja palauttaa siitä vain solmuobjektin (rivi 13). *GetClosestNode* toimii $O(n^2)$ -ajassa, mutta solmujoukkojen ollessa pieniä tällä ei ole käytännön merkitystä.

5.2.3 Pystypuiden luonti

Elementit luodaan jakamalla solmut tikapuumallin mukaan. Elementtien tikapuu koostuu pystypuista, poikkipuista ja niiden nurkkapisteissä olevista solmuista. Solmuista muodostetaan pystypuita funktiossa *CreateRails*. Pystypuut muodostetaan niiden solmujen välille, joilla on sama normaalisuoran indeksi. Tämän avulla elementeistä muodostuu luonnostaan suorakulmaisia. Jos jokin solmuista on projektiokäyrän ulkopuolella, sillä ei ole indeksiä. Tällaiset pisteet yhdistetään suoraan lähimpään vastakkaiseen solmuun *GetClosestNode*-funktion avulla. Pystypuut kootaan samanlaisiin jonoihin kuin solmutkin.



Kuva 26. *Pohja- ja huippujonoja yhdistävät pystypuut*

Kuva 26 esittää valmiita pystypuita. Solmulla 4 on sama normaalin indeksi kuin solmulla 14, joten niiden välille muodostetaan pystypuu. Solmulla 15 ei ole normaalin indeksiä tai normaalille ei löydy vastinetta pohjajonosta. Tämän vuoksi sille etsitään vastine pohjajonosta *GetClosesNode*:lla. Paras vastine on solmu 4. Myös näiden solmujen välille luodaan pystypuu, mikä muodostaa solmun 4 kohdalle haarakohdan.

Pystypuiden haarakohtiin muodostetaan myöhemmin pystysuuntainen kolmio. Pystysuuntaisilla kolmioilla voidaan muuttaa verkon tiheyttä niissä kohdissa, joissa pohjasolmujen

ja huippusolmujen lukumäärissä on eroa. Tämä on tarpeellista kohdissa, joissa rajapinnoissa on äkillinen käännös. Jos pohja- ja huippujonojen valinnassa olisi tehty virhe, ylimääräinen solmu olisi voinut saada pystypuut risteämään.

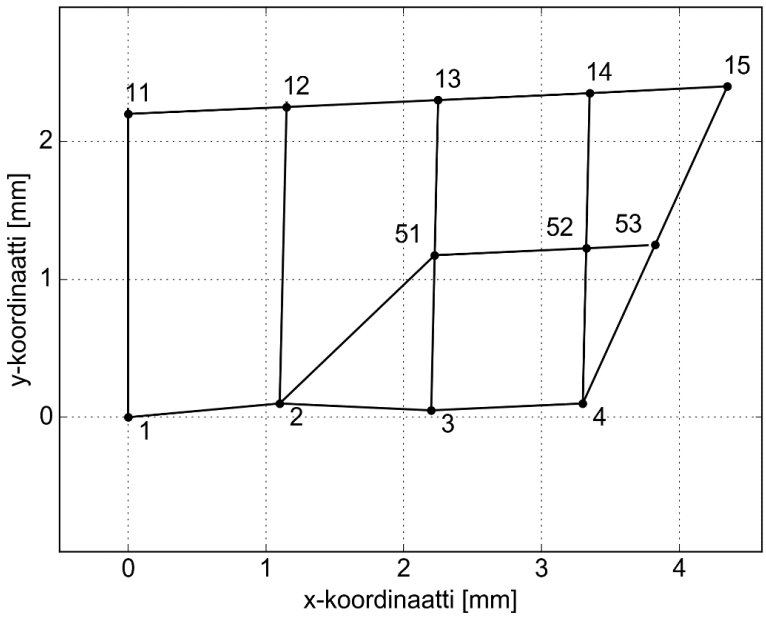
5.2.4 Pystypuiden välisolmujen luonti

Jos solmujen välinen etäisyys ylittää raja-arvon, solmujen välille muodostetaan yksi tai useampia välisolmuja ja pystypuu pilkotaan näiden välisiin osiin. Kuvassa 19 välisolmut 51, 52 ja 53 on muodostettu pystypuiden keskelle. Jos välisolmuja tarvitaan enemmän kuin yksi, ne luodaan pystypuulle tasan jakautuneina. Välisolmujen luonti tehdään funktiossa *CreateSplitNodes*. Välisolmujen koordinaatit saadaan päätysolmuista helposti painotetulla keskiarvolla. Uudet katkaistut pystypuut lisätään jonoihin alkuperäisen pystypuun tilalle. Luotaessa vahvikekuitujen kumielementtejä pystypuihin muodostetaan aina yksi välisolmu. Samalla nämä välisolmut poimitaan talteen. Myöhemmin näiden välisolmujen välille voidaan muodostaa vahvike-elementtejä.

Vierekkäisten alueiden välisolmujen tulee olla yhteneviä. Jos yhteen liitettävillä alueilla olisi erilainen määrä välisolmuja, elementtien väliin syntyisi aukkoja. Tämän vuoksi alueen pitää pystyä perimään naapurialueelta sen käyttämä päätyjako. Esimerkiksi kuvassa 18 solmun 53 kohdalla pitäisi olla naapurijoukossakin välisolmu, jotta malliin ei muodostu aukkoa. Jos viereisessä alueessa olisi valmiina solmu tällä paikalla, solmua 53 ei tarvitse luoda, vaan sen sijaan voitaisiin käyttää olemassa olevaa solmua.

5.2.5 Poikkipuiden luonti

Seuraavaksi solmuille muodostetaan poikkipuut. Poikkipuut luodaan funktiossa *CreateRungs*. Niitä luotaessa kahden vierekkäisen pystypuun solmuille etsitään yhteydet. Vähemmän solmuja sisältävästä pystypuusta muodostetaan pohjajono ja enemmän solmuja sisältävästä pystypuusta huippujono. Toisin kuin funktiossa *CreateRails,* poikkipuita ei liitetä suoraan saman normaalin solmuun, vaan siinä käytetään apuna Ohjelmassa 2 esitettyä funktiota *WriteRailRecipes*. Jos pystypuiden välisolmujen lukumäärissä on eroja, verkkoon täytyy luoda myös vaakasuuntaisia kolmioita. Vaakasuuntaisilla kolmioilla voidaan muuttaa verkon elementtikerroksien lukumäärää. Se on tarpeellista, kun esimerkiksi renkaan pintakerros ohenee keskeltä olkapäätä kohti.



Kuva 27. *Poikkipuiden luominen välisolmuille*

Kuvassa 27 on luotu poikittaisjaot uusien välisolmujen väliin. Lisäksi solmu 2 on yhdistetty välisolmuun 51. Vaihtoehtoisesti solmun 2 sijaan olisi voitu käyttää solmua 12. Tämä olisi luonut kolmioelementin lähemmäs pintaa. Kokoonpuristumattomat kolmioelementit voivat tilavuuslukittua, jolloin niiden siirtymät ja jännitykset häviävät. Tämä voi aiheuttaa niitä ympäröiviin elementteihin paikallisia jännityspiikkejä. Rengasmallin kontaktipinnoilla nämä jännityspiikit voivat vaikuttaa mm. kontaktien kitkaehtoihin. Siksi kolmioelementit halutaan mahdollisimman syvälle kauas kontaktipinnoista.

Useamman kuin yhden kolmion lisääminen on harvinaista, mutta ohjelmallisesti tämä tulee ottaa huomioon. Elementtejä muodostettaessa huomataan, että pystypuiden solmujen lukumäärästä voidaan helposti laskea tarvittavien elementtien lukumäärä:

$$tris = |n_1 - n_2|, \qquad quads = \min(n_1, n_2) - 1 \tag{56}$$

jossa *tris* on tarvittavien kolmioelementtien ja *quads* tarvittavien neliöelementtien lukumäärät, n_1 ensimmäisen ja n_2 toisen pystypuun solmujen lukumäärät.

```
def WriteRailRecipe(tris,quads):
  """Laadi resepti elementtijaolle."""
  # Muodosta neliöistä sana.
  output = quads * 'q'
  # Muodosta neliöistä suurin mahdollinen puolittava sana.
  qpart = int(quads/2)*'q'
  # Käy läpi kaikki kolmiot
  for t in range(tris):
    while True:
      # Löytyykö tulosteesta yhtenäinen sana?
      if (output.find(qpart *2) <> -1) and (qpart <> ''):
        # Lisää neliöiden keskelle kolmio
        output = output.replace(qpart*2, qpart +'t'+ qpart,1)
        break
      else:
        # Pienennä osalistauksen kokoa ja jatka yrittämistä
        qpart = qpart[:-1]
        # Jos osalistaus on tyhjä, lisää kolmio loppuun
        if qpart == '':
          output += 't'
          break
  return output
```

Ohjelma 2. Elementtijaon määrittäminen elementtien lukumäärän perusteella

Elementtien jakautuminen voidaan selvittää ohjelman 2 mukaisesti funktiolla *WriteRailRecipe*. Funktiolle syötetään kolmioelementtien lukumäärä *tris* ja neliöelementtien lukumäärä *quads*. Funktio tulostaa kolmioiden ja neliöiden järjestyksen tekstimuodossa. Funktio koostaa neliöistä sanan (rivi 4), esimerkiksi *qqqqq*, missä yksi *q* tarkoittaa neliöelementtiä. Seuraavaksi muodostetaan puolitettu sana (rivi 6), joka olisi tässä esimerkissä *qq*. Tämän jälkeen kolmioita *t* aletaan lisäämään niiden kohtien keskelle, joissa puolitettu sana *qq* esiintyy kaksi kertaa peräkkäin (rivi 13). Jokaisen lisäyksen jälkeen puolitettu sana lyhenee yhdellä neliöllä (rivi 17). Lopulta jos puolitettu sana on tyhjä, mutta kolmioita olisi vielä jäljellä, ne lisättäisiin tulosteen loppuun (rivi 20).

Taulukko 5. *Esimerkkejä elementtien jakautumisesta WriteRailRecipe-funktiossa*

tris	0	1	2	4
quads	2	3	4	3
WriteRailRecipe	QQ	QTQQ	QTQTQQ	QTQTQTT

Taulukossa 5 esitetään *WriteRailRecipe*-funktion tulosteita eräillä lähtöarvoilla. Kolmiot pyritään ensisijaisesti panemaan neliöiden väliin ja lähemmäs pohjaa kuin huippua. Jos kolmioiden lukumäärä kasvaa liian suureksi, jakoa ei voida enää järkevästi säilyttää. Taulukon viimeinen esimerkki vastaa tilannetta, jossa ensimmäisessä pystypuussa olisi neljä ja toisessa kahdeksan solmua. Näin suuria verkon muutoksia ei käytännössä voi tapahtua.

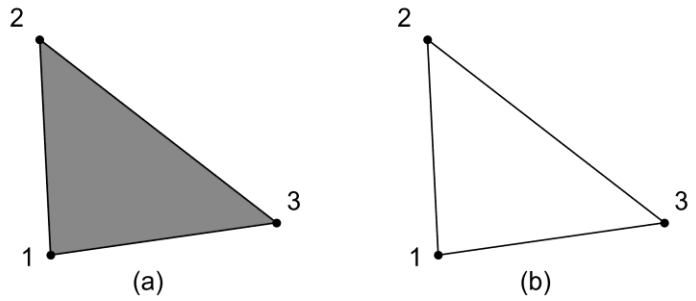
Tulosteita käsiteltäessä edetään vasemmalta oikealle ja samalla luodaan poikkipuita pohjajonosta huippujonoa kohti. Neliöelementin kohdalla kummaltakin pystypuulta poimitaan kaksi solmua. Kolmioelementin kohdalla enemmän solmuja sisältävältä pystypuulta poimitaan kaksi solmua ja vähemmän solmuja sisältävältä yksi solmu.

5.2.6 Elementtien luonti

Lopulta pystypuiden ja poikkipuiden perusteella muodostetaan elementtejä. Kolmioelementeillä ohjelma yrittää poimia yksittäisen pisteen kahdesti. Poimimalla solmut *set*-objektiin, niistä saadaan nopeasti karsittua ylimääräiset pisteet ilman erillistä tarkastusta. Jos poimittuja solmuja on neljä, muodostetaan neliöelementti, jos niitä on kolme, tehdään kolmioelementti.

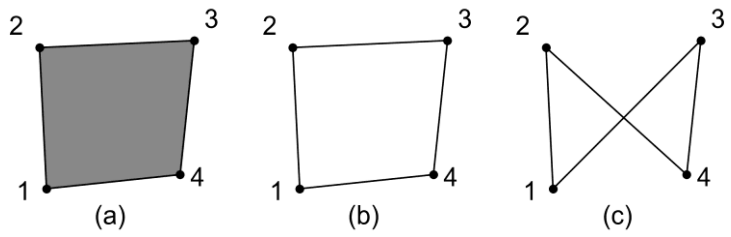
Elementtejä luotaessa viimeinen huomioitava asia on solmujen järjestys. Abaqus kääntää elementin normaalia elementin solmujen järjestyksen mukaan (Abaqus 6.14 Documentation, 2014). Jos solmut on järjestetty vastapäivään, elementin normaali on tasosta ulospäin. Tasomaisella kappaleella kaikilla elementeillä pitää olla sama normaalisuunta, jotta mallin analyysi tuottaisi järkeviä tuloksia.

Kuvassa 28a kolmioelementin solmujen järjestys on $(1, 2, 3)$. Solmujen järjestys on myötäpäivään, jolloin elementin normaali osoittaa kuvan tasoa kohti. Vastaavasti kuvassa 28b solmujärjestys on $(3, 2, 1)$. Järjestys on vastapäivään ja elementin normaali osoittaa kuvan tasosta ulospäin. Järjestyksen ensimmäisellä solmulla ei ole merkitystä. Solmujärjestykset$(1, 3, 2)$, $(2, 1, 3)$ ja $(3, 2, 1)$ ovat kaikki samanarvoisia.



Kuva 28. *Kolmioelementin solmujen järjestys vaikuttaa sen normaalisuuntaan.*

Neliöelementille pätevät samat normaalisäännöt kuin kolmioelementillekin. Lisäksi neliöelementillä solmujen järjestys voi tuottaa tiimalasin muotoisia elementtejä, joita halutaan välttää. Kuvassa 29 esitetään neliöelementille solmujärjestyksen vaikutukset. Kuten kolmioelementtien tapauksessa, 29a vastaa myötäpäivään etenevää solmujärjestystä $(1, 2, 3, 4)$, jonka ansiosta elementin normaali osoittaa kuvan tasoa kohti. 29b solmujärjestys on $(4, 3, 2, 1)$ ja elementin normaali osoittaa kuvan tasosta ulos. Viimeisenä 29c nähdään, että solmut muodostavat itsensä lävistävän nelikulmion. Tällöin elementin solmujen järjestys on $(4, 3, 1, 2)$ ja elementin normaali osoittaa kuvan tasosta ulos.



Kuva 29. *Neliöelementin solmujen järjestys vaikuttaa elementin normaalisuunnan lisäksi myös elementin muotoon.*

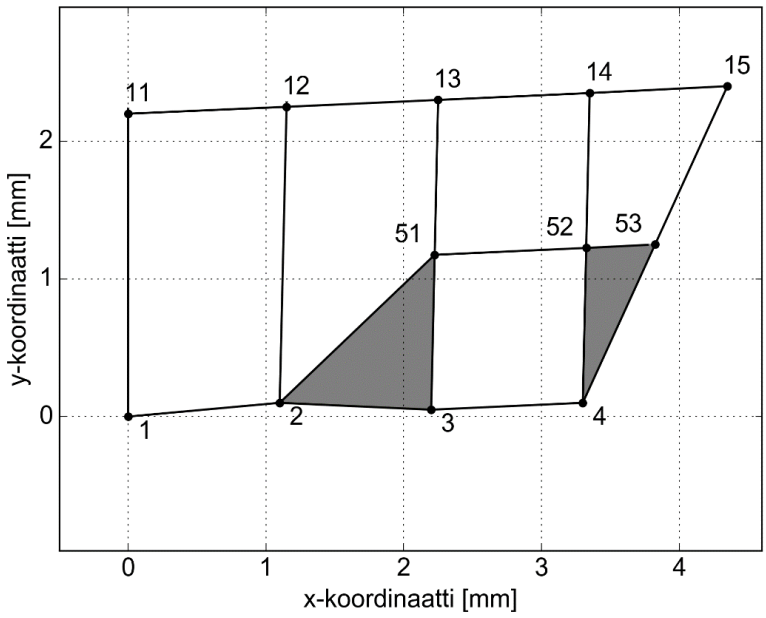
Elementin normaali- ja muotovirheiden estämiseksi elementin solmut pitää järjestää uudelleen ennen elementin luontia. Molemmat virheet saadaan korjattua yhdellä operaatiolla:

```
x0,y0,_ = nodes[0].coordinates
nodes[1:] = sorted(nodes[1:], key=lambda n:NodeSlope(x0,y0,n)),
```

jossa järjestettävät solmut kuuluvat listaan *nodes* ja listan ensimmäisen solmun koordinaatit ovat *x0 ja y0*. Listan muiden solmujen järjestämiseen käytetään funktiota *NodeSlope*, jolle syötetään listalta solmu *n*. Funktio valitsee mielivaltaisesti listan ensimmäisen solmun, laskee sen ja listan muiden solmujen muodostamien janojen väliset kulmat ja järjestää solmut tämän kulman mukaiseen nousevaan järjestykseen. Vektorien kulmat saadaan nopeasti laskettua funktiolla

```
def NodeSlope(x0,y0, node):
  # Pisteiden solmujen väliset koordinaattien erotukset
  dx, dy = node.coordinates[0]-x0, node.coordinates[1]-y0
  # Palauta kulma karteesisessa koordinaatistossa
  return atan2(dy,dx)
```

jossa *x0* ja *y0* ovat jälleen ensimmäisen solmun koordinaatit, *dx* ja *dy* ovat koordinaattien erotukset ja ensimmäinen solmu on *node*. Ensimmäisen solmun valinnalla ei ole väliä, koska *NodeSlope* osaa järjestää solmut oikein riippumatta *atan2*:n etumerkistä.



Kuva 30. *Verkotettu tikapuumalli*

Lopulta ohjelma saa verkotettua pysty- ja poikkipuiden osoittaman alueen. Kuvassa 30 kolmiot on korostettu harmaalla. Pystypuiden pituuden ylittäessä välisolmujen pituustoleranssin, pystypuulle muodostetaan uusi solmu. Tämän pituustoleranssin valinnalla voidaan vaikuttaa kolmioiden sijaintiin. Kuvan tapauksessa toleranssin rajaksi on valittu 2,25 mm, jolloin ensimmäinen välisolmut muodostetaan kuvan pystypuille 3-14, 4-14 ja 4-15. Jos toleranssiksi olisi valittu 2,3 mm, pystypuulle 3-13 ei olisi muodostettu välisolmua. Tällöin molemmat kolmiot olisi muodostettu vierekkäin.

Kun yksi alue on verkotettu, siirrytään seuraavaan alueeseen. Kun kaikki mallin osat on verkotettu, verkon osat liitetään joukkoihin materiaalin perusteella.

6. TULOKSET JA NIIDEN TARKASTELU

Automaattinen verkotus toimii nopeasti. Taulukossa 6 on jaoteltu mallin luomisen tehtäviin kuluva aika verkotettaessa kuvan 34 mukaista renkaan poikkileikkauksen puolikasta. Taulukossa vasemmalla on uuden automaattisen verkottajan ja oikealla manuaalisen verkotuksen tehtävien kesto. Automaattinen mallin luonti kuvaa tietokoneen toimintaan kuluvaa aikaa. Manuaalinen mallin luonti on vastaavien tehtävien suorittamista käyttäjän toimesta. Koska toimet ovat rutiininomaisia, tietokone pystyy suorittamaan niitä merkittävästi nopeammin kuin käyttäjä. Automaattisessa mallin luonnissa on lukuisia lyhyitä vaiheita. Manuaalisessa mallin luonnissa vaiheita on vähemmän. Automaattisen verkotuksen lopputuloksena syntyvä rengasmalli vaati kuitenkin vielä jälkikäsittelyä.

Taulukko 6. *Mallin luomiseen kuluva aika*

Automaattinen mallin luonti	**Kesto**
Ohjelmiston käynnistys	1,7 s
CAD-lisenssin varaaminen	12,1 s
Pistegeometrian lukeminen	9,3 s
Solmujen luonti ja verkottaminen	14,2 s
Mallin viimeistely	3,0 s
Yhteensä	**40,3 s**

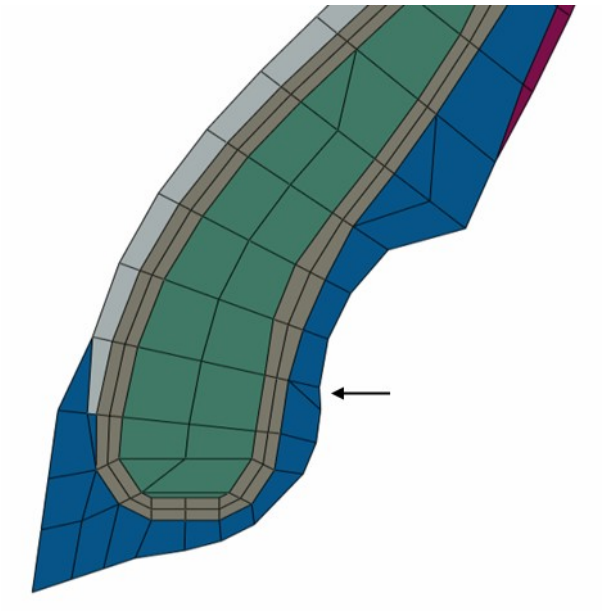
Manuaalinen mallin luonti	**Kesto**
Geometrian luonti	34 min
Verkotus ja osiointi	53 min
Yhteensä	**87 min**

Automaattisessa verkotuksessa eniten aikaa kuluu CAD-ohjelmistolisenssin varaamiseen, pistegeometrian lukemiseen ja itse verkotukseen. Lisenssin varaus ja geometrian lukeminen riippuvat CAD-ohjelmistosta ja tavasta, jolla siihen yhdistetään. Etenkin lisenssin varaamiseen kuluva aika on kiinteä osa ohjelmiston toimintaa. Vaihtamalla ohjelmistoa olisi mahdollista pudottaa kokonaisaika murto-osaan. Kestot voivat vaihdella riippuen verkottamiseen käytetyn tietokoneen ja lisenssiserverin kuormituksesta. Varsinainen solmujen luonti ja verkotus tapahtuu verrattain nopeasti.

Käsin tehtävän mallin verkotukseen kuluu perinteisesti tunteja. Ongelmatapauksissa verkotuksessa ilmenevät ongelmat pakottavat palaamaan takaisin geometrian luontiin. Mitattaessa mallin luonti saatiin tehtyä puolessatoista tunnissa, koska vastaan ei tullut merkittäviä ongelmia. Geometrian luontiin kului puoli tuntia ja verkotukseen vajaa tunti.

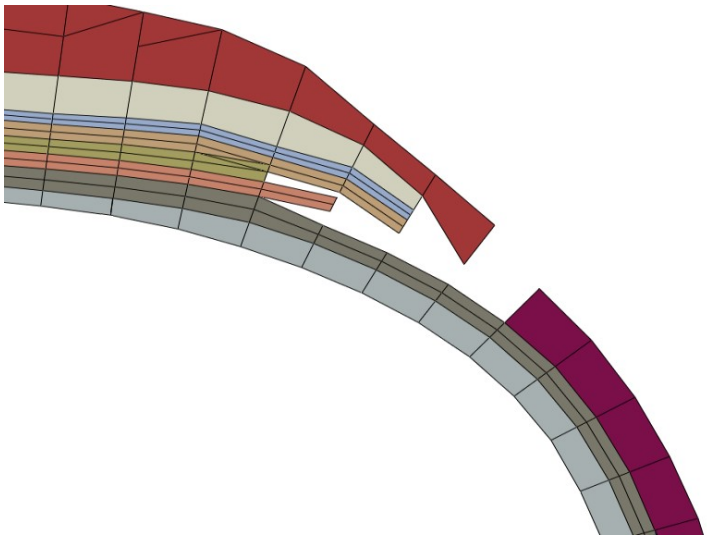
Automaattisesti luotua mallia piti vielä korjata käsin. Tähän kului ylimääräiset 10 minuuttia. Kuvassa 31 näytetään korjausta vaativia puutteita renkaan jalka-alueella. Verkottaja ei osaa periä välisolmua referenssisuunnan muuttuessa. Tämän takia renkaan kaapelin ja runkokoordin rajapintaan syntyy katkos elementtijaossa. Käytännössä tässä kohdassa on aukko, jossa neliöelementtien yksi nurkkasolmu ei ole liittynyt viereiseen elementtiin. Elementtijaon muuttuessa kahdesta yhteen verkottaja on tehnyt kaapelin kohdalle kolmion ja

ohuen neliöelementin. Ohut neliöelementti pitää katkaista ja syntynyt solmu liittää runkokoordin elementteihin. Virheen ohjelmallinen korjaus vaatisi välisolmun perimisen toisesta suunnasta tai välisolmun toleranssin muuttamisen paikallisesti. Kaapelivirheen lisäksi verkottaja on luonut renkaan kantapäälle ylimääräisen kolmion vannekontaktin alueelle nuolen osoittamaan kohtaan. Tämä kolmio kannattaa tuhota poistamalla sen alin solmupiste.



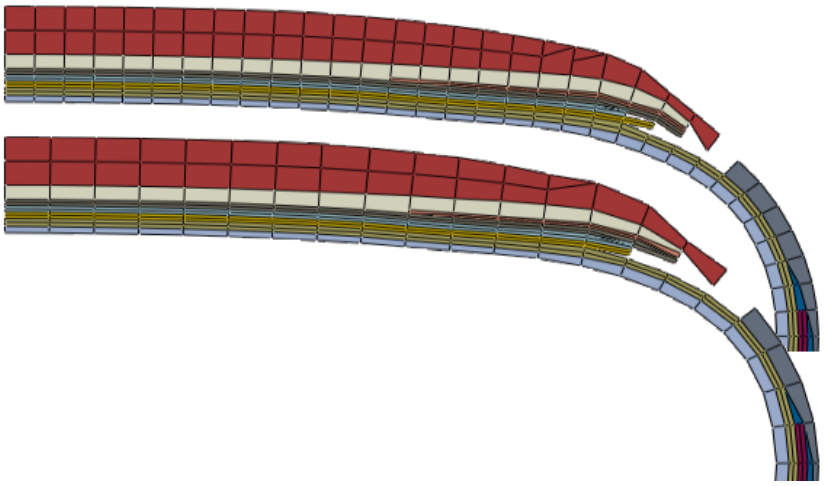
Kuva 31. *Elementtivirhe jalka-alueella*

Verkottamisessa vaikeinta on kertoa verkottajalle, mitkä solmujoukot pitää yhdistää. Joissakin tapauksissa geometriamallista ei löydy suoraan sopivia piirteitä, joita voisi käyttää verkon luomisen pohjana. Tällöin olemassa olevia piirteitä täytyisi pilkkoa ja yhdistellä ennen verkon luomista. Kuvassa 32 näkyy renkaan olkapäälle jäävä aukko, johon ei löydy helposti yhtä piirrettä.



Kuva 32. *Aukko renkaan olkapäällä.*

Kuvan 32 tapauksessa puuttuvat solmut ja elementit tulee luoda käsin valmiiden elementtien pohjalta. Uudet elementit saadaan yhdistettyä aukon reunojen solmuihin. Jakamalla nämä uudet elementit osiin saadaan solmupisteitä luotua tarvittaviin paikkoihin. Lopuksi uudet elementit lisätään naapurialueiden elementtijoukkoihin. Käsin tehtävän korjauksen välttämiseksi renkaan geometriamalliin pitäisi lisätä uusia piirteitä, joiden avulla aukon reunat voitaisiin liittää toisiinsa. Kuvassa 33 näkyy myös virheellinen, ylimääräinen kolmio renkaan olkapäällä. Tämäkin kolmio on helppo poistaa tuhoamalla sen alin solmupiste.



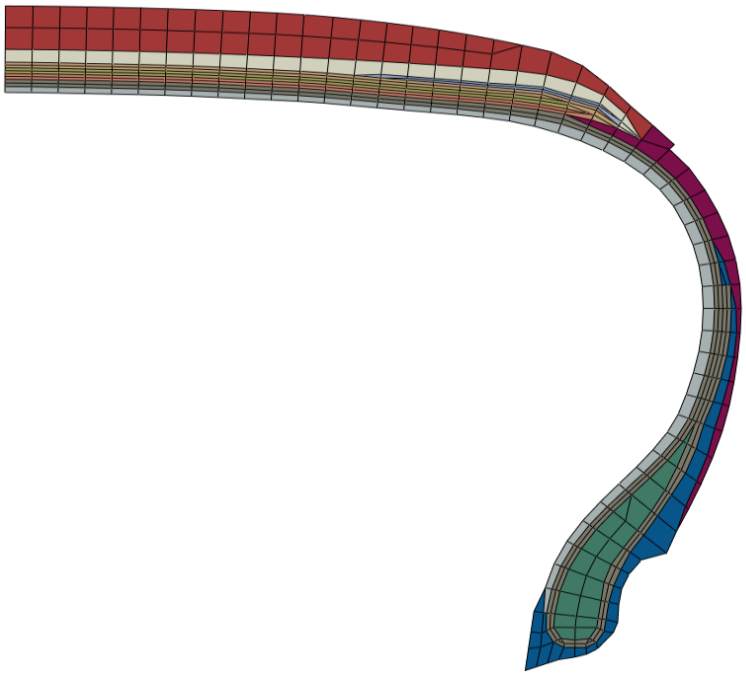
Kuva 33. *Elementtikoon vaikutus komponenttien limityksiin*

Koordinaattien normalisointi pakottaa elementit suorakulmaisiksi. Samalla se myös muuttaa komponenttien limityksiä. Kuvassa 33 näkyy ylempänä verkko, joka on luotu 4 mm elementtikoolla ja alla 6 mm koolla. Jos komponenttien todellinen limitys on 5 mm, kumpikin näistä tapauksista vääristää mallin geometriaa. 4 mm referenssipisteiden jako voi

liikuttaa komponentin laskennallista päätypistettä enimmillään ±2 mm ja 6 mm jako ±3 mm. 6 mm jako on tässä tapauksessa niin karkea, että vöiden päätypisteet luodaan samalle normaalille, jolloin niiden limitys katoaa.

Limitysmuutoksien mahdollisuus on olemassa myös käsin tehtävässä mallissa, mutta automaattinen verkottaja tuo ne selvemmin esiin. Limitysmuutokset voidaan minimoida, jos käytetään samaa referenssipisteiden jakoa ja elementin kokoa. Tällöin elementtien solmupisteiden normalisoimattomat koordinaatit ovat mahdollisimman lähellä normaalisuoria ja projisointivirheitä syntyy vähemmän.

Kuvassa 34 on valmiin renkaan poikkileikkauksen puolikas luotuna 5 mm elementtijaolla. Kuvasta nähdään, että syntyneet elementit ovat tasan jakautuneita ja hyvin suorakulmaisia.



Kuva 34. *Valmis laskentamalli 5 mm elementtikoolla.*

Taulukossa 7 on vertailtu käsin tehtävän mallin ja automaattisesti tehtävän mallin laskenta-aikaa. Tulokset eivät ole keskenään täysin vertailukelpoisia, koska Ojalan käyttämää laskentakonetta on päivitetty useita kertoja. Laskentakoneiden A ja B tehoista ei ole säilynyt tarkkaa tietoa. Myös käytetyt materiaaliarvot ja renkaan mallit ovat muuttuneet, joten nämä tulokset ovat korkeintaan suuntaa-antavia. Automaattista verkottajaa on vertailtu käsin tehtävään laskentamalliin laskentakoneella C, samalla renkaalla ja samoilla materiaaliarvoilla, joten nämä tulokset ovat keskenään vertailukelpoisia.

Kuvassa 34 esitetty malli on taulukossa viimeisenä. Tuloksista voidaan huomata, että uusimmilla laskentakoneilla elementtien määrää ja mallin tarkkuutta voidaan kasvattaa laskenta-ajan pysyessä murto-osassa Ojalan vastaavaan nähden. Samalla prosessorien lukumäärä voidaan pitää pienenä. Tämä laskee Abaqus-ohjelman lisenssikustannuksia.

Taulukko 7. *Verkotus ja laskenta-aika mallin tyypin mukaan*

Mallin tyyppi	Elementtejä	Laskenta-kone	CPUs	Laskenta-aika
Käsintehty puolikas rengas (Ojala, 2003, s. 37)	33871	A		10 h
Käsintehty kokonainen rengas		B	32	4 h 6 min 58 s
Käsintehty puolikas rengas	109752	C	5	35 min 11 s
Automaattinen puolikas rengas	113482	C	5	34 min 15 s

Lähtöoletuksena oli, että automaattinen malli olisi parempi kuin vastaava käsin tehty malli. Automaattisesti tehdyssä mallissa oli noin 3,4% enemmän elementtejä kuin käsin tehdyssä mallissa. Samalla sen laskenta-aika oli noin 2,7% lyhyempi. Lyhyempi laskenta-aika voi johtua elementtien paremmasta muodosta verrattuna käsin verkotettuun malliin. Käytännössä mallien erot eivät kuitenkaan ole merkittäviä. Vaihtelut serverin kuormituksissa voivat kääntää laskenta-ajat toisin päin. Tuloksista voidaan kuitenkin päätellä, että uusi automaattinen malli on laskennan suhteen täysin verrannollinen käsin tehtyyn malliin.

YHTEENVETO

Tämän työn tarkoituksena oli kehittää ohjelmisto, joka pystyy luomaan pyörähdyssymetrisiä renkaan laskentamalleja. Työssä tutkittiin, miten Abaqus käsittelee nelikulmioelementtejä ja etenkin niiden tiimalasikontrollia. Tarkastelun pohjalta laadittiin algoritmi, joka varmistaa oikeanlaisten elementtien luomisen. Algoritmin avulla verkotettavalle alueelle luotiin solmut ja elementit.

Nelikulmioelementtien tarkastelussa vertailtiin teoreettisen testikappaleen taipumia puhtaassa taivutuksessa. Pyörähdyssymetriselle laatalle laskettiin analyyttiset ja numeeriset taipumat kahdella eri painekuormalla. Analyyttiset taipumat laskettiin lineaarisella pienen taipuman kaavalla ja epälineaarisella suuren taipuman kaavalla. Numeeriset taipumat simuloitiin tavallisella nelikulmioelementillä, ali-integroidulla nelikulmioelementillä ja stabiloidulla, ali-integroidulla nelikulmioelementillä. Yllättäen suorakulmaiset stabiloidut elementit toimivat tarkasti puhtaan taivutuksen tapauksessa riippumatta elementtien tiheydestä. Jos elementit ovat suorakulmaisia, elementtijaon saa valita vapaasti. Elementtijako kannattaa valita siten, että se kuvaa taustalla olevaa geometriaa mahdollisimman tarkasti. Jatkotutkimuksessa olisi hyvä tarkastella lähemmin, miten elementtien muodon vääristyminen vaikuttaa laskenta-aikaan ja tarkkuuteen. Myös kolmioelementtien käyttäytyminen jäi tässä työssä tutkimatta.

Ohjelmiston toteutuksessa käytiin läpi suunnitellun verkotusalgoritmin toimintaa. Mallin luomisen kannalta tärkeimmiksi ratkaisuiksi huomattiin:

1. Koordinaattien normalisointi
2. Koordinaattien uudelleenjärjestäminen
3. Lähekkäisten solmupisteiden yhdistäminen
4. Normaalivektorien käyttö verkottamisessa

Normalisointi pakottaa neliöelementit suorakulmaisiksi, mikä parantaa elementtien toimintaa. Toisaalta käyttäjän täytyy tarkistaa, että normalisoidut elementit noudattavat geometriamallin mukaista komponenttien limitystä. Koordinaattien uudelleenjärjestäminen takaa, että toisiinsa liitettävät solmut on valittu oikein. Solmujen järjestäminen myös kääntää elementtien normaalit samaan suuntaan. Koordinaattien pyöristysvirheet saavat aikaan lähekkäisiä solmupisteitä, joita yhdistämällä estetään pienten, vääristyneiden elementtien luonti. Uudelleenkäyttämällä normaalivektoreita verkotuksessa säästetään aikaa ja taataan, että elementtien suorakulmaisuus säilyy.

Tässä työssä luotu verkottaja ei ole vielä valmis. Verkottaja tekee vielä aukkoja elementtien väliin, missä reunaviivojen tarkastelusuunta kääntyy. Ohjelmisto ei myöskään osaa vielä periä vierekkäisten alueiden solmujakoja, mikä aiheuttaa virheellisiä kolmioita solmujaon muutoskohtiin. Nämä muutoskohdat pitäisi pystyä tunnistamaan myös ohjelmal-

lisesti. Vaikeinta mallien kehittämisessä on geometrian määrittelyn teko renkaan geometriamallista. Tämän työn helpottamiseksi geometriamalliin tulisi luoda uusia piirteitä, joita voisi hyödyntää laskentamallissa.

Työn tavoitteena oli helpottaa mallien luomista ja niiden vertailtavuutta. Uusi ohjelmisto nopeuttaa mallien luomista merkittävästi. Laskentamallin verkottamiseen kuluva aika putosi tunneista minuutteihin. Mallin laskentaan kuluvaa aika pysyi ennallaan. Mallien vertailtavuus parani, koska automaattinen verkottaja muodostaa mallit systemaattisemmin kuin ihminen. Kasvanut systemaattisuus poistaa käyttäjän vaikutuksia laskennan tulokseen.

LÄHTEET

Abaqus 6.14 Documentation (2014). Saatavissa: http://abaqus.software.polimi.it/v6.14/index.html.

Belytschko, T. ja Bindeman, L. (1993) “Assumed Strain Stabilization of the Eight Node Hexahedral Element”, *Computer Methods in Applied Mechanics and Engineering*, 105. p., ss. 225-260, doi: 10.1016/0045-7825(93)90124-G

Blazek, J. (2015) “Chapter 11 - Principles of Grid Generation”, *Computational Fluid Dynamics: Principles and Applications*, 3. p., ss. 357-393, doi: 10.1016/B978-0-08-099995-1.00011-7

Chandrupatla, T. ja Belegundu, A. (2002) *Introduction to Finite Elements in Engineering,* 3. p., New Jersey: Prentice Hall

Flanagan, D. ja Belytschko, T. (1981) “A Uniform Strain Hexahedron and Quadrilateral with Orthogonal Hourglass Control”, *International Journal for Numerical Methods in Engineering*, 17. p., ss. 679-706, doi: 10.1002/nme.1620170504

Hakka Green 2 – mainosmateriaali, Nokian Renkaat Oyj

Hueck, U., Reddy, B. ja Wriggers, P. (1994) “On the stabilization of the rectangular 4-node quadrilateral element”, *Communications in Numerical Methods in Engineering*, 10(7), ss. 555-563, doi: 10.1002/cnm.1640100707

Laitila, A. (2018) *Henkilöautonrenkaan vierinvastuksen tutkiminen elementtimenetelmällä.* Diplomityö, Tampereen teknillinen yliopisto

Lehtoranta, I., Vesiliirron tutkiminen elementtimenetelmällä. Diplomityö, Tampereen teknillinen yliopisto, Konetekniikan koulutusohjelma 2008.

Leister, G. (2015) ”The Role of Tyre Simulation in Chassis Development – Challenge and Opportunity”, Proceeding of the 4th Internation Tyre Colloquium, ss. 1-10, University of Surrey

Ojala, J. (2003) *Henkilöauton renkaan analysointi elementtimenetelmällä.* Diplomityö, Tampereen teknillinen yliopisto

Pacejka, H. (2012) *Tire and Vehicle Dynamics,* 3. p., Butterworth-Heinemann, doi: 10.1016/c2010-0-68548-8

Selig, M. ym. (2017) ”Tire Simulation Models of the Future”, *VDI-Berichte,* 2296, ss. 33-45, VDI Düsseldorf

Timoshenko, S. ja Woinowsky-Krieger, S. (1987) *Theory of Plates and Shells*, *2*. p. New York: McGraw-Hill Book Company

LIITE A: ELEMENTTIEN MUODOSTAMINEN