Aalto-yliopisto
Insinööritieteiden
korkeakoulu

Tuomas Tiainen

Automaattinen mallinnustehtävien tarkastus konetekniikan CAD-opetuksessa

Diplomityö, joka on jätetty opinnäytteenä tarkastettavaksi diplomi-insinöörin tutkintoa varten Espoossa 27.11.2017.

Työn valvoja

Prof. Petri Kuosmanen

Työn ohjaajat

TkT Panu Kiviluoma

DI Kaur Jaakma

Aalto-yliopisto
Insinööritieteiden
korkeakoulu

Aalto-yliopisto, PL 11000, 00076 AALTO
www.aalto.fi
Diplomityön tiivistelmä

Tekijä Tuomas Tiainen

Työn nimi Automaattinen mallinnustehtävien tarkastus konetekniikan CAD-opetuksessa

Koulutusohjelma Konetekniikka

Pääaine Konetekniikka **Pääaineen koodi** ENG-25

Työn valvoja Prof. Petri Kuosmanen

Työn ohjaajat TkT Panu Kiviluoma, DI Kaur Jaakma

Päivämäärä 27.11.2017 **Sivumäärä** 64 **Kieli** Suomi

Tiivistelmä

Automaattisia tehtävien tarkastustyökaluja käyttämällä opiskelijoille on mahdollista antaa palautetta tehtävistä välittömästi niiden palautuksen jälkeen. Järjestelmät vähentävät opetushenkilökunnan rutiininomaiseen työhön käyttämää aikaa ja mahdollistavat opiskelijoiden suoritusten viiveettömän seurannan. Verkossa toimivien tehtävien tarkastustyökalujen ja oppimisympäristöjen kehitys liittyy myös MOOC-trendiin, jossa yliopistot ovat avanneet verkkoon kaikille avoimia automaattisesti toimivia kursseja.

Ohjelmoinnin opetuksessa on jo pitkään käytetty opiskelijoiden tehtäviä automaattisesti tarkastavia ohjelmia. Tässä työssä tarkasteltiin mahdollisuuksia toteuttaa vastaava automaattinen tarkastusjärjestelmä konetekniikan alan CAD-opetuksen mallinnustehtävien tarkastukseen. Työ sisältää kirjallisuuskatsauksen konetekniikan CAD-opetuksen nykytilaan, kolmiulotteisen tiedon tallennuksen ja vertailun teoriaan sekä nykyisiin oppimisympäristöihin ja automaattisiin tarkastusjärjestelmiin.

Työssä määritellään mallinnustehtävien tarkastusprosessi, kartoitetaan vaihtoehtoja automaattisen tehtävien tarkastuksen toteuttamiseksi ja valitaan tekniset ratkaisut automaattisen tarkastusjärjestelmän prototyypin toteuttamiseksi. Osana työtä toteutettiin tarkastusjärjestelmän prototyyppi, jota kokeiltiin onnistuneesti Aalto-yliopiston CAD-kurssin tehtäväkierroksella. Järjestelmän toiminnasta ja opiskelijoiden oppimiskokemuksista toteutettiin kyselytutkimus, jossa opiskelijoilta saatiin samansuuntaisia tuloksia kuin aikaisemmissa tutkimuksissa: ajasta ja paikasta riippumatonta tehtävien palautusmahdollisuutta pidetään tärkeänä. Käyttökokemusten ja kyselyn vastausten perusteella esitetään järjestelmään parannusehdotuksia.

Avainsanat CAD-opetus, CAD , MOOC

Aalto-yliopisto
Insinööritieteiden
korkeakoulu

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi
Abstract of the master's thesis

Author Tuomas Tiainen

Title Automatic assessment of mechanical engineering CAD modelling exercises

Degree programme Master's Programme in Mechanical Engineering

Major Mechanical Engineering **Code of major** ENG-25

Supervisor Prof. Petri Kuosmanen

Advisors D.Sc. (Tech.) Panu Kiviluoma, M.Sc. (Tech.) Kaur Jaakma

Date 27.11.2017 **Number of pages** 64 **Language** Finnish

Abstract

Automatic exercise assessment tools can be used to give instant feedback from submitted exercises to students. Automatic assesment tools also reduce the amount of routine work of the teaching staff and enable real time tracking of students' progress. The development of automatic exercise assessment tools also relates to the MOOC trend, in which universities have opened open online courses.

Automatic exercise assessment has been used for a long time in the education of computer programming. In this master's thesis the possibility to implement a similar system for automatic assessment of mechanical engineering CAD exercises is studied. The thesis contains a literature review on CAD education in general, existing automatic online learning environments and automatic assessment systems and the structuring and comparison of 3D information. The literature review also assessess different techical possibilities for implementing an automatic assessment system.

As a part of the thesis, based on the literature review results, a prototype automatic assessment system was implemented for use on Aalto university mechanical engineering CAD courses. The system was successfully tested on a master's level CAD course and a survey assessing the functioning of the system and learning experiences was conducted as a part of the research. Improvements are proposed to the system based on the user experience and survey results.

Keywords CAD education, CAD, MOOC

Esipuhe

Haluan kiittää professori Petri Kuosmasta sekä ohjaajiani Panu Kiviluomaa ja Kaur Jaakmaa hyvästä ohjauksesta.

Haluan esittää kiitokset myös Retuperän WBK:n letkumestarille, joka on mahdollistanut minulle turvallisen ja miellyttävän opiskeluajan Otaniemessä.

Lisäksi haluan kiittää A!OLE-tutkimushanketta työn rahoituksesta.

Otaniemi, 27.11.2017

Tuomas Tiainen

Sisällysluettelo

Lyhenteet

3D	3-dimensional, kolmiulotteinen
B-rep	Boundary representation, tilavuusmallin reunaesitys
CAD	Computer Aided Design, tietokoneavusteinen suunnittelu
CAE	Computer Aided Engineering, CAD:iä laajempi käsite: tietokoneen käyttö optimoinnissa ja simuloinnissa
CAM	Computer Aided Manufacturing, tietokoneavusteinen valmistus
ICP	Iterative Closest Point, algoritmi kolmiulotteisten pistepilvien sovittamiseen
IGES	Initial Graphics Exchange Specification, standardoitu neutraali tiedostoformaatti
JSON	JavaScript Object Notation, standardoitu muoto tiedon tallennukseen avain–arvo-pareina
MOOC	Massive Open Online Course, massiivinen avoin verkkokurssi
NURBS	Non-Uniform Rational B-spline
PDM	Product Data Management, tuotetiedon hallinta
STEP	ISO 10103, Standard for the Exchange of Product Model Data, standardoitu neutraali tiedostoformaatti

1 Johdanto

1.1 Tausta

Tietokoneavusteisia suunnitteluohjelmia (CAD) käytetään useilla tekniikan aloilla. Aalto-yliopistossa ja muissakin yliopistoissa CAD-mallinnuksen perusopetus on usein toteutettu kursseina, joilla opiskelijat mallintavat tehtäviä ja käyvät harjoituksissa näyttämässä ne kurssiassistenteille. CAD-mallinnuksen peruskursseilla ensimmäiset mallinnettavat kappaleet ovat niin yksinkertaisia, että ne ovat joko yksiselitteisen oikein tai väärin. Tämän vuoksi saattaisi olla mahdollista toteuttaa verkossa toimiva kurssi, jolla tehtäviä tarkastetaan automaattisesti. Kurssin toteuttamiseksi tarvitaan automaattinen tehtävien tarkastusjärjestelmä, jollaisia ei ole vielä olemassa.

Ohjelmoinnin opetuksessa on jo useiden vuosien ajan käytetty automaattisia tarkastusohjelmia, joilla opiskelijat voivat tarkastuttaa harjoitustehtäviään internetissä toimivan palvelun kautta ja saada niistä pisteitä tai palautetta välittömästi tehtävien suorittamisen jälkeen. Aalto-yliopiston ohjelmoinnin peruskursseilla ohjelmointitehtävien tarkastusohjelmat testaavat opiskelijoiden kirjoittamia ohjelmia erilaisilla syötteillä ja vertailevat ohjelman tulosteita mallitulosteisiin.

Ohjelmistokehityksessä ohjelmien automaattisella testauksella on pitkä historia ja useissa ohjelmistoprojekteissa pyritään kirjoittamaan testit kaikelle kirjoitetulle ohjelmakoodille – testeillä on mahdollista saada kiinni virheitä ja varmistua siitä, että ohjelmien toiminnallisuus on vaatimusten mukaista. Ohjelmistotestauksessa ohjelmien testausta luokitellaan eri ryhmiin. Savutestauksella tarkoitetaan nopeiden ja yksinkertaisten tarkastusten ajamista, jotta voidaan varmistua siitä, että ohjelma pääpiirteittäin toimii. Yksikkötestauksessa tavoitteena on määritellä ensin vaatimukset jokaiselle ohjelman funktiolle ja kirjoittaa niille testi ennen funktion kirjoittamista. [1, 697] Ohjelmistotestauksessa käytettäviä menetelmiä olisi mahdollista soveltaa myös mallinnustehtävien tarkastukseen.

Automaattinen tehtävien tarkastus liittyy myös Yhdysvalloista alkaneeseen MOOC-trendiin, jossa yliopistot ovat toteuttaneet ja avanneet kurssejaan verkkoon kaikkien saataville. MOOC-kurssit eli massiiviset avoimet verkkokurssit ovat internetissä järjestettäviä avoimia verkkokursseja, joiden materiaalit ovat yleensä avoimesti tarjolla internetin kautta. Automaattisen tarkastusjärjestelmän avulla olisi mahdollista toteuttaa myös CAD-opetuksen MOOC-kurssi, jollaisista ei tutkimuksen aikana löydetty olemassa olevia toteutuksia. Kun automaattinen kurssi on kerran toteutettu, kustannukset lisäopiskelijoista ovat pienet.

1.2 Tutkimusongelma

Tämän työn tarkoituksena on tarkastella mahdollisuuksia toteuttaa automaattinen tehtävien tarkastusjärjestelmä konetekniikan CAD-opetuksen mallinnustehtävien tarkastukseen. Osa tutkimusongelmaa on myös tutkia, minkälaisia tehtäviä olisi mahdollista tarkastaa automaattisesti ja kuinka suuri työmäärä tarkastuksen automatisoinnissa on. CAD-peruskurssien ensimmäiset tehtävät ovat joko yksiselitteisen oikein tai väärin, mutta vapaita kaarevia pintoja sisältävien kappaleiden tai suunnittelutehtävien automaattinen tarkastus saattaa muuttua hankalaksi. Koneenosien suunnittelussa osien vaatimuksiin vaikuttavat ilmiöt liittyvät toisiinsa, ovat monimutkaisia ja erilaisten suunnitteluratkaisujen hyötyjen ja haittojen punnitseminen ja järkevien kompromissien tekeminen on osa suunnittelijan ammattitaitoa.

1.3 Tutkimusmenetelmät

Työssä käytettävät tutkimusmenetelmät ovat:

- Kirjallisuustutkimus, katsaus CAD-opetuksen nykytilaan, MOOC-kurssien ja kolmiulotteisen tiedon tallennuksen ja vertailun teoriaan sekä CAD-ohjelmista löytyviin työkaluihin ja rajapintoihin.
- Automaattisen tarkastusjärjestelmän prototyypin toteuttaminen ja kokeilu, osana työtä toteutetaan prototyyppi, jota kokeillaan Aalto-yliopiston kurssilla.
- Käyttäjätutkimus, opiskelijoilta automaattisen tarkastusjärjestelmän kokeilun jälkeen kerättävä kyselytutkimus ja palaute.

1.4 Rajaus

Erilaisia CAD-ohjelmia on käytössä myös muilla tekniikan aloilla. Työn käytännön osuudessa käsitellään vain konetekniikan alalla käytettäviä parametrisia CAD-ohjelmia. Työssä käsitellään ensisijaisesti kolmiulotteisten mallinnustehtävien tarkastusta. CAD-ohjelmien työkalujen ja rajapintojen tarkastelu rajataan kolmeen Aalto-yliopistossa käytössä olevaan yleiseen CAD-ohjelmaan. Tehtävien tarkastusjärjestelmän prototyyppi konfiguroidaan tarkastamaan Aalto-yliopistossa käytettävää perustason parametrista CAD-mallinnustehtävää Creo Parametric 3.0 -ohjelmiston M110-versiolla.

2 Kirjallisuuskatsaus

2.1 Tietokoneavusteinen mallinnus konetekniikassa

Tietokoneavusteista mallinnusta käytetään useilla tekniikan ja taiteen aloilla. Nykyaikaisten tuotteiden muotoilu ilman tietokoneavusteisia työkaluja ei ole mahdollista, sillä tuotteissa esiintyy suorien pintojen ja niiden välisten pyöristyksien lisäksi tarkkaan muotoiltuja kaksoiskaarevia pintoja. Kaksoiskaarevien pintojen tarkkaa muotoa koskevan tiedon välittäminen käsin tai kaksiulotteisten piirustusten avulla ei ole mahdollista. Konetekniikassa tietokoneavusteisia malleja käytetään esimerkiksi tiedon välittämiseen suunnittelusta valmistukseen sekä erilaisiin simulointeihin.

Konetekniikassa osien tuotannossa pyritään *osien vaihdettavuuteen*. Osien vaihdettavuus edellyttää, että osia pystytään systemaattisesti valmistamaan samankaltaisiksi – tätä tarkoitusta varten on kehitetty erilaisia toleranssi- ja sovitejärjestelmiä sekä teknisen piirustuksen standardeja. Osan suunnittelun jälkeen on tarve välittää kaikki tarvittava tieto osan ominaisuuksista valmistukseen. Tietokoneavusteisten valmistusmenetelmien yleistyessä tämä tiedon välittäminen tapahtuu entistä useammin piirustusten sijaan erilaisilla kappaletta kuvaavilla mallitiedostoilla, ilman, että osasta siirretään tietoa paperille.

Kaksiulotteisten piirustusten ja CAD-ohjelmissa aikaisemmin yleisesti käytettyjen rautalankamallien avulla ei ole aina mahdollista esittää yksikäsitteisesti kaarevia pintoja. Tieto pinnan tarkasta muodosta menetetään, mikäli malli esitetään yksinkertaistetussa muodossa. [2] Tietokoneavusteiset valmistusmenetelmät ovat tehneet tällaisten muotojen valmistamisen mahdolliseksi, ja käyttöön on otettu menetelmiä kaarevien muotojen suunnitteluun (pintamallinnustyökalut) sekä tallentamiseen (tiedostoformaatit).

2.1.1 Tietokoneavusteisen mallinnuksen opetus

CAD-mallinnus konetekniikassa vaatii kolmiulotteista hahmottamista, ohjelmien työkalujen sekä oikeiden ja tehokkaiden työskentelytapojen hallintaa. CAD-mallinnuksen osaaminen voidaan jakaa strategiseen osaamiseen, eli mallin rakenteen suunnittelun osaamiseen ja deklaratiiviseen osaamiseen, eli työkalujen hallintaan [3].

Strategisen osaamisen merkitystä mallinnusprosessissa on tutkittu analysoimalla sellaisen tehtävän suorittamista, jossa opiskelijat suorittavat muutoksia olemassa olevaan CAD-malliin [3]. On olemassa näyttöä sen puolesta, että deklaratiivinen osaaminen siirtyy eri CAD-järjestelmien välillä – oikeiden työskentelytapojen opettaminen tietyn valmistajan järjestelmällä ei siis mene

hukkaan. Eri valmistajien ohjelmissa työtavat, työkalut ja niiden toiminta muistuttavat toisiaan.

Kyselytutkimuksen mukaan opiskelijat eivät pidä CAD-opetuksen tasoa riittävänä erityisesti strategisen osaamisen osalta – työelämässä ovat usein tarpeen muutokset muiden käyttäjien tekemiin malleihin. Mallinnustyökalujen kehityksen myötä CAD-perusopetus keskittyy edelleen liikaa deklaratiiviseen opetukseen ja yksittäisten työkalujen käytön opetteluun. [3]

CAD-opetuksen tavoitteet ovat erilaisia eri koulutusasteilla. Yliopistojen tutkinto-ohjelmissa ei pidetä edes tavoitteena opetella kaikkia mahdollisia ohjelmien työkaluja, vaan mallinnusprosessin ymmärtäminen on tärkeämpää.

Laajojen kokonaisuuksien ja monimutkaisempien osien mallinnuksessa pelkkä mallinnusohjelmiston työkalujen tekninen hallinta ei riitä. Mallien on oltava järkevällä tavalla rakennettuja ja parametrisoituja, jotta ne ovat muokattavissa jälkeenpäin. Mallien muokattavuutta on mahdollista parantaa hyvin yksinkertaisten käytäntöjen avulla, esimerkiksi nimeämällä osia ja piirteitä muokkausten vaatiman ajan on havaittu laskevan 10-20 prosentilla. [3] Hyvien CAD-käytäntöjen opettamisen merkitys yliopisto-opetuksessa korostuu verrattuna ohjelmien yksittäisten työkalujen toiminnan opettamiseen.

Yliopistotason CAD-opetuksen tavoitteena on antaa opiskelijoille tarvittavat perustaidot itsenäiseen jatko-opiskeluun. Mallin rakenteeseen tulisi pystyä välittämään "design intent", eli suunnittelun tarkoitus – tämä voi toteutua esimerkiksi parametrisoimalla mallin hallitsevat piirteet. Tuotteita suunnittelevien insinöörien on myös osattava tunnistaa kappaleiden muodosta ne asiat ja piirteet, joilla on merkitystä kappaleen fyysisten ominaisuuksien kannalta [4].

Pääsääntöisesti CAD-mallinnusta opetetaan suomalaisissa korkeakouluissa kursseina, joilla opiskelijoiden tehtävät tarkastetaan ja hyväksytään paikan päällä. Opetusvideoiden käyttö opetuksessa on kreikkalaisessa tutkimuksessa parantanut opetuksen vaikuttavuutta ja nostanut keskimääräisiä arvosanoja. Eräs läsnäolopakollisen opiskelun hyödyistä on se, että opiskelijat saavat käsityksen omasta tasostaan vertailemalla suorituksiaan muiden opiskelijoiden suorituksiin. [5]

Konetekniikan alan insinöörin tarvitsemiin taitoihin sisältyvät myös muut kuin varsinaisten mallinnustyökalujen käyttö. Nykyaikaiseen tuotesuunnitteluprosessiin liittyy CAD-ohjelmien lisäksi useita muitakin ohjelmallisia työkaluja. Kattotermillä CAx tarkoitetaan mitä tahansa tietokoneavusteista suunnitteluprosessin vaihetta. Varsinainen suunnittelu ja muotoilu tapahtuu CAD-ohjelmilla, simulointeja voidaan suorittaa erillisillä elementtimenetelmää (FEM) käyttävillä ohjelmilla. Valmistuksessa käytetään tietokoneavusteiseen valmistukseen (CAM) tarkoitettuja ohjelmia ja tuotetietoa hallitaan tuotetiedon ja elinkaaren hallintajärjestelmillä (PDM/PLM). Useissa kone-

tekniikan koulutusohjelmissa opetetaan CAD-opetuksen ohella myös muiden tuotteiden suunnitteluun tarkoitettujen ohjelmien käyttöä. [6] Suurissa projekteissa tarpeen tulee myös hallita usean käyttäjän yhtäaikaista osallistumista jonkin kokonaisuuden mallinnukseen.

Teollisuudessa CAD-mallinnusta ja työkalujen käyttöä opetetaan yleisesti myös valmistajien ja koulutusta järjestävien yritysten tarjoamilla kursseilla.

2.1.2 Konetekniikan CAD-opetus suomalaisissa korkeakouluissa

Korkeakoulujen CAD-opetuksessa suoritetaan tavanomaisesti alalla yleisesti käytössä olevalla ohjelmalla erilaisia mallinnus- ja harjoitustehtäviä. Useissa yliopistoissa tarjolla opetukseen sisältyy tuotetiedon hallintajärjestelmien opetusta, jotka mahdollistavat usean henkilön samanaikaisen työskentelyn CAD-kokoonpanojen kanssa. Yliopistoissa CAD-opetuksen tavoitteena on opettaa mallinnusprosessia ylipäätään ja yleispäteviä mallinnusmenetelmiä jonkin tietyn ohjelman toiminnan ja käyttöliittymän opettelun sijaan. [7] Opinto-oppaissa CAD-kurssien oppimistavoitteissa mainitaankin säännönmukaisesti parametriset mallinnusmenetelmät. CAD-mallinnusta opetetaan Suomessa myös ammattikorkeakouluissa.

Suomalaisten yliopistojen konetekniikan alan opinto-oppaisiin suoritetun katsauksen perusteella tyypilliseen konetekniikan diplomi-insinöörin tutkintoon kuuluu varsinaista tietokoneavusteisen mallinuksen perusteiden opetusta 5-10 opintopisteen verran. Koneensuunnittelun osaamistavoitteita ovat konejärjestelmien mallinnuksen ja simuloinnin lisäksi esimerkiksi suunnittelumetodiikan ja tuotekehityksen hallitseminen. [8]

Aalto-yliopisto

Aalto-yliopistossa kone- ja rakennustekniikan koulutusohjelman opiskelijat suorittavat kaikille pakollisen tietokoneavusteisen mallintamisen 5 opintopisteen peruskurssin *Tietokoneavusteiset työkalut insinööritieteissä*, jolla opetetaan CAD-mallinuksen perusteita eri ohjelmilla. Kurssi koostuu eri ohjelmilla toteutettavista moduuleista, joista opiskelijat voivat valita itseään kiinnostavat ohjelmat. Kurssilla käsitellään perustason mallinnustehtävien lisäksi teknisen (2D) piirustuksen perusteita. Konetekniikan alan CAD-mallinnukseen tarjolla kurssilla ovat Solid Edge- ja Creo-moduulit. Aalto-yliopiston peruskurssien CAD-opetus on kontaktiopetusta, jossa kurssiassistentit tarkastavat opiskelijoiden tehtävät ja merkitsevät ne suoritetuiksi. Kursseilla on myös sähköisiä palautusmahdollisuuksia, tarvittaessa peruskurssin tehtävät voi palauttaa myös sähköpostitse.

Ennen tutkintouudistusta, peruskurssin jälkeen CAD-opetusta on tarjolla myös parametriseen mallintamiseen ja mallinnustyökaluihin keskittyvällä Tietokoneavusteisen suunnittelun jatkokurssilla (6 op). Aalto-yliopistossa CAD-mallintaminen on myös osa useiden muiden kone- ja rakennustekniikan koulutusohjelman kurssien tehtäviä. [9]

Lappeenrannan teknillinen yliopisto

Lappeenrannan teknillisessä yliopistossa konetekniikan koulutusohjelmassa on tarjolla luennoista, harjoituksista ja projektityöstä koostuva kurssi *Tekninen dokumentointi ja 3D-mallinnus*. Kurssilla opetellaan mallintamaan erityyppisiä geometrioita SolidWorks-ohjelmalla, perehdytään teknisten dokumenttien laadintaa koskeviin standardeihin ja tuotetiedon hallinnan perusteisiin. Lappeenrannan teknillisessä yliopistossa tietokoneavusteisten työkalujen käyttöä edellyttäviä tehtäviä on myös muilla konetekniikan kursseilla. [10]

Tampereen teknillinen yliopisto

Tampereen teknillisessä yliopistossa konetekniikan opetuksen koneensuunnittelun pääaineessa opetusta tietokoneavusteisten työkalujen käyttöön tarjotaan 4 opintopisteen kurssilla *Computer Aided Engineering*, jolla opetetaan tuotetiedon hallintajärjestelmien (PDM) käyttöä ja CAD/CAE-ohjelmia, sekä mekanismien ja systeemien simulointia. Tampereella CAD-opetuksessa käytetään SolidWorks- ja NX-ohjelmia [11]. Tampereen teknillisessä yliopistossa ovat lisäksi tarjolla kurssit *Koneensuunnittelu ja CAD-mallinnuksen perusteet* (5 op) ja *Model-based Design of Machine Systems*. Koneenosien suunnittelun opetuksessa on tarjolla tietokoneavusteista lujuuslaskentaa käsittelevä kurssi. Lisäksi tarjolla on teknillisen dokumentoinnin kurssi. [8]

Oulun yliopisto

Oulun yliopiston konetekniikan koulutusohjelmassa tietokoneavusteista mallinnusta opetetaan kurssilla *Koneenpiirustus ja CAD*, jolla opetetaan myös perinteistä koneenpiirustusta ja konetekniikan keskeistä käsitteistöä. Maisterivaiheen opinnoissa tarjolla on *Tietokoneavusteisen suunnittelun opintojakso*, jonka lisäksi tietokoneavusteisia työkaluja käytetään muilla tutkintoon sisältyvillä kursseilla. Oulun yliopistossa käytetään Siemensin NX-ohjelmistoa ja Catiaa, tulevaisuudessa on tarkoitus opettaa myös tuotetiedon hallintaa NX Teamcenter -ohjelmistolla. [12] [13]

2.1.3 MOOC-kurssit

MOOC-kurssit eli massiiviset avoimet verkkokurssit ovat internetissä järjestettäviä kaikille avoimia verkkokursseja, joille ei tyypillisesti ole esitietovaatimuksia ja joista ei aina saa virallista suoritusmerkintää. MOOC-kurssien materiaalit ovat yleensä avoimesti tarjolla internetin kautta. Kurssit yleistyivät, kun vuonna 2011 Yhdysvaltalaiset yliopistot avasivat tietyillä kursseilla kurssi-ilmoittautumisen ilmaiseksi yleisölle. MOOC-kursseista on käyty viime vuosina laajaa kansalaiskeskustelua ja kursseilla nähdään olevan mahdollisuus rikkoa perinteisen yliopisto-opetuksen vakiintuneita menetelmiä. Yliopistoissa on aikaisemminkin ollut tarjolla verkkokursseja itsenäisellä suoritusaikataulumahdollisuudella, mutta yliopistokurssien avaaminen laajalle yleisölle on vasta 2010-luvulla yleistynyt ilmiö. [14]

Suomessa Helsingin yliopisto ja Aalto-yliopisto ovat tarjonneet avoimella MOOC-alustalla osallistumismahdollisuutta ohjelmoinnin ja matematiikan kursseille. Aalto-yliopisto tarjoaa tällä hetkellä viittä avointa MOOC-kurssia ohjelmoinnista ja matematiikasta. [1]

Muistakin verkossa toimivista automaattisista oppimisympäristöistä on julkaistu tutkimusta. Opiskelijoilta on saatu positiivista palautetta järjestelmien käytettävyydestä ja hyödyllisyydestä. Automaattiset ympäristöt nostavat myös huomattavasti palautettujen tehtävien määrää. [15]

[1] `http://mooc.aalto.fi/`

2.2 Kolmiulotteiset mallit

Kappaleen mallilla tarkoitetaan sellaista tietokokonaisuutta, jonka avulla voidaan vastata kappaletta koskeviin kysymyksiin. Kolmiulotteisiin malleihin voidaan tallentaa tietoa muistakin ominaisuuksista kuin kappaleen geometriasta, kuten ulkonäöstä, materiaalista tai lujuusominaisuuksista.

2.2.1 Kolmiulotteisen tiedon tallentaminen

Kolmiulotteisen geometrian tallentamiseen on olemassa useita erilaisia menetelmiä, joiden soveltuvuus riippuu siitä, mihin käyttötarkoitukseen kolmiulotteista tietoa halutaan välittää. Tietoa kolmiulotteisesta geometriasta on mahdollista tallentaa esimerkiksi pistepilvinä, monikulmioina, käyrinä ja viivoina (rautalankamalleina) sekä implisiittisesti tai parametrisesti määriteltyinä pintoina ja käyrinä [16]. Konetekniikassa valmistuksen kannalta olennaista on yleensä vain tarkka tieto geometriasta, mutta malleihin tallennetaan usein myös muuta tietoa. Renderöintejä varten tai tietokonegrafiikassa on tarpeen tallentaa myös kappaleen pintojen ulkonäköön liittyvää tietoa. Simuloinnissa malleihin liitetään myös tietoa materiaalin ominaisuuksista.

Tietokonegrafiikassa tietoa kolmiulotteisista kappaleista tallennetaan tavallisesti kolmioina. [17] Pisteiden välille määritellään monikulmioita, yleensä kolmioita, joilla kuvataan kappaleen pintaa. Konetekniikassa polygonimalleja käytetään 3D-tulostuksessa. Tietokonegrafiikassa käytetyt menetelmät eivät sovellu sileiden muotojen tallentamiseen, koska kolmioiden väleille muodostuu aina kulmia. [17] Esimerkiksi virtauksien hallinnan tai valumuottien toiminnan kannalta on tärkeää, että pinnat kaareutuvat oikealla tavalla.

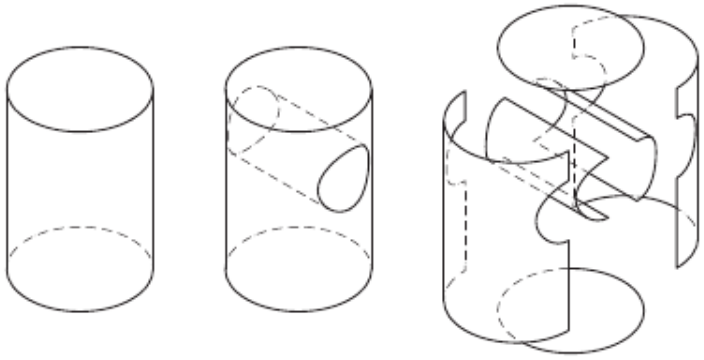
Parametrisiin CAD-malleihin tietoa geometrista ei tallenneta pistepilvinä tai monikulmioina vaan ne ovat “matemaattisesti määriteltyjä”, usein parametristen käyrien ja pintojen avulla. CAD-ohjelmien mallinnusytimet pystyvät mallin sisältämien tietojen avulla laskemaan, missä pinnat ja käyrät leikkaavat toisensa ja määrittämään, mitkä pisteet sijaitsevat mallin ulko- ja sisäpuolella. CAD-malleja on myös mahdollista muuttaa polygonimalleiksi. Muunnos tapahtuu luomalla CAD-mallin pinnoille muotoa mahdollisimman hyvin kuvaava pistejoukko ja muodostamalla kolmioita pisteiden välille.

2.2.2 Tilavuusmallit

Kolmiulotteisessa CAD-mallinnuksessa kappaleita voidaan kuvata tilavuusmalleina. Tilavuusmallit “ovat riittäviä vastaamaan algoritmisesti mihin tahansa geometriaa koskevaan kysymykseen“ ja ne sisältävät yksiselitteisen tiedon siitä, mitkä avaruuden pisteet kuuluvat kappaleen sisälle ja mitkä ulkopuolelle [18, s. 46].

Solidien eli tilavuuksien esitykseen on kehitetty erilaisia menetelmiä, joista nykyään yleisin esitystapa CAD-ohjelmissa on reunaesitys (B-rep). Reunamalliesitystä käyttävät esimerkiksi SolidWorks, Creo ja Solid Edge. Reunamalliesityksen lisäksi tilavuusmalleja on mahdollista määritellä konstruktiivisesti. CSG-malleissa mallit rakentuvat tilavuusprimitiiveistä, joita yhdistetään toisiinsa boolean-operaatiolla (lisäys tai poisto).

Solidien reunamalliesityksessä (kuva 1) matemaattisesti määritellyt funktiot määrittävät tilavuutta rajaavat pinnat tai käyrät. [19] CAD-ohjelmissa on myös tilavuusmallinnustyökaluja, jotka pitävät käyttäjän puolesta huolen siitä, että pinnat muodostavat jokaisen työvaiheen jälkeen suljetun tilavuuden. Vaikka pintamallinnuksen logiikka piilotetaan käyttäjältä kaikki reunamalliesitystä käyttävät CAD-tilavuusmallit koostuvat pintalapuista.



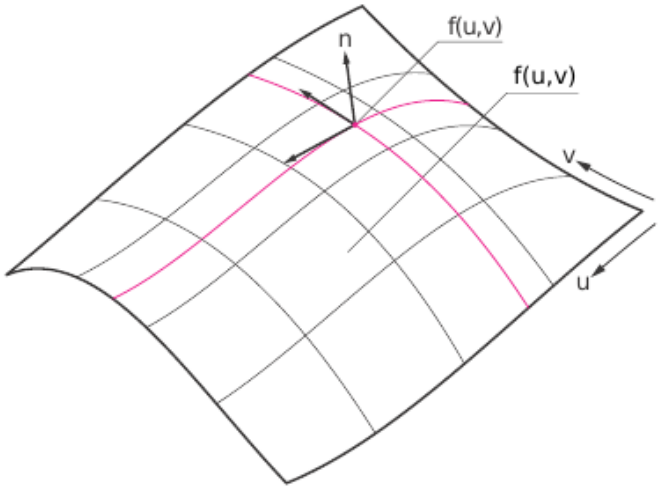
Kuva 1: Solidin reunamalliesitys. Tilavuuden rajaavat sitä ympäröivät parametrisesti määritellyt pinnat. (Kuva: Janne Ojala)

Kaarevien muotojen tallentamiseen CAD-ohjelmissa käytetään parametrisesti määriteltyjä NURBS-käyriä, joiden avulla on mahdollista määritellä parametrisesti myös kaarevia pintoja (kuva 2). NURBS-käyrän muoto määritellään kontrollipisteiden avulla, jotka määrittävät intuitiivisella tavalla käyrän muodon. Suunnittelija voi pisteiden avulla määrittää helposti pisteitä seuraavia kaarevia muotoja.

Valmistajien omat tiedostomuodot ovat usein binäärimuotoisia. Niihin tallentuu myös runsaasti erilaista metatietoa malleista (esimerkiksi mallinnushistoria peruutusominaisuuden käyttämiseksi). Useiden valmistajien CAD-ohjelmissa mallitiedostoihin on mahdollista määritellä esimerkiksi kappaleen lujuusominaisuuksiin tai ulkonäköön liittyviä parametreja.

2.2.3 Neutraalit tiedostoformaatit

Neutraalit tiedostoformaatit ovat kansainvälisissä standardeissa määriteltyjä tiedostoformaatteja, joilla voidaan siirtää tietoa eri valmistajien CAD- ja CAM-



Kuva 2: Parametrisen funktion $f(u, v)$ määrittämä parametrinen pinta. Pintaan on merkitty nuolilla gradienttivektorit ja niiden ristitulona saatava pinnan normaali **n**. (Kuva: Janne Ojala)

ja PDM-järjestelmien välillä. Yleisesti käytössä olevat neutraalit tiedostoformaatit ovat 1990-luvulla ensimmäisen kerran julkaistu STEP sekä vanhempi IGES. [19]

Standardit ovat kattavasti määriteltyjä ja tukevat monia tuotetiedon siirtoon liittyviä ominaisuuksia, joita kaikkiin CAD-ohjelmiin ei ole toteutettu. Neutraaliin tiedostoformaattiin CAD-ohjelmasta tallennettaessa tiedot piirrepuusta ja mallinnushistoriasta menetetään ja osatiedostoihin tallentuu vain tietoa kappaleen geometriasta. STEP-standardi tukee myös toleransseja, mitoitusta sekä useasta osatiedostosta ja niiden välille määritellyistä reunaehdoista koostuvia kokoonpanomalleja [20].

2.2.4 Parametrinen mallinnus

CAD-mallin mallinnustyövaiheet muodostavat parametrisissa mallinnusohjelmissa mallipuuhun loogisessa järjestyksessä piirteitä, jotka voivat sisältää viittauksia aiemmin tehtyihin piirteisiin. Parametrisessa mallinnuksessa on olennaista, että mallin hallitsevat piirteet parametrisoidaan oikealla tavalla. Usein käytettävässä *skeleton*-tekniikassa nämä pääpiirteet määritellään käyrien tai muiden piirteiden avulla mallin ylimmälle tasolle. Näin yhtä ylätason parametria muuttamalla voidaan hallita koko mallia.

Parametrisen mallin rakentamisessa on olennaista se, mitkä piirteet viittaavat

toisiinsa ja millä tavalla. Jos malli on oikein rakennettu, mallin muokkaaminen jälkikäteen muuttuu mahdolliseksi ja helpottuu. Esimerkiksi reikien seinämän paksuus voidaan kytkeä seuraamaan reiän halkaisijaa, jolloin yhtä parametria muuttamalla muut mallipuussa olevat piirteet mukautuvat tehtyihin muutoksiin.

2.2.5 Muodon samankaltaisuuden vertailu

Kolmiulotteisten mallien ja muodon vertailua tarvitaan useilla tekniikan aloilla, esimerkiksi kartoituksessa, konetekniikassa, virtuaalitodellisuuden tutkimuksessa, tekijänoikeuksien valvonnassa ja lääketieteessä. Kun 3D-mallien samankaltaisuutta vertaillaan, on ensin määriteltävä mitä samankaltaisuudella tarkoitetaan. Ihmisen kokema samankaltaisuus ei välttämättä tarkoita matemaattisesti samankaltaista geometriaa.

Yksinkertainen tapa yhdistää samankaltaisia malleja toisiinsa on liittää niihin ihmisen kirjoittama kuvaus ja hakea sen perusteella tietokannasta samankaltaisia malleja. Menetelmä ei kuitenkaan ole luotettava, sillä tutkimuksen perusteella on todettu, että ihmisten kirjoittamat kuvaukset muodoista ovat kulttuurisidonnaisia. [21] [22] Koska ihmisen käsittämänä muoto on kulttuuri- ja tilannesidonnaista, sen luokittelemisessa on pohjimmiltaan kyse siitä, miten määritellään sellaisia matemaattisia työkaluja, jotka kuvaavat matemaattista muotoa tavalla, joka liittyy siihen, miten ihminen muodon käsittää.

Kolmiulotteisen muodon samankaltaisuuden vertailuun on viimeisen vuosisadan aikana kehitetty useita algoritmeja ja matemaattisia menetelmiä. Mallien samankaltaisuutta mittaavia algoritmeja käytetään esimerkiksi 3D-skannauksessa, lääketieteessä, CAD-ohjelmistoissa sekä kolmiulotteisten mallien hakutietokannoissa. Menetelmissä mallit muutetaan muotoa kuvaavan funktion (*shape descriptor*) avulla sellaiseen muotoon, joka on riippuvainen ainoastaan kappaleen muodosta. Tavoitteena on tuottaa kappaleesta sen muotoa kuvaava yksinkertaistettu esitys (“sormenjälki”), joka esitetään numeroita sisältävänä vektorina. [23]

Osa muotoa kuvaavista funktioista on määritelty niin, että ne saavat mallien rotaatiosta ja translaatiosta huolimatta saman arvon. Osa menetelmistä kuvaa samalla tavalla kappaleita, jotka ovat toistensa peilikuvia, deformoituneita tai toistensa isometrisiä muotoja sisältäviä. [23] Laskentatehon lisääntyessä entistä useammat aikaisemmin laskennallisesti liian kalliit algoritmiset vertailumenetelmät ovat tulleet mahdollisiksi käyttää. [24]

Tietokantojen hauissa käytettävissä algoritmeissa korostuu vaatimus löytää samankaltainen malli mahdollisimman nopeasti, tarkkuudenkin kustannuksella. [24] CAD-harjoitustehtävien tarkastuksessa tarkkuuden on oltava riittävä. Nopeus ei ole yhtä merkitsevää, koska malleja vertaillaan vain kahta kerrallaan.

Tietoa siitä, miten suljetun lähdekoodin kaupallisten CAD-ohjelmien vertailutyökalut tarkalleen toimivat ei ole julkisesti saatavilla, koska tiedot ohjelmien työkalujen tarkemmasta toiminnasta ovat ohjelmien valmistajien liikesalaisuuksia. SolidWorksin dokumentaatiosta löytyy maininta siitä, että ohjelma vertailee kaarevia muotoja näytteistämällä. Lisäksi on kehitetty menetelmiä, joissa reunamalleja konstruoidaan pistepilvistä – näihin sisältyy pistepilvien sovittaminen toistensa päälle [25]. Todennäköisesti CAD-ohjelmien työkalut eivät käytä luvussa käsiteltyjä muodon vertailualgoritmeja. Ne sovittavat mallit päällekkäin ja suorittavat sitten vertailua vähentämällä tilavuudet toisistaan.

2.2.6 Kolmiulotteisen muodon vertailumenetelmiä

Muotojen luokitteluun on viimeisen neljänkymmenen vuoden aikana kehitetty useita erilaisia tilastollisia menetelmiä. Geometrian luokittelumenetelmät voidaan yleisesti jakaa muotoon ja topologiaan perustuviin menetelmiin. Molemmissa vertaillaan toisiinsa kappaleiden geometrisia jakaumia – muotoon perustuvissa reunojen tai monikulmioiden jakaumaa, topologiaan perustuvissa erilaisten topologisten funktioiden arvoja. Suuri osa alan tutkimuksesta onkin keskittynyt sellaisten mittareiden luomiseen, jotka tekevät muodon ja piirteiden kvantifioimisen mahdolliseksi. [24] Koska muodon käsitteen määrittely yksiselitteisesti on hankalaa, on muodon luokitteluun käytettävä menetelmä valittava käyttötarkoitusta silmällä pitäen.

Spektraaliset menetelmät

Spektraalisilla menetelmillä tarkoitetaan Laplace-Beltramin operaattorin ominaisvektorien ja ominaisarvojen laskentaan perustuvia menetelmiä. Laplace-Beltramin operaattori on määritelty funktion gradientin divergenssinä.

Kappaleen muodon spektrillä tarkoitetaan sen Laplace-Beltrami-operaattorista laskettuja ominaisarvoja, joiden avulla on mahdollista määrittää kappaleen geometriaa kuvaava “sormenjälki”. Useat spektraaliset menetelmät ovat invariantteja erilaisten geometristen muunnosten kuten rotaation, translaation tai skaalauksen suhteen. Muotojen matemaattiseen luokitteluun käytetään myös Reeb-graafeja ja harmonisia pallofunktioita (engl. spherical harmonics). [23]

Mediaaliakselit

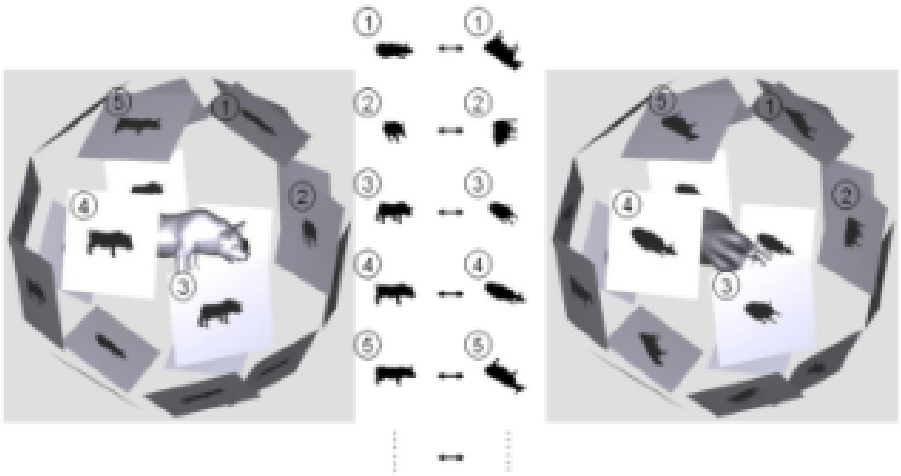
Ennen kuin muodon luokitteluun oli kehitetty menetelmiä, oli jo määritelty työkaluja muiden kappaleen ominaisuuksien kuin muodon (väri, reunat, kulmat, valaistus) kvantifioimiseen. Uusina menetelminä julkaistiin tutkimusta

muotojen luokittelemisesta mediaaliakselien avulla. Kappaleen mediaaliakseli on se pistejoukko, jolla on useampi kuin yksi lähin piste jonkin kappaleen ulkoreunaan. Käytännössä kappaletta ohennetaan viivoiksi ja pinnoiksi. Mediaaliakseli on muodon mittarina intuitiivinen ja muotojen tunnistaminen ihmiselle saattaa olla mahdollista pelkästään mediaaliakselin perusteella. Mediaaliakselien avulla on mahdollista luokitella tilastollisesti sekä kaksi- että kolmiulotteisia kappaleita. [21]

Pistepilvien sovittaminen

Kolmiulotteisten mallien sovittamista päällekkäin kutsutaan *rekisteröinniksi* (engl. registration) tai *lokalisaatioksi* (engl. localization) [23].

ICP-algoritmia (Iterative Closest Point) voidaan käyttää pistepilvien sovittamiseen toistensa päälle, myös CAD-malleista on mahdollista muodostaa pistepilviä. Nopeasti konvergoituva ICP-algoritmi määrittää translaation ja rotaation, joka pistejoukkoon pitäisi kohdistaa, jotta kahden pistejoukon väliset erot minimoituisivat. Pistejoukkojen sovitusten paremmuutta algoritmissa mitataan pienimmän neliösumman menetelmällä. [16]



Kuva 3: Kahden 3D-mallin vertailu dodekaedrin tahkoilta otetuista projektioista. Projektiot muodostavat *Light Field* -kuvauksen mallista. Kolmiulotteisen muodon vertailu suoritetaan vertailemalla sarjaa kaksiulotteisia kuvia. Muodostettujen kuvien määrä vaikuttaa vertailun tarkkuuteen. [24]

Visuaaliset menetelmät

Osa vertailualgoritmeista toimii visuaalisesti. Algoritmien periaatteena on se, että kolmiulotteiset mallit ovat toisiinsa nähden samankaltaisia mikäli ne näyttävät samankaltaisilta kaikista katselukulmista. Mallista tuotetaan

kaksiulotteinen kuva monesta eri katselukulmasta, jonka jälkeen kuvia vertaillaan referenssimallista otettuihin kuviin. Kaksiulotteisten kuvien vertailu ohjelmallisesti on mahdollista nopeasti ja siihen on olemassa useita tarkkoja menetelmiä. [22]

Light Field -menetelmää (kuva 3) voidaan käyttää kolmiulotteisten mallien hakutietokannoissa, joissa pyritään löytämään vertailumallilta näyttävä malli kannasta mahdollisimman nopeasti. Tietokonegrafiikassa kappaleiden sisäisillä muodoilla ei yleensä ole mitään merkitystä ja mallissa on tietoa ainoastaan kuoresta ja näkyvistä osista. Konetekniikan alan CAD-malleissa myös kappaleiden sisäiset muodot ovat merkityksellisiä, joten menetelmä soveltuu vain puutteellisesti CAD-mallien automaattiseen tarkastukseen.

Tilavuuden ja pinta-alan vertailu

Koska tilavuusmallit kuvaavat todellisia kappaleita, joilla on ulko- ja sisäpuolet, niille on aina mahdollista laskea pinta-ala ja tilavuus. Mikäli on kyse monimutkaisemman muotoisista kappaleista, on epätodennäköistä, että pinta-alaltaan sekä tilavuudeltaan identtiset kappaleet olisivat erilaisia. Tilavuutta ja pinta-alaa olisi mahdollista käyttää sellaisenaan vertailukriteereinä.

2.3 Tehtävien automaattinen arviointi ja tarkastus

Tehtävien automaattisen tarkastuksen mahdollisuuksia on tutkittu useilla eri aloilla. Suurin osa tutkimuksesta on tehty ohjelmoinnin opetuksen osalta, mutta tietokoneavusteista arvostelua on käytetty myös esimerkiksi kirjoitustehtävien, 2D-piirustustehtävien, kuvankäsittelytehtävien ja ääntämisen tarkastukseen. Useat automaattiset tarkastus- ja arvostelujärjestelmät käyttävät tietokonegrafiikkaa tai muuta multimediaa tiedonvälityksen parantamiseen. [26]

Myös CAD-mallinnuksen ja teknisen piirustuksen tehtävien automaattisen tarkastuksen mahdollisuuksia on tutkittu. CAD- ja piirustusohjelmat tuottavat ja tallentavat tietoa käyttäjän ohjelmassa suorittamista toimenpiteistä, jota olisi mahdollista käyttää tehtävien automaattisessa tarkastuksessa. [26]

Opiskelijoiden työpanosta kurssilla ohjaavat usein sellaiset osasuoritukset, joilla on selkeä vaikutus kurssin arvosanaan. Kaikissa arvostelujärjestelmissä on riski, että opiskelijat keskittyvät optimoimaan tehtävän suorittamista pisteitä maksimoivalla, mutta oppimisen kannalta haitallisella tavalla. Toisin kuin ihmistarkastajat, automaattiset tarkastusjärjestelmät eivät arvioi kokonaisuutta. [26] Järjestelmät voivatkin antaa opetushenkilökunnalle tietoa siitä, onko jokin tehtävä tai aihe opiskelijoille vaikea tai nostaa esiin yksittäisiä opiskelijoita, jotka eivät saa tehtäviä suoritettua odotetusti [27].

Työkalut automaattiseen tehtävien tarkastukseen mahdollistavat sen, että kurssien osallistujamäärää ei tarvitse rajata mitenkään. Automaattiset tarkastusmenetelmät siirtävät myös työtä pois opetushenkilökunnalta. Automaattisten tarkastusohjelmien lisäksi arvostelutyön siirtäminen pois kurssin opetushenkilökunnalta on mahdollista esimerkiksi opiskelijoiden toisilleen tekemänä vertaisarviointina.

Eräs tapa toteuttaa kurssihenkilökunnasta riippumaton tehtävien tarkastus on automatisoitu tehtävien vertaisarviointi. Perimmäinen syy automaattisten järjestelmien käyttöön on lopulta vähentää kurssihenkilökunnan työtä ja parantaa oppimista. Automaattisesti tehtäviä arvosteleva ohjelma on usein vain yksi osa laajempaa järjestelmäkokonaisuutta, josta löytyy erilaisia opettajien ja opiskelijoiden työtä helpottavia ominaisuuksia. Useista järjestelmistä löytyy esimerkiksi ominaisuuksia vertaisarviointiin tai mahdollisuus yhdistää käsintarkastus ja automaattinen tarkastus. Opettajien saattaa myös olla mahdollista antaa palautetta jo automaattisesti arvostelluista tehtävistä.

Järjestelmissä on usein tarjolla myös tehtävänannot palautettaviin tehtäviin. Tehtävänantoja voi olla mahdollista parametrisoida, jolloin eri opiskelijat saavat tehtäväkseen erilaiset tehtävät. Usein oppimisympäristössä rajoitetaan palautusten ja tarkastusten määrää ja palautuksille on myös määritelty tietty aikaikkuna, jonka jälkeen palauttaminen ei ole mahdollista.

Oppimisympäristöjen ja tarkastusjärjestelmien vaatimukset muuttuvat nopeasti, koska sekä niiden toteuttamiseen käytetty tekniikka että niillä opetettavat tekniikat kehittyvät ja muuttuvat jatkuvasti [28]. Muutoksiin on mahdollista varautua suunnittelemalla järjestelmät modulaarisiksi ja helposti konfiguroitaviksi.

Eräs tapa toteuttaa kurssihenkilökunnasta riippumaton tehtävien tarkastus on automatisoitu tehtävien vertaisarviointi.

2.3.1 Ohjelmointitehtävien automaattinen tarkastus

Ohjelmien testaus on olennainen osa nykyaikaista ohjelmistokehitysprosessia ja testausta on tutkittu jo 80-luvulta saakka. Julkaistavan ohjelman jokaiselle ominaisuudelle saatetaan kirjoittaa ohjelmallinen testi. Kattavilla testeillä pyritään myös helpottamaan vaatimusten hallintaa.

Ohjelmoinnin opetuksessa onkin jo useiden vuosien ajan käytetty automaattisia tehtävien tarkastus- ja arviointijärjestelmiä. Osa järjestelmistä tarkastelee vain tarkastettavien ohjelmien tulosteita tietyillä syötteillä, mutta osassa järjestelmistä ajetaan myös kattavampia testejä yksittäisille funktioille tai tarkastetaan erilaisten apuohjelmien avulla, että ohjelma on toteutettu oikein. Osassa tarkastusohjelmista muodostetaan palautetun ohjelman logiikasta graafi, jonka avulla ohjelman toimintaa on mahdollista tarkastaa. [29]

Ohjelmointitehtävien automaattiset tarkastusohjelmat ajavat erilaisia määriteltyjä testejä joko opiskelijoiden koneilla tai erillisellä tarkastuspalvelimella. Kun vierasta lähdekoodia suoritetaan palvelimella, on olemassa riski, että ohjelmakoodi on haitallista. Tarkastettavia ohjelmointitehtäviä ajetaankin usein hiekkalaatikkoympäristössä, josta ei ole pääsyä muulle palvelimelle. [29]

Hiekkalaatikkoympäristö ei ole erityisen relevantti CAD-mallien tarkastuksessa, koska automaattisen tarkastusohjelman käyttö ei tietoturvanäkökulmasta juuri eroa muusta sellaisesta CAD-ohjelmien käytöstä, jossa avataan muualta tulleita malleja. On joka tapauksessa CAD-ohjelmistojen valmistajien vastuulla varmistaa, että ohjelmat toimivat tietoturvallisesti, eikä asiaan voi juuri vaikuttaa.

2.3.2 Kolmiulotteisten mallinnustehtävien automaattinen tarkastus

Luku on katsaus löydettyihin olemassa oleviin 3D-mallien automaattisiin tarkastus- ja arvostelujärjestelmiin. Kirjallisuuskatsauksen aikana löytyi erilaisia toteutettuja CAD- ja 3D-mallinnuksen tehtävien tarkastukseen tarkoitettuja järjestelmiä.

Kirjallisuustutkimuksessa löytyneet tarkastusjärjestelmiä kuvaavat julkaisut kuvaavat järjestelmiä ja niissä käytettyjä algoritmeja tai menetelmiä vain yleisellä tasolla. Hyvin yksityiskohtaisia tietoja teknisestä toteutuksesta, esimerkiksi käytettyä lähdekoodia ei katsauksessa löytynyt yhdestäkään julkaisusta. Usein julkaisuissa arvioidaan sitä, miten järjestelmällä voitaisiin helpottaa tarkastusta juuri kyseisessä oppilaitoksessa.

Kirjallisuuskatsauksen perusteella näyttää myös siltä, että suurin osa automaattisista tarkastusjärjestelmistä on jäänyt prototyyppiasteelle. Internetin kautta saatavilla olevia tällä hetkellä käytössä olevia CAD-mallien tarkastusjärjestelmiä ei kirjallisuustutkimuksen aikana löytynyt ainuttakaan.

Case: 3D-mallinnuskurssin kokeiden automaattinen tarkastus

Italialaisessa yliopistossa *First School of Architecture of the Politecnico di Torinossa* on julkaistu ehdotus automaattiseksi tarkastusmenetelmäksi graafisen ja virtuaalisen suunnittelun koulutusohjelmassa olevalle neljän opintopisteen 3D-mallinnuksen kurssille, jolla käytetään Blender-mallinnusohjelmaa. Kurssin suorittaa noin 120 opiskelijaa vuodessa ja sen tehtävien arvostelua on pidetty työläänä ja vaikeana. Järjestelmän tarkoituksena on arvostelun työmäärän vähentämisen lisäksi vähentää arvostelun subjektiivisuutta. Järjestelmän käyttöönoton jälkeen opiskelijoiden pyynnöt arvostelun perustelemiselle ovat vähentyneet. Automaattisesti ja käsin arvosteltujen tehtävien arvosanat ovat myös vastanneet toisiaan varsin hyvin. [26]

Ehdotuksen mukainen tarkastusjärjestelmä on tarkoitettu 3D-mallinnuskokeiden arvostelemiseen. Opiskelijoille annetaan tehtäväksi mallintaa ruutukaappausten perusteella tehtävänannon kanssa mahdollisimman yhtenevä malli, geometrian, polygonien määrän ja määriteltyjen materiaalien (tekstuurien) osalta. Järjestelmällä tarkastetaan ainoastaan lopputulosta, vaikka itse mallinnusprosessi on myös usein merkitsevä. Pelkkää lopputuloksen ulkonäköä tarkastettaessa on huomioitava mahdollisuus siihen, että pienen oppimiseen vaikuttamattoman virheen seurauksena on mahdollista päätyä täysin erinäköiseen lopputulokseen. 3D-mallinnuksessa samankaltaiseen lopputulokseen on usein myös mahdollista päästä useilla täysin erilaisilla menetelmillä, joiden soveltuvuus riippuu käyttökohteesta. [26]

Tarkastus perustuu palautetun mallin ja oikeaksi tiedetyn mallin vertailuun. Vertailun tekninen toteutus perustuu kolmeen mittariin: järjestelmä laskee tunnusluvut muotoverkon samankaltaisuudelle, materiaalien yhteneväisyydelle sekä polygonien määrälle. Muodon vertailu perustuu siihen, että malli on saman muotoinen, jos se näyttää samalta kaikista suunnista. Mallin siluetista renderöidään kuvia (projektioita) dodekaedrin tahkoilta, joita vertaillaan samalla menetelmällä oikeasta mallista tuotettuihin kuviin. Materiaalin vertailu on toteutettu ottamalla yksittäisten pikselien värinäytteitä

tietyistä mallin kohdista. Pisteytys määräytyy sen perusteella, kuinka etäällä värit ovat toisistaan väriavaruudessa. Mallin polygonien määrä on mittarina vain suuntaa-antava ja vertailussa oikean mallin polygonimäärään on suuri toleranssi. Ainoastaan 10-100 kertaisista ylittyvistä polygonimääristä vähennetään pisteitä. [26]

Konetekniikan alalla käytettävät parametriset CAD-ohjelmat eroavat 3D-mallinnusohjelmista siten, että 3D-malleissa geometria ei muodosta samanlaista viittauksia hallitsevaa piirrepuuta. CAD-kursseilla tekstuurien tai materiaalien määrittelyä ja mallien renderöintiä ei tyypillisesti opeteta peruskursseilla, mutta 3D-mallinnuksen opetuksessa ne ovat osa perusteita.

Eroista huolimatta konetekniikan alan CAD-tehtävien tarkastukseen voisi kuitenkin soveltua samankaltainen prosessi, jossa erilaisilla menetelmillä ja mittareilla analysoitaisiin sitä, onko malli rakennettu oikein. On huomattava, että suuri osa ehdotetun 3D-mallien tarkastusjärjestelmän tarkastuksista on varsin yksinkertaisia.

Case: SolidWorks-ohjelmalla tehtyjen tehtävien automaattinen tarkastus

Vuonna 2003 Rensselaerin polyteknillisessä instituutissa on julkaistu raportti SolidWorks-ohjelmalla toteutetun yhden opintopisteen CAD-peruskurssin arvostelun automatisoimiseksi. Tarkastusten lähtökohdaksi on otettu kurssille kehitetty luokittelu tehtävien arvosteluun. Arvostelussa huomioidaan erillisinä osa-alueina visualisaatio, tekniset relaatiot ja suunnittelutarkoitus (engl. design intent).

Raportissa on mainittu yleisellä tasolla, että automaattinen arvostelu toimii tekemällä eri kriteerejä vasten tarkastuksia SolidWorksin makrojen ja rajapintojen kautta. [27]

Vaikka raportti sisältää kuvauksen kurssin tarpeista, arvosteluprosessista ja sen tiedonkulusta ylipäätään, yksityiskohtaista tietoa automaattinen arvostelun teknisestä toteutuksesta ei ole. Toteutusta on muutenkin pidettävä jo vanhanaikaisena, nykyään tarjolla on valmiita työkaluja useiden järjestelmän osa-alueiden toteuttamiseen.

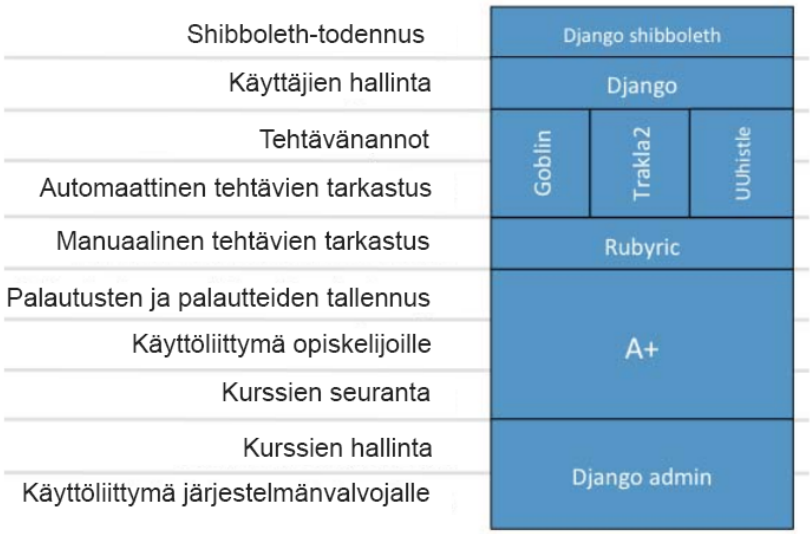
2.3.3 Automaattiset tarkastusjärjestelmät Aalto-yliopistossa

Ohjelmoinnin, tietotekniikan, matematiikan ja fysiikan opetuksessa Aalto-yliopistossa on käytössä useita automaattisia tehtävien tarkastusohjelmia. Matematiikan opetuksessa on laajalti käytössä STACK-järjestelmä, jota käyteään matematiikan ja fysiikan tehtävien automaattiseen tarkastamiseen. Sähkötekniikan korkeakoulun C-ohjelmoinnin kurssilla käytetään Helsingin yliopistossa kehitettyä vuonna 2011 Aalto-yliopistossa käyttöön otettua Test

My Code -järjestelmää. Tietotekniikan laitoksella noin kymmenellä kurssilla on käytössä Aalto-yliopistossa kehitetty A-plus-järjestelmä, jossa tarkistimet ovat ulkoisia tilattomia (erilliset palautukset eivät riipu toisistaan ja samanlaisella syötteellä saadaan aina sama tulos) palveluita, joihin tehtävät lähetetään tarkastettaviksi. Aikaisemmin käytössä ovat olleet Webcat- ja Ceilidh-nimiset automaattiset tarkastusjärjestelmät. [30]

A+-järjestelmä

Aalto-yliopistossa käytössä oleva A+-järjestelmä on useasta eri osajärjestelmästä koostuva kokonaisuus (kuva 4), jossa käyttöliittymä, tiedon tallennus ja varsinaiset tarkastukset on erotettu toisistaan.



Kuva 4: A+-järjestelmän ominaisuudet ja niistä vastaavat ohjelmiston osat. [28]

A+:n käyttämät automaattiset tarkastusohjelmat on toteutettu ulkoisina palveluina, jotka ottavat tarkastuksessa tarvittavat tiedot sisään HTTP-kutsuina. Tarkastuskutsuissa mukana tulee tunniste, jolla tarkastuksen tulokset voidaan palauttaa ja yhdistää palautettuihin tehtäviin. A+:n tarkastusohjelmat ovat tilattomia, eli ne eivät tallenna mitään tietoa tai ole muutenkaan riippuvaisia toisistaan. Tilattomat tarkastukset mahdollistavat myös saman tehtävän samanaikaisen käyttämisen usealla eri kurssilla.

Arkkitehtuuri, jossa järjestelmä koostuu erillisistä osajärjestelmistä ja niiden välille määritellyistä rajapinnoista mahdollistaisi myös esimerkiksi sen, että eri oppilaitokset voisivat hyödyntää toistensa järjestelmiä. [28]

Test My Code

Helsingin yliopistossa kehitetty *Test My Code* -järjestelmä on tarkoitettu ohjelmoinnin opetuksessa toistuvien työtehtävien vähentämiseen. Osana järjestelmää on automaattisen tarkastusohjelman lisäksi tehtävien hallintapalvelu, jolla opiskelijoiden ohjelmointitehtäviä voidaan hallita. Järjestelmää käytetään myös palautteen keräämiseen ja välittämisen molempiin suuntiin.

Test My Coden tarkastukset ja tehtävät pyritään määrittelemään osittain siten, että ne tukevat ohjelmoinnin oppimisprosessia. Kurssien taustalla oleva XA-metodi (Extreme Apprenticeship) korostaa jatkuvaa tiedonvaihtoa opiskelijoiden ja opettajien välillä.

Test My Codea on mahdollista käyttää web-käyttöliittymän tai integroituun ohjelmistoympäristöön (Netbeans) toteutetun lisäosan kautta. Järjestelmä on myös integroitu Git-versionhallintajärjestelmään, johon kurssien tehtävät tallennetaan. Tarkastuksia on mahdollista suorittaa sekä opiskelijoiden omilla koneilla että järjestelmän palvelimella. [31]

3 Automaattisen mallinnustehtävien tarkastusjärjestelmän prototyyppi

3.1 Tarkastusprosessin määrittely

Mallinnustehtävien tarkastuksen ja arvostelun lähtökohtana on lopulta tarkastaa, miten hyvin palautettu tehtävä täyttää jotkin ennalta asetetut kriteerit. Automaattinen tarkastus CAD-mallinnustehtäville voisi toimia seuraavasti: opiskelijat palauttavat tehtävänantojen perusteella laaditut mallit järjestelmään, järjestelmä tarkastaa ne (vertailemalla joko tallennettuun tietoon tai olemassa olevaan oikeaan malliin), antaa opiskelijoille palautteen tarkastuksen tuloksista ja tallentaa sen (kuva 5).

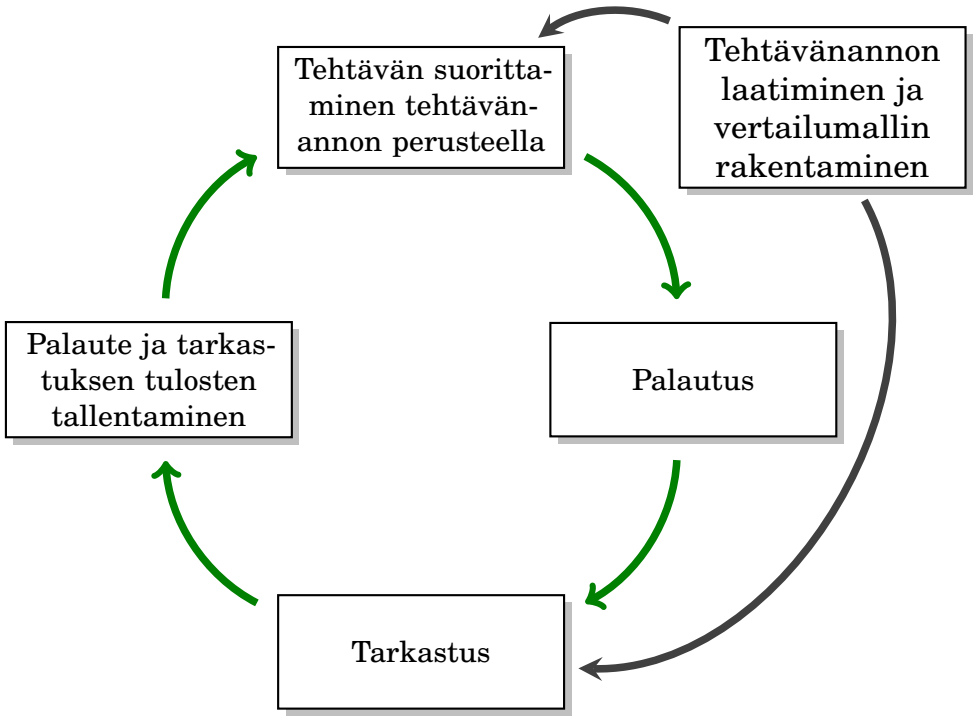
Lähtökohtana automaattiselle mallinnustehtävien automaattiselle tarkastukselle on se, että palautetusta mallinnustehtävästä luetaan ohjelmallisilla työkaluilla tietoa, jonka avulla voidaan varmistua siitä, että malli on oikein rakennettu. Tietojen lähde voi olla vastaava oikein rakennettu CAD-malli, johon tarkastusjärjestelmä voi vertailla tarkastettavaa mallia samoilla menetelmillä kuin millä tietoa tarkastettavista malleista luetaan. Vaihtoehtoisesti palautetusta mallista luettuja tietoja voidaan vertailla esimerkiksi valmiiksi määriteltyihin arvoihin tai voidaan erilaisten tarkastusfunktioiden avulla määrittää toteuttavatko ne muita mallille asetettuja vaatimuksia.

3.1.1 Järjestelmän vaatimusten määrittely

Järjestelmässä käyttäjät lataavat web-käyttöliittymän kautta tarkastettavaksi ohjeiden mukaan mallinnettuja CAD-malleja. Järjestelmään on määritelty tieto oikeasta mallista, josta se automaattisesti lukee tarkastuksessa tarvittavat tiedot. Kullekin käyttäjälle näytetään kaikki saman käyttäjän aikaisemmin palveluun lataamat mallit ja niiden tarkastusten tila.

Mallien vertailu toteutetaan käyttämällä rajapintojen tai trail-tiedostojen avulla Creosta löytyviä työkaluja, tai muita ohjelmallisia työkaluja, oikean mallin ja palautettavan mallin vertailuun. Tarkastusjärjestelmään on määriteltynä erilaisia tarkastusfunktioita, jotka voidaan tehtäväkohtaisesti tarpeen mukaan kytkeä päälle tai pois ja joiden parametreja voidaan tehtäväkohtaisesti muuttaa. Käyttöliittymän kautta on mahdollista hallita tehtäväkohtaisesti tarkastuksia ja nähdä niiden tulokset. Tarkastuksia voivat olla esimerkiksi:

- Täsmäävä tilavuus eri parametreilla
- Täsmäävä pinta-ala eri parametreilla
- Tietyn piirteen olemassaolo mallipuussa



Kuva 5: CAD-mallien tarkastusprosessi.

- Mallipuun piirteiden järjestys
- Malli on symmetrinen
- Piirteiden keskinäinen järjestys mallipuussa
- Malli regeneroituu erilaisilla parametreilla
- Geometrian vertailu compare geometry -työkalulla
- Malli läpäisee ModelCheck-tarkastukset
- Täsmälleen samaa tiedostoa ei ole aikaisemmin palautettu järjestelmään toisen käyttäjän toimesta
- Mallia samoilla metatiedoilla ei ole palautettu aikaisemmin järjestelmään toisen käyttäjän toimesta

Järjestelmän tulee antaa ja tallentaa malleista arvosana. Järjestelmä antaa myös tarkempaa palautetta ja palauttaa tiedon siitä, mitkä määritellyt tarkastukset malli läpäisee kussakin tarkastuksen vaiheessa. Arvosana voi määräytyä esimerkiksi sen perusteella mitkä tarkastukset läpäistiin. Kun

tietyt kriteerit täyttyvät, järjestelmä merkitsee tehtävän hyväksytysti suoritetuksi ja tallentaa tiedon tietokantaan. Vertailluista malleista voidaan myös tuottaa ja palauttaa kuva, jossa näytetään mallien eroavat kohdat (esimerkki compare geometry -vertailusta kuvassa 8 sivulla 34).

3.2 Vaihtoehtojen kartoitus tarkastuksen toteuttamiseksi

Luvussa tutkitaan, mitkä mallien ominaisuudet ovat olennaisia automaattisen tehtävien tarkastuksen kannalta ja mitä valmiita työkaluja ohjelmista näiden ominaisuuksien vertailuun löytyy. Työssä käsiteltäviksi ohjelmistoiksi valittiin Aalto-yliopiston opetuksessa käytettävät Solid Edge ja Creo Parametric, sekä koneenosien suunnittelun alalla suurimman markkinaosuuden omaava SolidWorks. Aalto-yliopistossa opetuksessa on käytössä myös Siemensin NX.

Tarkastettavaa tietoa malleista on mahdollista lukea CAD-ohjelmien valmiilla työkaluilla (rajapinnat, valmiit mallien tarkastustyökalut ja mallien vertailutyökalut). Mikäli mallit palautettaisiin neutraaleihin tiedostoformaatteihin muutettuna, voisi niistä suljetun lähdekoodin CAD-ohjelmien lisäksi lukea tietoa avoimilla ohjelmallisilla työkaluilla.

Tietoja, joita CAD-malleista on saatavilla ja joita tehtävien tarkastukseen voisi käyttää, ovat esimerkiksi mallin pinta-ala, tilavuus, piirrepuun piirteet ja itse geometria, jonka vertailuun CAD-ohjelmista löytyy valmiita työkaluja. Lisäksi mallit sisältävät metatietoja, esimerkiksi mallin tehneen käyttäjän käyttäjätunnuksen ja päivämäärän.

Teollisuudessa CAD-ohjelmien toimintaa automatisoidaan usein ohjelmista löytyvien rajapintojen kautta, jotka sisältävät tyypillisesti monipuolisia ominaisuuksia. Rajapintojen avulla CAD-malleista olisikin helposti saatavilla tarkastuksissa käyttökelpoista tietoa. Mallien tarkastus CAD-ohjelmien rajapintojen avulla voidaan jakaa geometrian tarkastukseen ja mallipuun tarkastukseen. Mallipuu voi alkeisharjoitustehtäviä lukuun ottamatta olla geometrialtaan samanlaisessa mallissa täysin erilainen – CAD-mallit voivat olla monimutkaisia ja samanlainen malli on mahdollista rakentaa useilla eri työkaluilla eri järjestyksessä. Mallipuun sisältämää tietoa saattaa silti olla mahdollista käyttää automaattisessa tehtävien tarkastuksessa. Geometrian tarkastuksessa tavoitteena on lukea mallista sellaista tietoa, jonka avulla voidaan varmistua siitä, että se on oikein tehty. Valmiita osien vertailutyökaluja voisi myös olla mahdollista ajaa rajapintojen kautta ja vertailla tarkastettavan mallin geometriaa referenssimalliin.

CAD-ohjelmista löytyy myös työkaluja, joilla kaksi samankaltaista mallia voidaan sovittaa päällekkäin ja vertailla niitä. Geometrian vertailuun voidaan käyttää myös ohjelmien kautta saatavilla olevia tilavuus- ja pinta-alatietoja.

Myös mallipuuta vertailevia työkaluja löytyy CAD-ohjelmista. Ne kuitenkin vaativat valmistajakohtaisen tiedostomuodon käyttämistä.

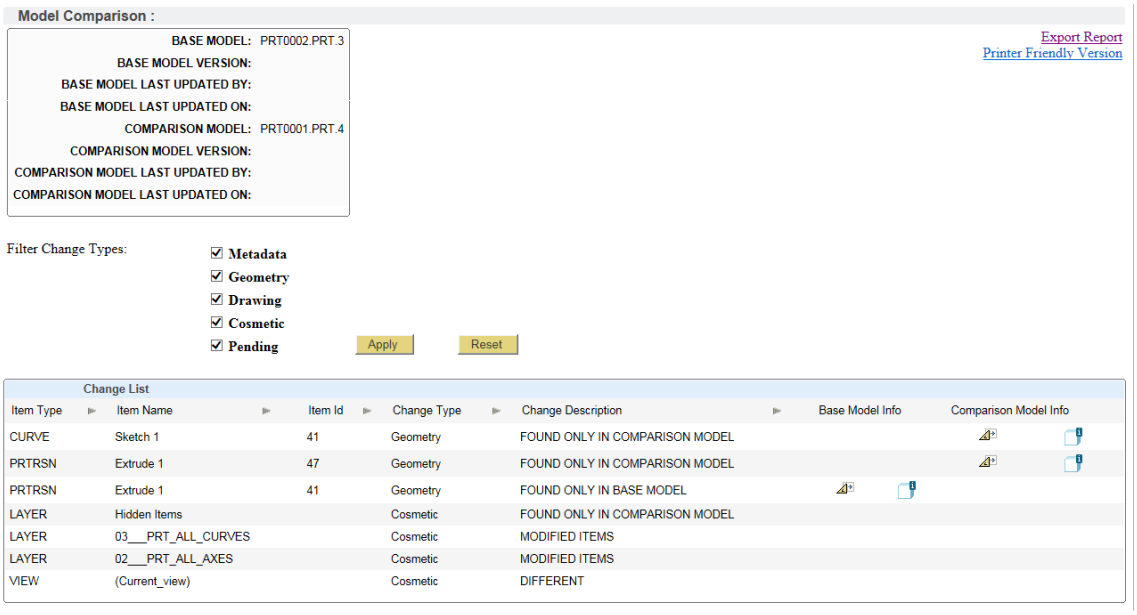
CAD-ohjelmista löytyy valmiita mallien tarkastukseen tarkoitettuja työkaluja. Teollisuudessa työkaluilla tarkastetaan enemmän kuin “oikein” suoritettua mallinnusta sitä, vastaako malli organisaatioon määriteltyjä vaatimuksia, joita voivat olla esimerkiksi osien mallintaminen referenssitasoihin nähden tietyllä tavalla. Tarkastustyökaluilla voi myös suorittaa tarkastuksia mallinnettujen osien valmistettavuuteen tai yhteensopivuuteen organisaatioissa määriteltyihin mallinnusvaatimuksiin liittyen. CAD-ohjelmista valmiiksi löytyvillä tarkastustyökaluilla on myös mahdollista tarkastaa esimerkiksi mallien eheys tiedostoformaatin muutoksen jälkeen ja valmistettavuuteen liittyviä kysymyksiä (esimerkiksi 3D-tulostuksessa muotojen soveltuvuus tulostukseen tai jyrsinnässä nurkkapyöristysten säteet).

Opiskelijat mallintavat tarkastettavat tehtävät tehtävänantojen perusteella. Tehtävänantojen yksityiskohtainen, yksiselitteinen ja tarkastusmenetelmän rajoitteet huomioiva laatiminen on tärkeää. On myös määriteltävä, mikä lasketaan tehtävän hyväksytyksi ratkaisuksi. On tehtävä valinta, vaaditaanko täsmälleen samalla tavalla rakennettu malli vai määritelläänkö tarkastusjärjestelmään väljemmät kriteerit, jotka tehtävän tulee täyttää?

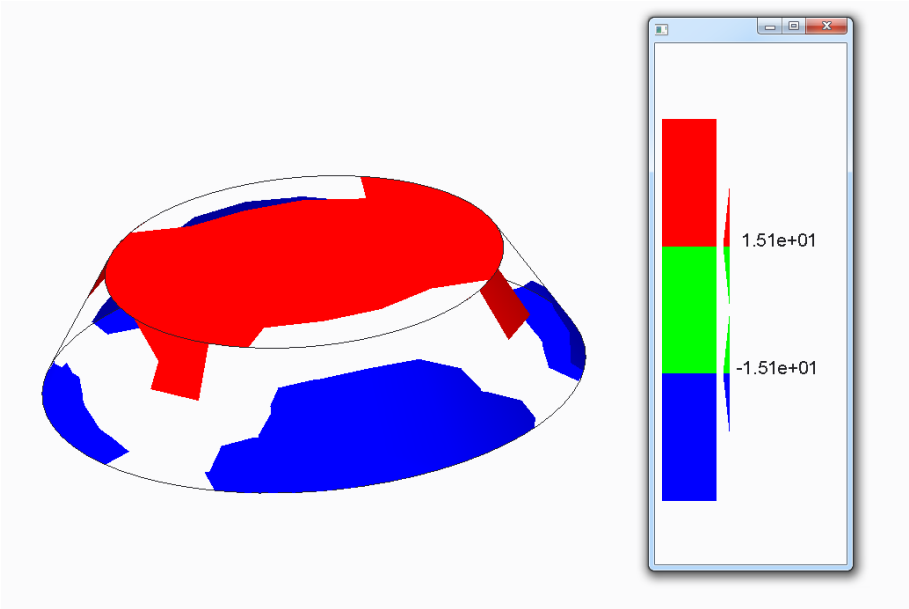
CAD-mallinnuksessa eri tavoilla mallinnetut kappaleet voivat olla geometrialtaan samankaltaisia ja samojen muotojen mallintaminen eri tavoilla voi olla perusteltua riippuen mallin käyttötarkoituksesta. Perustasonkin tehtäviä on mahdollista mallintaa esimerkiksi eri päin referenssitasoihin nähden. Opiskelijoilta olisi mahdollista vaatia täsmälleen samalla tavalla mallinnettua mallia. Täsmälleen tietyllä tavalla mallinnetun tehtävän vaatimista voisi perustella sillä, että teollisuudessa saattaa olla käytössä CAD-mallinnusohjeita tai standardeja, jotka määrittävät lähtökohdat sille miten osat tulee mallintaa.

Koska samanlaisen mallin voi parametrisissa CAD-ohjelmissa rakentaa eri tavoilla ja tilavuudeltaan hyvin lähellä oikeaa oleva malli voi olla perustavanlaatuisella tavalla virheellinen, oikeudenmukainen arvostelu ja pisteytys automaattisesti on hyvin hankalaa. Pieni virhe mallinnuksessa voi merkitä erittäin suurta virhettä geometriassa. Mikäli ohjelmallinen pisteytys on hankalaa, järjestelmä voisi palauttaa vain tiedon (hyväksytty/hylätty) siitä, läpäiseekö malli sille asetetut tarkastuskriteerit.

Tarkastusjärjestelmän olisi suotavaa antaa mallista muutakin palautetta opiskelijalle. Esimerkiksi geometrian vertailussa voitaisiin esittää kuvina rinnakkain tai päällekkäin palautettu malli ja oikea malli, ja näiden yhtenevät ja eriävät tilavuudet eri väreillä.



Kuva 6: Creon *Compare Model (features)* -työkalu vertailee mallien mallipuita.



Kuva 7: Creon *Model Compare (geometry)* -työkalulla on mahdollista tuottaa kuva vertailluista malleista. Kuvassa on vertailtu kahta toisistaan eroavaa kartiota.

3.2.1 CAD-ohjelmien tarkastukseen soveltuvat työkalut

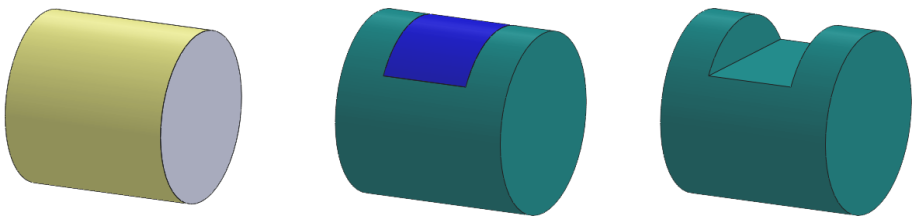
Luku on katsaus kolmesta CAD-ohjelmasta (Creo Parametric 3.0 M110, SolidWorks 2015 ja Solid Edge ST8) löytyviin tehtävien tarkastuksen kannalta olennaisiin työkaluihin. Ohjelmista löytyviä työkaluja on usein mahdollista käyttää mallien eheyden tarkastukseen sekä määriteltyjen mallinnustandar-

dien noudattamisen seurantaan. Mallien standardinmukaisuuden tarkastamista varten kehitettyjä ohjelmia on myös olemassa CAD-ohjelmien liitännäisohjelmina.

Solid Edge-ohjelmassa on *Geometry Inspector* -työkalu, jolla mallien geometrian eheyttä voi tarkastaa. Työkalulla on mahdollista tunnistaa pienikokoisia pintoja tai reunoja, joita voi syntyä kun CAD-malleja muunnetaan tiedostoformaatista toiseen. Työkalu ei tuota automaattisen mallinnustehtävien tarkastuksen kannalta olennaista tietoa.

SolidWorksin *Compare Geometry*- ja *Compare Features* työkaluilla on mahdollista vertailla kahden eri mallitiedoston piirrepuita ja geometriaa toisiinsa. Geometrian vertailua varten mallien on oltava samassa paikassa suhteessa origoon. SolidWorks tarjoaa tähän *Align parts* -työkalun, joka tulee ajaa ennen vertailun suorittamista. Geometrian vertailussa (kuva 8) mallit sijoitetaan päällekkäin. Työkalu vähentää mallit toisistaan, jonka jälkeen poikkeavan ja yhtenevän tilavuuden saa näkyviin.

SolidWorks vertailee tasoja ja muita analyyttisesti määriteltyjä pintoja matemaattisten yhtälöiden avulla, mutta splinikäyrien ja kaarevien pintojen vertailu tapahtuu näytteistämällä, josta saattaa aiheutua epätarkkuuksia. [32] CAD-opetuksen mallintarkastukseen tarkkuus on kuitenkin riittävä. Työkaluilla ei voi vertailla täysin erilaisia osia. [32] Esimerkiksi suurempaa kappaletta pienemmän kappaleen voisi sovittaa suuremman sisään miten päin tahansa, jolloin vertailua ei voi suorittaa.



Kuva 8: SolidWorks-ohjelman *Compare Documents* -työkalulla on mahdollista tuottaa kuvia vertailtavista malleista ja niiden eroavaisuuksista (keskellä eriävä tilavuus sinisellä).

Creo Parametric -ohjelmassa on useita automaattisen tehtävien tarkastuksen kannalta käyttökelpoisia valmiita työkaluja. Creossa on SolidWorksin tapaan osien vertailutyökaluja (kuvat 6 ja 7). Lisäksi Creosta löytyy *ModelCheck*-työkalupaketti, joka tarkastaa mallin yhteensopivuuden työkaluun määriteltyjen vaatimusten kanssa. ModelCheck-työkaluja käytetään teollisuudessa varmistamaan mallien yhteensopivuus yrityskohtaisten mallinnusstandardien ja valmistusmenetelmien kanssa. Tiettyjä ModelCheck-tarkastuksia olisi todennäköisesti mahdollista käyttää hyödyksi mallinnustehtävien tarkastuksessa. ModelCheck-tarkastuksien avulla voidaan varmistua siitä, että malli on tietyllä tavalla mallinnettu.

Creosta ja SolidWorksista molemmista löytyvät kahden osan vertailuun tarkoitetut työkalut, jotka toimivat sekä piirteiden että geometrian tasolla. Työkalut ottavat syötteenä kaksi mallitiedostoa ja tuottavat mallien eroavaisuuksista tietoa, jota olisi mahdollista käyttää tarkastuksessa. Molemmat ohjelmat tuottavat vertailusta myös kuvan ja tarkastusraportin, jotka voisivat olla suoraan käyttökelpoisia tehtävien tarkastuksessa. Creon ja SolidWorksin geometrian vertailutyökalut toimivat myös neutraaleilla tiedostoformaateilla.

3.2.2 CAD-ohjelmien rajapinnat

Luvussa kartoitetaan kolmen CAD-ohjelman, Creon, SolidWorksin ja Solid Edgen rajapinnoista löytyviä automaattiseen mallinnustehtävien tarkastukseen soveltuvia työkaluja.

Taulukko 1: CAD-ohjelmissa saatavilla olevat mallintarkastuksen kannalta tarpeelliset ominaisuudet

	Creo	Solid Edge	SolidWorks
Piirteiden listaaminen	x	x	x
Vertailutyökalut	x		x
Osan mass properties -tiedot	x	x	x

Useat CAD-ohjelmat sisältävät erilaisia rajapintoja tai skriptikieliä, joiden avulla ohjelmien työkaluja on mahdollista käyttää ohjelmallisesti. Rajapintoja on käytetään yleisesti malliautomaation toteuttamiseen, jossa ulkopuolisella käyttöliittymällä tai muulla parametrien syöttötavalla muutetaan mallien parametreja ja tuotetaan niistä automaattisesti tietoa ja dokumentaatiota valmistusta varten.

Mallien tarkastuksen kannalta tarpeellisia ominaisuuksia rajapinnoissa olisivat:

- Työkalut, joilla on mahdollista suorittaa CAD-ohjelmissa makroja.
- Ohjelmista löytyvien vertailutyökalujen ajaminen ja tulosten hakeminen rajapinnan kautta. Tieto siitä, voiko vertailun suorittaa neutraaleilla tiedostoformaateilla.
- Tietojen lukeminen kappaleen pinta-alasta ja tilavuudesta.
- Funktio mallipuun piirteiden listaamiseen.

3.2.3 Creo

Creossa on saatavilla synkronisia ja asynkronisia rajapintoja. Creo TOOLKIT -rajapinnan avulla toteutetut asynkroniset lisäosat suorittuvat taustalla estämättä samanaikaista ohjelman käyttöä. Automaattista tarkastusta varten

asynkronisuudelle ei ole tarvetta, koska tarkastusohjelmaa ajettaisiin kerran jokaista vertailua kohden, ilman käyttäjää.

Creossa tiedot mallien sisältämistä piirteistä tallentuvat ohjelmaan myös tekstimuotoisina tiedostoina, joista olisi mahdollista lukea tiedot mallin sisältämistä piirteistä. Creon *Compare parts* -työkalun käyttö on mahdollista rajapintojen kautta. [33]

Creoon on versiopäivitysten yhteydessä lisätty uusia rajapintoja. Ohjelman versiossa 3.0 käytettävissä on 2.0-version sisältämien rajapintojen lisäksi C++ -rajapinta. Creo Parametric 3.0 -ohjelman ohjelmalliseen käyttöön on tarjolla on viisi rajapintaa:

- **VBAPI**
  Creon VBAPI on Windowsin COM API-rajapinnan kautta toimiva rajapinta, joka on käytettävissä sen rekisteröinnin jälkeen. Rajapinnan kautta voi käyttää kattavasti ohjelman eri työkaluja suoraan rajapinnan funktiolla. Myös makrojen suorittaminen on mahdollista, eli rajapinnan kautta voi käyttää mitä tahansa käyttöliittymän kautta saatavilla olevaa työkalua.

- **J-link**
  J-Link on Java-pohjainen rajapinta Creoon. Rajapinnassa tarjolla olevat komennot ovat samankaltaisia kuin VBAPI-rajapinnassa.

- **Web.link**
  Creon web.link-rajapinta on ActiveX-selainlisäosan kautta toimiva ohjelma, jonka avulla internet-sivuille upotetulla javascript-koodilla on mahdollista ajaa Creon komentoja. ActiveX-lisäosan asentaminen on ylimääräinen työvaihe ohjelman käyttäjälle ja nykyaikaisen web-kehityksen kannalta web.link-rajapinnan toteutusta voidaan pitää vanhanaikaisena.

- **Creo TOOLKIT**
  Creo Toolkit -rajapinnan avulla on mahdollista totetuttaa Creoon C-kielellä ohjelmoituja asynkronisesti toimivia lisäosia. Asynkroniset lisäosat toimivat taustalla eivätkä estä ohjelman samanaikaista käyttöä toisin kuin ohjelman kautta ajetut skriptit. Toolkit-sovellusten kehittäminen vaatii erillisen lisenssin, joka kuitenkin sisältyy oppilaitosten käytössä olevaan ohjelmistopakettiin.

- **Trail-tiedostot ja mapkey-makrot**
  Creo Parametric tallentaa kaikki ohjelman istunnon aikana käytetyt komennot trail-tiedostoon (polkutiedosto). Tiedostoon tallentuvat kaikki käyttöliittymän kautta tehdyt komennot, ja niitä on mahdollista tallentaa ja ajaa. Alkuperäinen käyttötarkoitus trail-tiedostoille on ollut mahdollistaa komentojen nopea uudelleensuorittaminen ohjelman kaatuessa. Creoon on mahdollista asettaa asetus (config.pro), joka pakottaa ohjelman suorittamaan trail-tiedoston komennot yksi askel kerrallaan:

```
set_trail_single_step = yes
```

Trail-tiedostoja on mahdollista käyttää myös ohjelman käytön automatisointiin. Tiedoston syntaksi on ihmisen luettavissa ja sitä on mahdollista muuttaa ohjelmallisesti. Tiedostoihin tallentuu myös automaation kannalta epäolennaisia komentoja, kuten kameran asennon muuttaminen viiveen kanssa, jotka on kuitenkin mahdollista poistaa tiedostosta ennen niiden suorittamista.

Trail-tiedostoja vastaavia komentosarjoja Creossa ovat myös mapkey-makrot. Creossa on mahdollista nauhoitta mapkey-makroja, jotka tallentavat käyttöliittymässä tehtyjä toimenpiteitä tekstimuotoisina tiedostoihin samassa muodossa kuin trail-tiedostot. Tallennettuja makroja on mahdollista suorittaa VBAPI-rajapinnan RunMacro-komennolla, joka ottaa syötteenä makron merkkijonona. On huomattava, että makroihin ja trail-tiedostoihin tallennetut komennot ja niiden nimet eivät pysy samoina Creon eri versioissa. Mikäli Creon versiota vaihdetaan, makroihin perustuva automaatio tulee määrittää uudelleen käyttöliittymän kautta.

3.2.4 SolidWorks

SolidWorks API on Visual Basic -pohjainen rajapinta, jonka kautta on mahdollista käyttää samoja ohjelmiston työkaluja kuin käyttöliittymänkin kautta. SolidWorksille kirjoitetut ohjelmat (“makrot”) suoritetaan ohjelman käynnistyttyä VB-makrotiedostoista.

SolidWorksin rajapinnan kautta on mahdollista käyttää ohjelman osien vertailuun tarkoitettuja *Compare parts* ja *Compare features* -työkaluja, tulostaa mallipuun piirteet ja hakea tietoja mallin pinta-alasta ja tilavuudesta. [32]

```
Dim swApp As SldWorks.SldWorks
Dim swModel As SldWorks.ModelDoc2
Dim swFeat As SldWorks.Feature
Dim swSubFeat As SldWorks.Feature

Sub main()
        Set swApp = Application.SldWorks
        Set swModel = swApp.ActiveDoc
        Set swFeat = swModel.FirstFeature
        While Not swFeat Is Nothing
        Debug.Print swFeat.Name
        Set swSubFeat = swFeat.GetFirstSubFeature
        While Not swSubFeat Is Nothing
    %           Debug.Print "    " & swSubFeat.Name
                Set swSubFeat = swSubFeat.GetNextSubFeature
        Wend
        Set swFeat = swFeat.GetNextFeature
```

```
        Wend
End Sub
```

SolidWorks-makro: Mallipuun piirteiden tulostaminen SolidWorks APIn kautta. (Esimerkki: Keith Rice) [34]

3.2.5 Solid Edge

Solid Edge on ominaisuuksiltaan vähemmän kattava kuin Creo, NX tai SolidWorks eikä siitä löydy samanlaista geometrian vertailutyökalua kuin SolidWorksista ja Creosta. Ohjelman valmistaja Siemens tarjoaa myös kattavampaa CAD-ohjelmistopaketti NX:ää.

Solid Edgen rajapinnan tarkasteluun on olemassa avoimen lähdekoodin projekti Solid Edge Spy `https://github.com/SolidEdgeCommunity/SolidEdgeSpy`

Solid Edgessä on ohjelman ominaisuuksien ohjelmalliseen käyttöön tarjolla kaksi rajapintaa: [35, s. 24]:

- Solid Edge Visual Basic .NET API
- Solid Edge C API

3.3 Teknisten ratkaisujen valinta

Järjestelmä toteutetaan VBAPI-rajapinnan kautta toimivana tarkastimena Creo Parametric -ohjelmistolle. Valinta ei perustu Creon tai siihen saatavilla olevien työkalujen teknisiin ominaisuuksiin vaan on puhtaasti käytännöllinen: Creo on käytössä yliopiston CAD-perusopetuksessa. Lisäksi Creon eri rajapinnoista löytyvät työkalut ovat käytännössä samoja.

Samanlaisen tarkastusohjelman olisi voinut toteuttaa myös muiden valmistajien CAD-ohjelmille, joista löytyy samanlaisia työkaluja tarkastuksessa tarvittavien tietojen lukemiseksi. Järjestelmän olisi samankaltaisella työmäärällä voinut toteuttaa myös muiden valmistajien CAD-ohjelmien työkaluilla tai valmistajariippumattomasti neutraaleja tiedostoformaatteja tukevilla avoimilla ohjelmilla ja ohjelmistokirjastoilla.

Mainituista tarkastuksista toteutetaan prototyyppiin ainakin pinta-alan, tilavuuden ja mallipuun vertailu sekä mahdollisuus asettaa malliin eri parmetriyhdistelmiä ja regeneroida se. Järjestelmän tulee tukea myös vertailua neutraaleilla tiedostoformaateilla, mallipuun piirteisiin liittyviä tarkastuksia

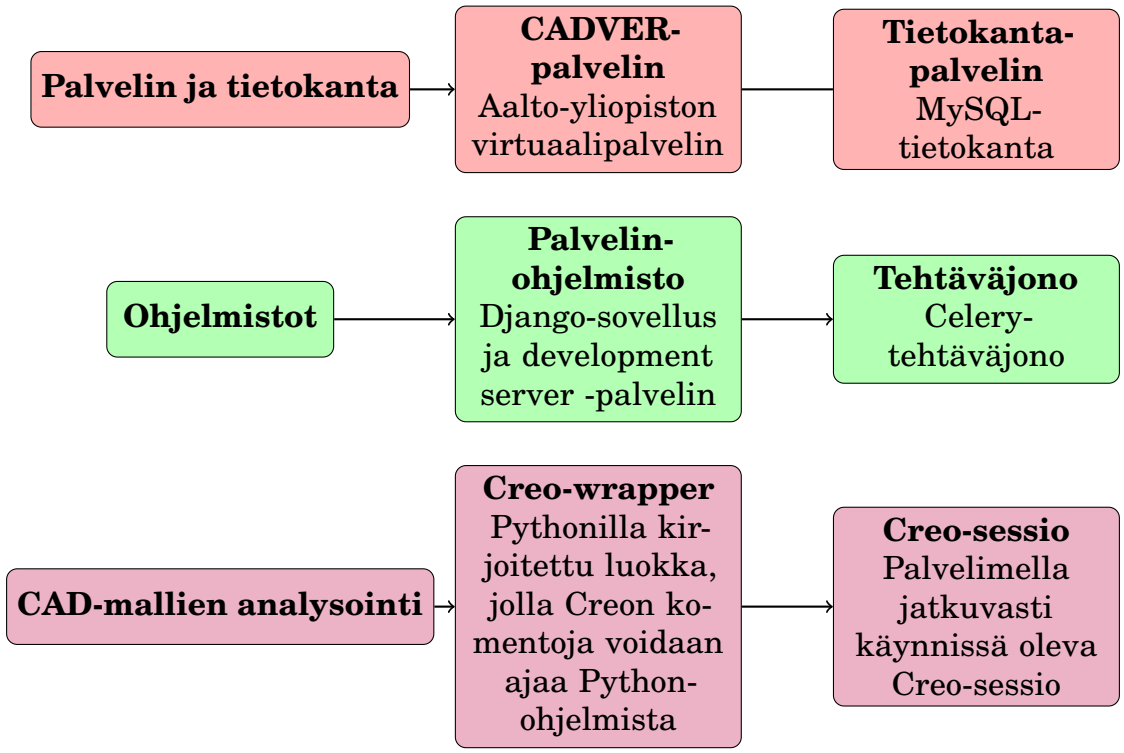
ei voida neutraaleilla tiedostoformaateilla kuitenkaan tehdä. Järjestelmää testataan ennen käyttöönottoa ajamalla siitä erilaisia virheellisiä ja eri tavoilla mallinnettuja oikeita malleja läpi.

3.4 Tarkastusjärjestelmän toteutus

Tarkastusjärjestelmä toteutetaan web-sovelluksena. Käyttöliittymä ja toiminnallisuus toteutetaan Python-kielellä Django-nimisellä verkkosovellusten kehitykseen tarkoitetulla kehitysympäristöllä. Käyttäjät kirjautuvat sisään järjestelmään, valitsevat tehtävän, siirtävät tarkastettavat tiedostot selaimen kautta sovellukseen, ja sovellus näyttää mitkä tehtävälle määritellyt tarkastukset läpäistiin.

Varsinaiset tarkastukset malleille toteutetaan vertailemalla järjestelmään oikeaksi määriteltyä ja palautettua mallia keskenään Creon VBAPI-rajapinnan kautta saatavilla olevilla työkaluilla. Palvelimelle saapuvien HTTP-pyyntöjen kautta lähetetään tiedosto tarkastettavaksi. Palvelinkoneella on jatkuvasti käynnissä Creo-sessio, jossa tarkastuksia ajetaan.

3.4.1 Järjestelmän rakenne



Kuva 9: CADVER-järjestelmän osat

Tarkastusjärjestelmä koostuu useasta palvelimelle asennetusta ohjelmasta: tietokannasta, web-palvelimesta, tehtäväjonosta ja jatkuvasti käynnissä olevasta Creo-sessiosta (kuva 9).

3.4.2 Tietomalli

Järjestelmän tietomalli on esitetty kaaviossa (kuva 10). Tarkastettavat tehtävät määritellään Assignment-tauluun (tehtävän nimi, kuvaus ja oikeaksi tiedetty tiedosto). Jokaiselle tehtävälle määritellään siihen liittyvät tarkastuspohjat (CheckTemplate) parametreineen. Tarkastuspohjat sisältävät tarkastuksen nimen ja kuvauksen, tarkastusfunktion nimen sekä vapaavalintaisia parametreja.

Tarkastusfunktioille syötettävät parametrit voivat sisältää mitä tahansa tarkastuksessa tarvittavaa tietoa, ja niihin voitaisiin tallentaa myös tietoa siitä mikä tarkastuksen tuloksen tai jonkin tarkistettavan arvon pitäisi olla. Parametrit voivat olla esimerkiksi lista malliin asetettavista parametreista JSON-muodossa[2]:

```
[{"SIZE_V": 1, "SIZE_H": 5, "THIN": true}, {"SIZE_V": 3, "SIZE_H": 2, "
    THIN": true}, {"SIZE_V": 2, "SIZE_H": 1, "THIN": true}, {"SIZE_V":
    5, "SIZE_H": 1, "THIN": false}, {"SIZE_V": 1, "SIZE_H": 1, "THIN":
    false}]
```

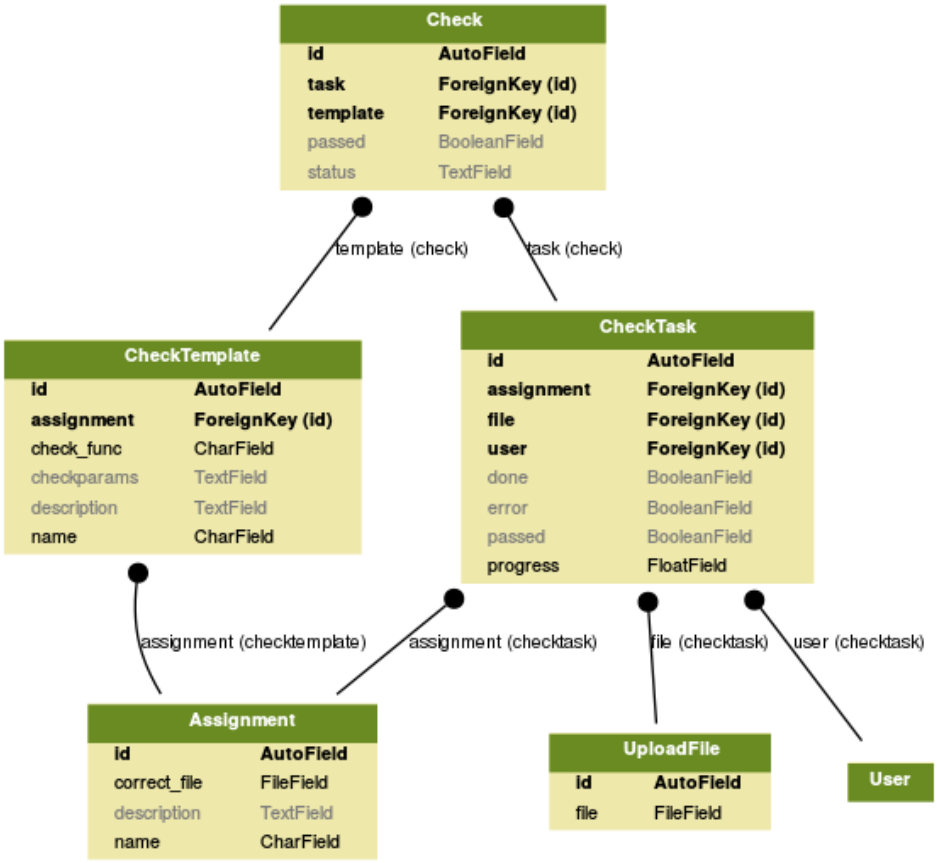
CheckTemplatesta JSON-muodossa tarkastusfunktiolle syötettävä lista parametreja. Parametrit tallennetaan tietokantaan tarkastusfunktioihin liitettyinä. Järjestelmä tukee sekä numeroarvoisten että boolean-parametrien asettamista malleihin.

Kun tarkastus ajetaan, luodaan uusi tarkastustehtävä (CheckTask), johon tallennetaan tiedot kokonaistarkastuksen tilasta ja tarkastuksen suorittaneesta käyttäjästä. Jokaisen tehtävään määritellyn tarkastuspohjan (CheckTemplate) pohjalta luodaan tarkastustehtävään liittyvä tarkastus (Check), johon tiedot yksittäisten tarkastusten suorittamisesta ja läpimenosta tallennetaan.

3.4.3 Celery-tehtäväjono

Tarkastusjärjestelmän on pystyttävä samanaikaisesti sekä suorittamaan varsinaisia tarkastuksia että vastaamaan niiden tilaa koskeviin kyselyihin. Asynkroninen tarkastusten suoritusmahdollisuus on tärkeä myös järjestelmän skaalautuvuuden kannalta. Creon VBAPI-rajapinta ei mahdollista yhtäaikaista yhteyttä useampaan kuin yhteen Creon sessioon.

[2]JSON on standardoitu tiedostomuoto, jossa tietoa välitetään avain/arvo-pareina. [36]

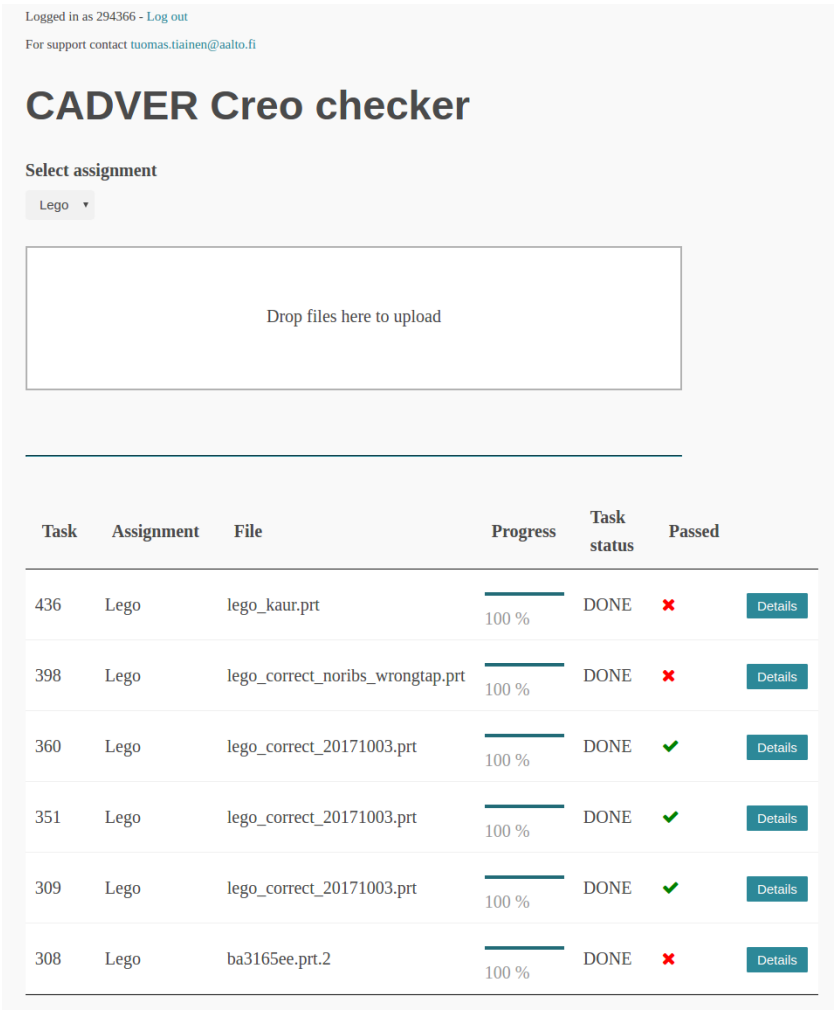


Kuva 10: Kaavio tarkastusjärjestelmän tietokannan rakenteesta.

Celery on Pythonilla toteutettu asynkroninen tehtäväjonokirjasto, joka suorittaa tehtäviä asynkronisesti eri prosessissa kuin varsinainen palvelinohjelma. Jonon toiminta perustuu siihen että funktiokutsuja parametreineen lähetetään jonon kautta “työntekijöille” suoritettavaksi myöhemmin. Celery on avoimen lähdekoodin projekti. [37]

Tehtäväjono tarkastusjärjestelmässä vastaa kahteen tarpeeseen. Useamman eri palvelimen käyttäminen tarkastamiseen on tehtäväjonon avulla mahdollista. Lisäksi on varmistuttava siitä, että kuhunkin Creo-sessioon on kullakin hetkellä avoinna vain yksi rajpaintayhteys. Tarkastusjärjestelmän tapauksessa Celery on konfiguroitu niin, että kerrallaan suorittuu vain yksi tehtävä. Koska tarkastuksia ajetaan Creon VBAPI-rajapinnan kautta, yksi käynnissä oleva Python-prosessi voi olla yhteydessä vain yhteen Creon sessioon kerrallaan [37]. Tehtäväjonon kautta olisi myös mahdollista jakaa tarkastustehtäviä käsittelevät Celery-workerit ja Creo-sessiot useammalle eri palvelimelle. HTTP-kutsuja ja tiedostoja käsittelevä palvelin voisi olla suorituskyvyltään vähäisempi kuin Creo-sessioita suorittavat palvelimet.

3.4.4 Käyttöliittymä



Kuva 11: Järjestelmän web-käyttöliittymä.

Sovellukseen toteutettiin yksinkertainen web-käyttöliittymä, jonka kautta tiedostot palautetaan tarkastettavaksi (kuva 11). Palautetut tiedostot ja niiden tarkastusten tila näytetään tiedostojen palautusalueen alla taulukossa. Taulukosta pääsee tarkastuskohtaiseen näkymään, jossa yksittäisten tarkastusten tietoja voi tarkastella. Sovellukseen toteutettiin myös kurssihenkilökunnalle tunnusten ja tehtävien hallintaan tarkoitettu salasanasuojattu näkymä.

3.4.5 Tarkastukset Creon VBAPI-rajapinnan kautta

Tarkastuksessa tarvittavien tietojen lukemiseen järjestelmän keskeinen osa on Creon VBAPI-rajapintaan Pythonilla kirjoitettu ohjelmistokerros (liite A), joka mahdollistaa Creon eri ominaisuuksien ohjelmallisen käyttämisen.

Luokka pitää automaattisesti kirjaa avoimista tiedostoista ja ikkunoista sekä varmistaa, että avatut yhteydet tulevat suljetuiksi.

PythonCreoConnection-luokkaa ja siitä löytyviä funktiota on mahdollista käyttää tarkastuksen toteuttamiseen. Python-luokkaan on toteutettu valmiiksi metodit seuraavien Creon toimintojen käyttämistä varten:

- Tiedoston avaaminen
- Tiedoston sulkeminen
- Ikkunan aktivoiminen
- Parametrien asettaminen
- Mallin regenerointi

Luokan kautta on mahdollista myös ajaa mitä tahansa VBAPI-rajapinnan komentoja conn-muuttujaan tallennetun connection-objektin kautta. Esimerkiksi mallipuun piirteiden listaukseen käytetään VBAPI-rajapinnan GetFeatureByName-komentoa. Kaikista Pythonin win32com-rajapinnan kautta saatavilla olevista VBAPI-rajapinnan komennoista muodostettiin osana projektia tekstitiedosto.

VBAPI-rajapinta on asennettava Creon asennuksen yhteydessä lisäosana. Ennen kuin rajapintaa voi käyttää, on asetettava tietyt ympäristömuuttujat (taulukko 2) ja rekisteröitävä rajapinta suorittamalla järjestelmänvalvojan oikeuksilla Creon asennuskansiosta löytyvä ohjelma vb_api_register.bat.

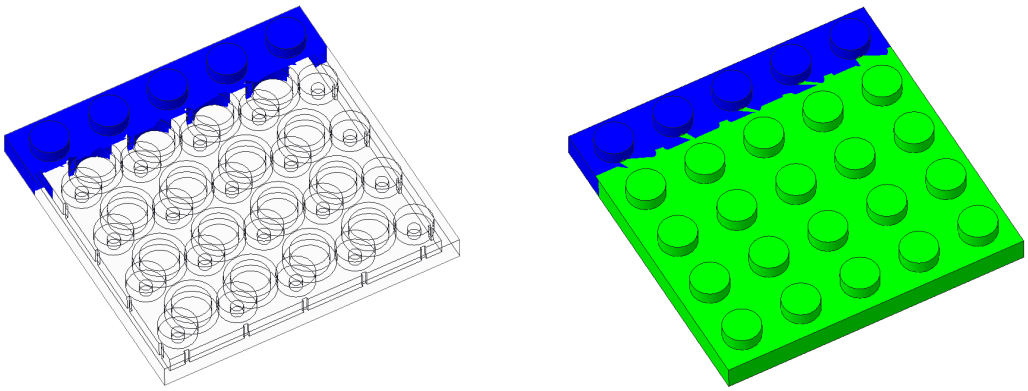
Taulukko 2: VBAPI-rajapinnan rekisteröintiä varten tarvittavat ympäristömuuttujat

PRO_COMM_MSG_EXE	"C:/Program Files/PTC/Creo 3.0/M060/ Common Files/x86e_win64/obj/pro_comm_msg.exe"
PRO_DIRECTORY	"C:/Program Files/PTC/Creo 3.0/M060/Parametric/"

Pythonin win32com-kirjaston kautta käytettynä VBAPI-rajapintaan lähetettävät parametrit tai polkujen nimet eivät saa sisältää erikoismerkkejä kuten Ö- tai Ä- kirjaimia.

Lisäksi Creon asetuksiin (config.pro) on mallien regenerointia varten asetettava regenerointivirheiden käsittelyyn vaikuttava asetus sekä mallin *mass properties* -tietojen automaattiseen päivitykseen liittyvä asetus:

```
regen_failure_handling = resolve_mode
mass_property_calculate = automatic
```



Kuva 12: Kaksi esimerkkiä Creon compare geometry -työkalulla tuotetuista vertailukuvista kahdesta osasta, jotka eroavat toisistaan. Työkalun avulla on mahdollista esittää eri väreillä osien eroavat ja yhtenevät tilavuudet. Vertailussa on päällekkäin 4x5 ja 5x5 -versiot Lego-mallista. Ohjelmallinen tarkastus voisi perustua esimerkiksi vihreän ja sinisen värin tunnistamiseen työkalun tuottamasta kuvasta. Compare geometry -työkalun ohjelmallinen käyttö Creossa on mahdollista trail-tiedostojen sekä mapkey-makrojen avulla.

3.4.6 Tarkastus trail-tiedostoilla

Session aikana Creo tallentaa jokaisen ohjelmassa käyttöliittymän kautta suoritetun komennon trail-tiedostoon. Trail-tiedostoja on mahdollista tallentaa ja suorittaa myöhemmin ja Creon käyttöä on mahdollista automatisoida niiden avulla. Mapkey-makrot ja trail-tiedostot käyttävät tiedon tallentamiseen samaa syntaksia, joka ei ole yhteensopiva Creon eri versioiden välillä.

Trail-tiedostoihin tallentuu myös paljon automaattisen käytön kannalta mallintarkastuksessa epäolennaista tietoa, kuten tietoa kuvakulman muutoksista tai hiiren liikkeistä. Trail-tiedostojen tehokasta käyttöä varten olisi tarpeen toteuttaa ohjelma, jonka avulla tiedostoista olisi mahdollista joustavasti poistaa sovellutuskohteen kannalta epäolennaista tietoa.

Osana projektia suoritettiin kokeiluja trail-tiedostolla, jossa tiedostopolkua muuttamalla olisi mahdollista ajaa Creon compare geometry -työkalu kahdelle eri osatiedostolle. Compare geometry -työkalulla on mahdollista tuottaa vertailluista osista kuva, jossa osien (tietyllä, määriteltävällä toleranssilla) yhtenevä geometria näytetään vihreällä ja eroava geometria sinisellä (kuva 12).

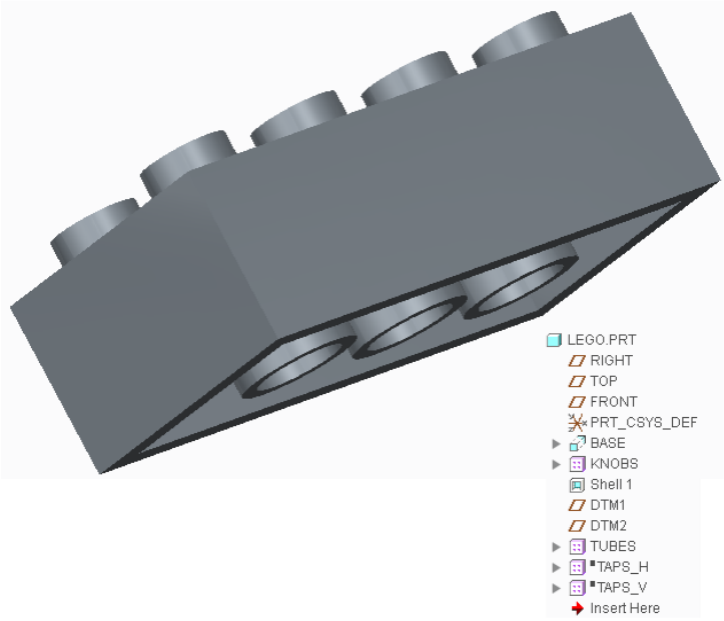
Kuvan tallentaminen mallintarkastusjärjestelmän käsiteltäväksi Creosta olisi mahdollista trail-tiedoston avulla, VBAPI-rajapinnan kautta ajettavalla makrolla tai VBAPI-rajapinnasta valmiiksi löytyvällä kuvantallennusfunktiolla. Varsinainen tarkastus olisi mahdollista toteuttaa tästä kuvasta esimerkiksi tunnistamalla löytyykö kuvasta sinistä väriä (eroavaa geometriaa). Kuvan palauttaminen voisi muutenkin olla hyödyllistä opiskelijalle tehtävän virheiden havainnollistamiseksi. Lisäksi kuvasta saa heti käsityksen siitä, mikä malli järjestelmään on palautettu.

4 Tarkastusjärjestelmän käyttöönotto ja kokeilu

Osana työtä toteutettu mallintarkastusjärjestelmä konfiguroitiin käyttövalmiiksi tarkastamaan Aalto-yliopiston tietokoneavusteisten työkalujen käyttöön keskittyvällä kurssilla *Machine Design* käytettävää parametrista Lego-mallinnustehtävää (kuva 13). Kurssin opiskelijoille annettiin pakollisena osasuorituksena mallinnustehtävän palautus ja tarkastusjärjestelmän toimintaa käsittelevään kyselyyn vastaaminen.

4.1 Tehtävänanto

Joitakin Aalto-yliopiston kursseilla käytössä olevia tehtävänantoja tulee muuttaa automaattista tarkastusta varten siten, että ne ovat tarkastettavissa. Tehtävänantojen muuttaminen on useissa tapauksissa mahdollista toteuttaa pelkästään loppuun liitettyinä palautusohjeina, joissa ohjeistetaan miten mallia tulee palautusta varten muuttaa. Jos kurssilla käytetään automaattista tarkastusjärjestelmää, olisi sen rajoitteet hyvä huomioida jo siinä vaiheessa kun kurssin tehtäviä laaditaan.



Kuva 13: Lego-tehtävässä mallinnettava kappale ja mallin mallipuu (kuva: Lego-tehtävän tehtävänanto).

Järjestelmän kokeilussa käytetään Lego-mallinnustehtävää. Peruskursseilla käytössä olevan tehtävän tarkoituksena on opettaa ja havainnollistaa parametrista mallinnusta. Tehtävässä mallinnetaan Lego-palikka, jonka leveys, pituus ja paksuus ovat parametrisesti säädettävissä. Tehtävään sisältyy Creon

Pro/Program-ominaisuuden avulla toteutettuja ehtoja: mallin piirteitä on oltava mahdollista kytkeä päälle ja pois, kun parametreja muutetaan: kun Legon leveys johonkin suuntaan on 1, mallin on tuotettava kappaleen keskilinjalle ontot tapit.

Opiskelijoiden tehtävässä tyypillisesti tekemät virheet liittyvät siihen että kaikkia erikoistapauksia ei ole otettu huomioon, esimerkiksi 1x1-muotoa tai mallin generoitumista kaikilla leveyksillä molempiin suuntiin.

4.2 Käyttöönotto

Tarkastusjärjestelmä otettiin käyttöön Aalto-yliopiston CADVER-palvelimelle osoitteessa `http://cadver.org.aalto.fi/`. Palvelimelle asennettiin järjestelmä, sen tarvitsemat apuohjelmat (tietokantaohjelmisto, Python-tulkki, Python-Win32com VBAPI-rajapinnan käyttämiseen sekä Python-kirjastoja) ja Creo.

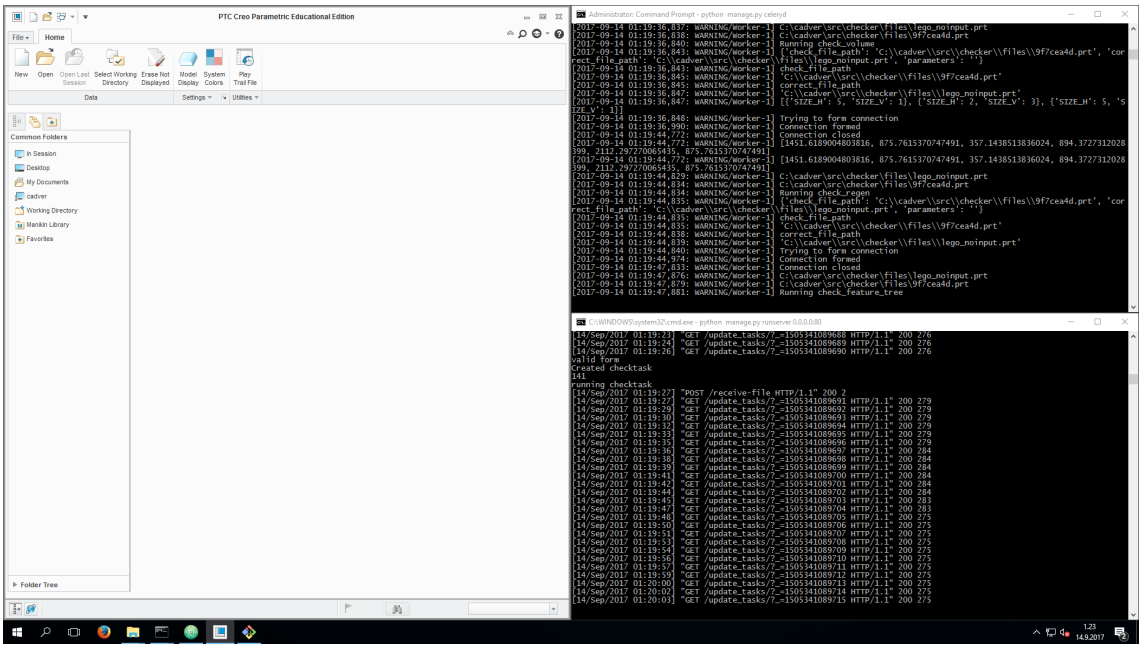
Tarkastusjärjestelmä integroitiin Aalto-yliopiston MyCourses-järjestelmään, josta saatavalla linkillä opiskelijanumeron tallentava sessio aktivoitiin tarkastusjärjestelmään.

Ennen palautuksen avaamista opiskelijoille järjestelmän toimintaa testattiin erilaisilla oikeilla ja väärillä mallitiedostoilla sekä kymmenellä edellisen vuoden CAD-kurssin suorittaneen opiskelijan tehtävällä.

Kokeilua varten järjestelmä konfiguroitiin käyttämään pinta-alan, tilavuuden ja mallin regeneraation tarkastusfunktioita. Toteutettu mallipuiden yhteneväisyyden vertailu havaittiin testeissä liian tiukaksi ja se jätettiin pois. Varsin yksikäsitteisistä ohjeista huolimatta samanlainen malli on mahdollista toteuttaa usealla eri tekniikalla. Mallipuun tarkastuksessa voisi täyden samankaltaisuuden vertailun sijasta käyttää erilaisia laskettuja samankaltaisuuden vertailulukuja tai tarkastaa, että vähintään tietyt piirteet esiintyvät mallipuussa tietyssä järjestyksessä.

4.3 Tehtävien palautus ja kyselytutkimus

Aalto-yliopiston *Machine Design* -kurssi on tarkoitettu suoritettavaksi maisterivaiheen opintojen ensimmäisen vuoden aikana – enemmistöllä opiskelijoista on CAD-kokemusta aikaisemmista kandidaatin tutkinnon opinnoista. Kurssin opiskelijoille annettiin pakollisena kurssiin kuuluvana osasuorituksena Lego-tehtävä, joka tuli palauttaa automaattiseen tarkastusjärjestelmään. Olemassa olevaa tehtävänantoa täydennettiin palautusohjeilla. Lisäksi tehtävänannosta jätettin pois vapaavalintainen lisätehtävä.



Kuva 14: CADVER-palvelimella käynnissä olevia ohjelmia: Creo sekä cmd-komentotulkissa web-palvelin ja Celery worker.

Palautuksen jälkeen opiskelijoilla teetettiin kyselytutkimus osana pakollista tehtävää (Liite B). Kyselyn tarkoituksena oli mitata opiskelijoiden esitietoja, automaattisen tarkastusjärjestelmän käyttäjäkokemusta sekä oppimiskokemusta. Koska kurssin opetuskielenä on englanti, kysely toteutettiin englanniksi.

5 Tulokset

Luvussa esitetään havaintoja järjestelmän toiminnasta kokeilun aikana ja analysoidaan toteutetun kyselytutkimuksen tuloksia. Tarkastusjärjestelmää pidettiin auki kaksi viikkoa ja tehtävän palautti 102 opiskelijaa, joista 95 hyväksytysti. [3] Kyselytutkimukseen vastasi 91 opiskelijaa.

Tehtävän hyväksytysti suorittaneet opiskelijat palauttivat sen keskimäärin 2,6 kertaa, hylätyt palauttivat keskimäärin 5,4 tiedostoa. Suuri osa palautuksista keskittyi järjestelmän aukiolon viimeiselle viikolle. Palautusaikana tarkastusjärjestelmän toiminnassa oli lyhyt käyttökatko yliopiston IT-palvelujen käynnistettyä palvelimen uudestaan ilman ennakkoilmoitusta.

Taulukko 3: Väittämämuotoisten kysymysten asteikko

1	Eri mieltä
2	Jonkin verran eri mieltä
3	Ei samaa mieltä eikä eri mieltä
4	Jonkin verran samaa mieltä
5	Samaa mieltä

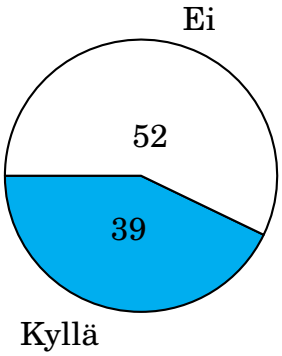
Osa kyselytutkimuksen kysymyksistä on väittämämuodossa, jossa vastaajan tarkoituksena on arvioida kuinka samaa mieltä, asteikolla yhdestä viiteen, vastaaja väittämän kanssa on (taulukko 3). Ei voida olettaa, että asteikko olisi lineaarinen, joten tällä tavalla kerätyn tiedon analysoinnissa keskiarvo ei ole hyvä mittari. Ottamalla tuloksista keskiarvo menetettäisiin lisäksi tietoa siitä miten vastaukset kysymykseen jakaantuvat. Tulosten arvioinnissa on esitetty kysymyskohtaisesti vastausten jakauma kullekin vastausvaihtoehdolle. Kyselytutkimuksen toteuttamisessa ja sen vastauksissa korostuvat muutenkin tyypilliset kyselytutkimuksissa esiintyvät haasteet: kysymysten yksiselitteinen asettelu on vaikeaa ja eri ihmiset saattavat kokea asteikot eri tavoilla.

5.1 Opiskelijoiden osaamistausta

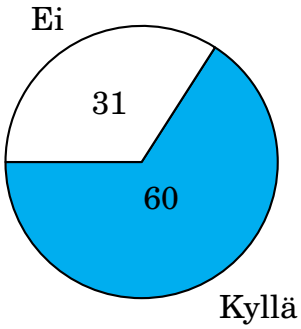
Kurssin opiskelijat arvioivat oman parametristen CAD-mallinnusohjelmien käyttökokemuksensa keskinkertaiseksi (kuva 16). Erityisesti kokemusta mainittiin olevan eniten Aalto-yliopiston kursseilla käytössä olevista ja opetettavista CAD-ohjelmistoista. Hieman yli puolella opiskelijoista oli aikaisempaa kokemusta Creo Parametric -ohjelmiston käytöstä.

Noin puolella opiskelijoista oli aikaisempaa kokemusta tehtävien palauttamisesta automaattisin tarkastusjärjestelmiin. Vapaissa vastauksissa korostuivat

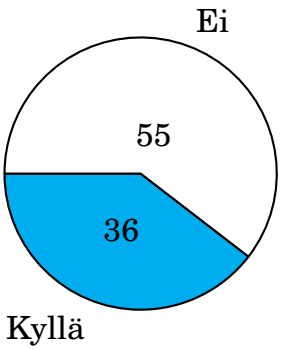
[3] Diplomityön kirjoittajan huomio: osallistuin kurssille myös itse, oma osuuteni on jääviyden vuoksi jätetty kaikista tuloksista pois.



Kuva 15: Oletko tehnyt Lego-tehtävän aikaisemmin?



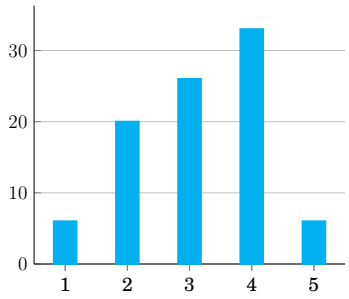
Kuva 16: Oletko käyttänyt Creo Parametric -ohjelmistoa aikaisemmin?



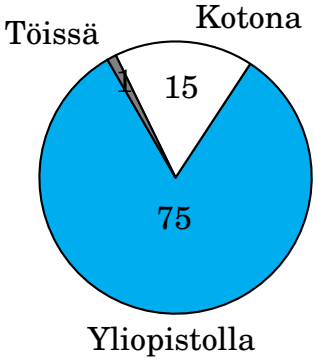
Kuva 17: Onko sinulla aikaisempaa kokemusta automaattisista tehtävien tarkastusjärjestelmistä?

erityisesti ohjelmoinnin kursseilla käytettävät automaattiset tarkastustyökalut (kuva 17).

Koska samaa Lego-tehtävää on käytetty aikaisemmin muilla Aalto-yliopiston kursseilla (Tietokoneavusteisen mallinnuksen peruskurssi, Tietokoneavusteiset työkalut insinööritieteissä), 31 opiskelijaa oli tehnyt saman tehtävän jo aikaisemmin (kuva 15).



Kuva 18: Kuinka kokenut parametristen CAD-mallinnusohjelmien käyttäjä olet? (Asteikolla 1-5)

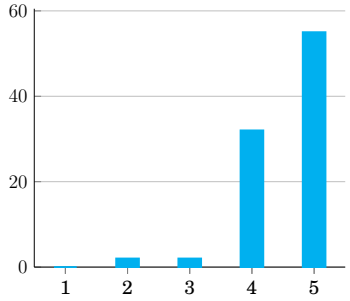


Kuva 19: Missä palautit tehtävän?

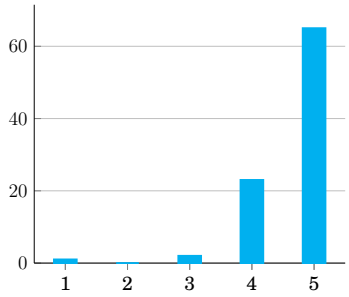
Suurin osa opiskelijoista palautti tehtävän yliopistolta (kuva 19) – kurssihenkilökunnan apua tehtävien suorittamiseen ja palauttamiseen oli kurssilla tarjolla kahdesti viikossa järjestettyjen harjoitusten aikana. Suurin osa opiskelijoista palautti tehtävän Firefox- tai Chrome-selaimella.

5.2 Käyttäjäkokemus

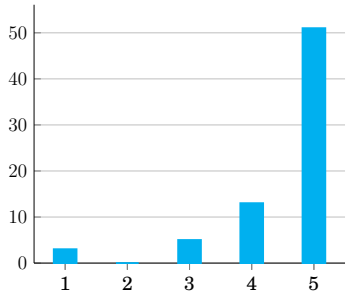
Tehtävänantoa ja ohjeita automaattisen tarkastusjärjestelmän käyttämiseksi pidettiin yleisesti ottaen riittävän selkeinä (kuvat 20 ja 21). Tarkastusjärjestelmän helppokäyttöisyys sai myös hyvän arvion (kuva 22), mikä ei ole yllättävää, koska järjestelmän käyttöliittymä on hyvin yksinkertainen. Se koostuu alueesta, jolle tarkastettavat tiedostot pudotetaan sekä taulukosta, joka näyttää viimeisten palautettujen tiedostojen tarkastuksen tulokset.



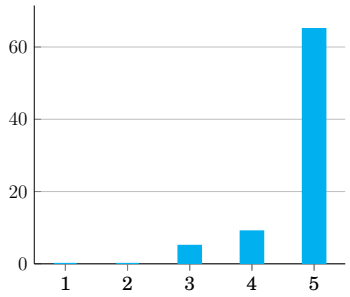
Kuva 20: Tehtävänanto oli selkeä.



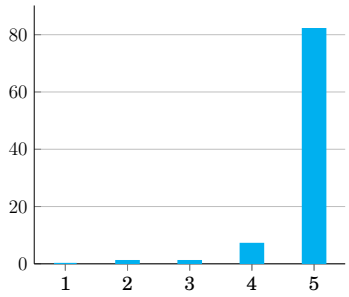
Kuva 21: Ohjeet automaattisen tarkastusjärjestelmän käyttöön olivat selkeät.



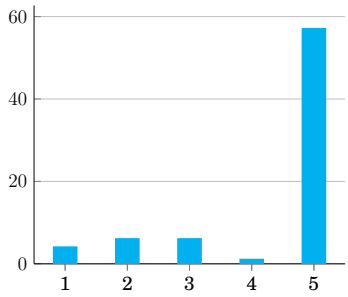
Kuva 22: Automaattisen tarkastusjärjestelmän käyttäminen oli helppoa.



Kuva 23: Pidän mahdollisuudesta palauttaa tehtäviä ajasta riippumatta.



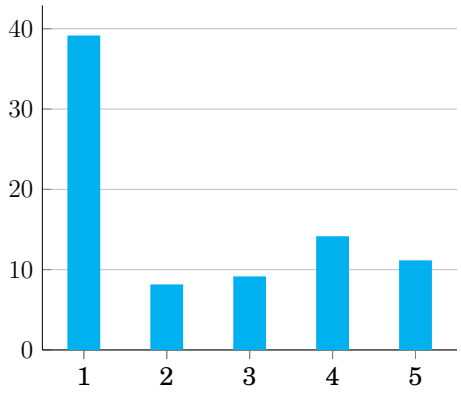
Kuva 24: Pidän mahdollisuudesta palauttaa tehtäviä paikasta riippumatta.



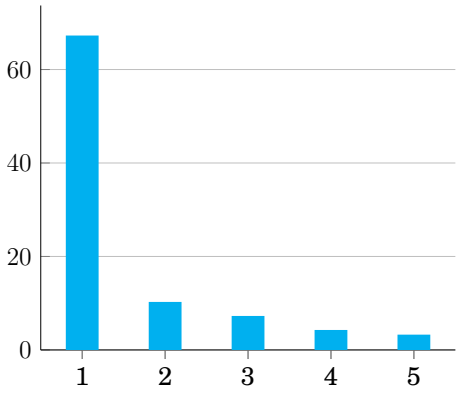
Kuva 25: Tarkastusjärjestelmän vasteaika oli riittävän lyhyt.

Järjestelmä toimi kokeilun aikana kiitettävästi. Erityisen positiivista palautetta kyselyyn vastanneet opiskelijat antoivat mahdollisuuksille palauttaa tehtäviä ajasta ja paikasta riippumatta (kuva 23 ja 24).

Järjestelmän vasteaikaa pidettiin yleisesti ottaen hyvänä (kuva 25). Suurin osa opiskelijoista ei tarvinnut kurssihenkilökunnan apua tehtävän tekemiseen (kuva 26) tai tarkastusjärjestelmän käyttämiseen (kuva 27).

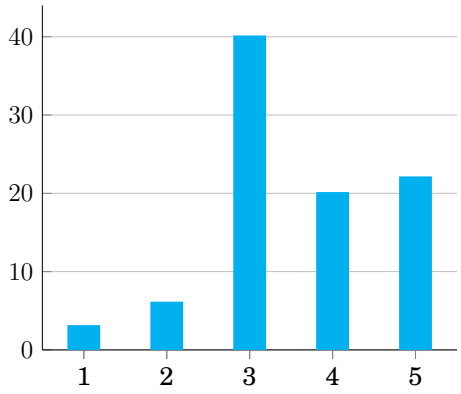


Kuva 26: Tarvitsin kurssihenkilökunnan apua tehtävän suorittamiseen.

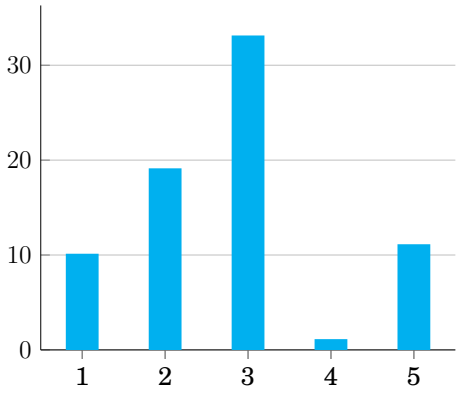


Kuva 27: Tarvitsin kurssihenkilökunnan apua automaattisen palautusjärjestelmän käyttöön.

5.3 Oppimiskokemus



Kuva 28: Automaattinen tarkastusjärjestelmä tuki oppimistani.

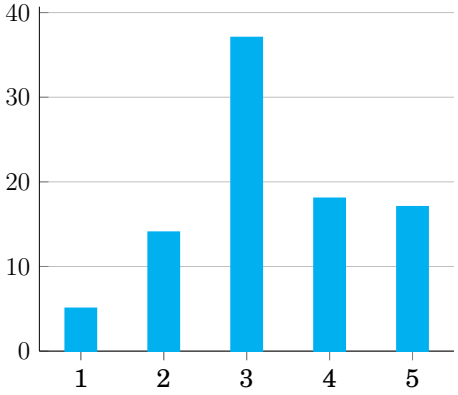


Kuva 29: Automaattisen tarkastusjärjestelmän antama palaute on riittävää.

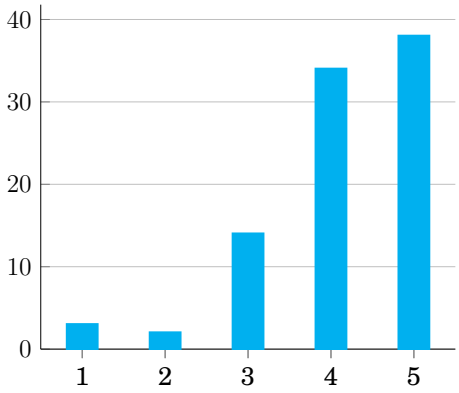
Järjestelmän antamaa palautetta mittaavien kysymysten perusteella järjestelmän antama palaute on puutteellista. On huomattava, että kokeilun aikana järjestelmä ei juuri antanut varsinaista *palautetta*, käyttöliittymässä esitettiin vain tieto palautetulle tiedostolle suoritettujen tarkastusten tuloksista. Opiskelijoiden luottamus järjestelmän tarkastuksen toimintaan oli kohtalainen.

Vapaissa palautekentissä saadun palautteen perusteella kaikki opiskelijat eivät ymmärtäneet, että järjestelmä kokeilee mallia eri parametriyhdistelmillä. Tarkempaa tietoa järjestelmän tarkemmasta toiminnasta ei käynyt ilmi järjestelmän käyttöliittymästä tai tehtävän palautusohjeista.

Järjestelmän käyttöliittymässä esitettiin punainen rasti merkkinä tehtävästä,

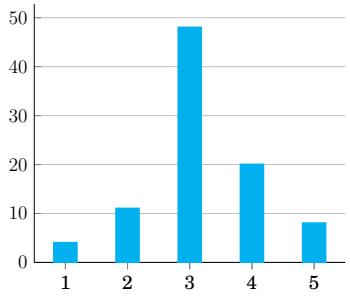


Kuva 30: Ymmärsin tarkastusjärjestelmän antaman palautteen.

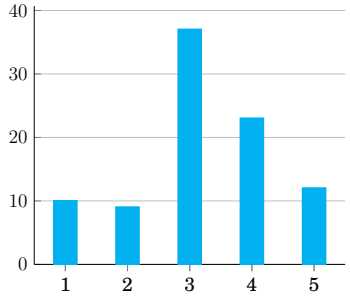


Kuva 31: Luotan, että järjestelmä tarkasti tehtäväni oikein.

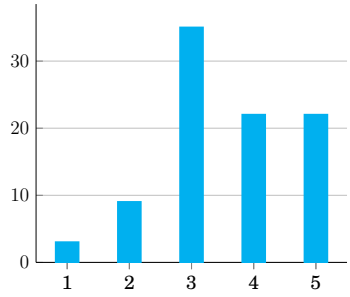
jota ei ole vielä tarkastettu oikeaksi (kuva 11). Palautteen perusteella punaisen rastin välittämä viesti sekoittuu siihen, että tehtävä olisi tarkastettu ja ei mennyt läpi. Tarkoituksenmukaisempaa olisi esittää punainen rasti vasta, jos tehtävä on varmuudella havaittu väärin tehdyksi.



Kuva 32: Automaattisen tarkastusjärjestelmän käyttö oli motivoivaa.



Kuva 33: Automaattisen tarkastusjärjestelmän käyttö oli innostavaa.



Kuva 34: Palautan tehtävät mieluummin automaattisen järjestelmään kuin kurssihenkilökunnalle.

Kyselyn perusteella automaattisen tarkastusjärjestelmän käyttö koetaan jossain määrin motivoivaksi ja innostavaksi (kuvat 32 ja 33). Useampi kyselyyn vastannut opiskelija palauttaisi mieluummin tehtäviä automatisoituun järjestelmään kuin kurssihenkilökunnalle (kuva 34).

6 Johtopäätökset

6.1 Yleisiä johtopäätöksiä

Tässä diplomityössä tehty katsaus tarjoaa pohjan automaattisen tarkastuksen jatkokehittämiselle. Tehdyn kokeilun perusteella on selvää, että konetekniikan parametristen CAD-mallinnustehtävien automaattinen tarkastus ja arvostelu riittävän laadukkaasti on teknisesti mahdollista. Teollisuudessa jo käytössä olevia työkaluja kuten CAD-ohjelmien rajapintoja ja mallien tarkastustyökaluja on mahdollista käyttää ohjelmallisesti myös mallinnustehtävien oikeellisuuden tarkastamiseen ja arvosteluun.

Vaikka kokeilussa käytetyt tarkastukset olivat yksinkertaisia, toteutettu tarkastusjärjestelmä mahdollistaa joustavasti myös monipuolisempien tarkastusfunktioiden toteuttamisen esimerkiksi CAD-ohjelmien makrojen avulla. Mallipuun, useiden muiden mallin ominaisuuksien (esimerkiksi mallin metatietojen), ja geometrisen muodon vertaileminen on mahdollista ohjelmallisesti käyttämällä CAD-ohjelmista valmiiksi löytyviä työkaluja ja rajapinnoista löytyviä työkaluja.

Samanlaisia menetelmiä voisi soveltaa myös 2D-piirustustehtävien tarkastukseen, mutta niiden tarkastus olisi huomattavasti vaikeampaa. Kolmiulotteisista malleista muodostuu hierarkkinen mallipuu ja kolmiulotteisilla malleilla on enemmän sellaisia ominaisuuksia, joista on mahdollista saada määrämuotoista tietoa tarkastusten suorittamista varten.

Peruskurssien ensimmäisten tehtävien tasoisten tehtävien tarkastaminen on helppoa, ne ovat usein joko yksiselitteisen oikein tai väärin. On helppoa tarkastaa, ovatko mallit täsmälleen samanlaisia. Parametrisia malleja on myös mahdollista regeneroida eri parametriyhdistelmillä ennen samankaltaisuuden tarkastamista.

Parametristen mallien tarkastuksen automatisointi vaatii jonkin verran manuaalista työtä, mutta kun työ tehtävien tarkastuksen automatisoimiseksi on kerran tehty, lisäopiskelijoista seuraava kustannus on hyvin pieni. Automaattisen tarkastuksen tehokas käyttö edellyttää, että tehtävänantojen pitää olla laadittuja automaattista tarkastusta silmälläpitäen. Automaattista tarkastusmenetelmää käytettäessä kurssien tehtäväkierrokset voisivat koostua useammasta pienemmästä osapalautuksesta kuin kursseilla, joilla assistentit tarkastavat tehtäviä käsin.

CAD-kursseista olisi myös teknisesti mahdollista tehdä avoimia MOOC-kursseja, mutta kaupallisten CAD-ohjelmistojen lisenssiehdot rajoittavat kurssien potentiaalista osallistujamäärää. Ohjelmoinnin opetuksessa käytettävät työkalut ovat useimmiten avointa lähdekoodia tai muuten vapaasti käytettävissä, mutta CAD-ohjelmistojen opiskelijalisenssejä on tyypillisesti tarjolla

ainoastaan oppilaitoksissa kirjoilla oleville opiskelijoille ja henkilökunnalle.

Toteutetussa tarkastusjärjestelmässä käytettiin tarkastuksiin CAD-ohjelmista jo valmiiksi löytyviä työkaluja. Mikäli muotoa vertailevan ohjelman haluaisi toteuttaa itse ilman riippuvuutta CAD-ohjelmista, tulisi mallit trianguloida tai muuttaa pistepilviksi ja käyttää algoritmia, joka sovittaa mallit päällekkäin ja vertailee niitä. Näiden toimintojen toteuttamiseen on olemassa valmiita ohjelmistokirjastoja. Vaikka suljetun lähdekoodin kaupallisten CAD-ohjelmien työkalujen sisäisestä toiminnasta ei ole saatavilla tietoa, on todennäköistä, että CAD-ohjelmien geometrian vertailutyökalut toimivat samalla periaatteella.

Opiskelijoilta on aikaisemmissa tutkimuksissa saatu positiivista palautetta automaattisista tehtäväntarkastusohjelmista ja mahdollisuuksista palauttaa tehtäviä vapaassa aikataulussa paikasta riippumatta. Sama tulos toistui toteutetussa kyselytutkimuksessa – vastauksista voidaan vetää johtopäätös, että erityisesti aikaan ja paikkaan liittyvien rajoitteiden voittaminen on automaattisen tarkastusjärjestelmän suurimpia hyötyjä. Oppimiseen, motivaatioon ja innostavuuteen liittyvissä kysymyksissä opiskelijoiden vastaukset hajaantuivat huomattavasti enemmän. On huomattava, että kyselytutkimuksen avulla ei voi luotettavasti tutkia automaattisen tehtävien tarkastuksen todellisia vaikutuksia oppimiseen. Erilaista opetusta saavia vertailuryhmiä tutkimalla olisi mahdollista saada luotettavampaa tietoa.

Edistyneimmät tarkastusjärjestelmät ovat ohjelmointitehtävien tarkastamiseen kehitettyjä. Tämä johtunee siitä, että testit ovat muutenkin osa nykyaikaista ohjelmistokehitystä, siinä missä konetekniikan alalla ei ole yhtä paljon samanlaisia kaupallisia tarpeita. Järjestelmien arkkitehtuuri saattaa olla sellainen, että niihin toteutettujen valmiiden osien käyttäminen CAD-tehtävien tarkastuksessa voisi olla mahdollista. Esimerkiksi Aalto-yliopistossa ohjelmoinnin kursseilla käytössä olevaa A+-järjestelmää olisi mahdollista käyttää myös mallinnustehtävien tarkastukseen. A+-järjestelmässä käyttöliittymä ja tehtävänannot toimivat erillään varsinaisista tarkastuksista ja se sisältää valmiiksi esimerkiksi sisäänkirjautumisen Aalto-yliopiston tunnuksilla.

6.2 Tarkastusjärjestelmän jatkokehitys

Saadun palautteen ja käyttökokemusten perusteella automaattisen tarkastusjärjestelmän jatkokehitystä on mahdollista suunnata. Kyselyssä saadussa palautteessa korostui erityisesti järjestelmän tarkastuksista antaman palautteen puutteellisuus. Toteutettavissa olisi myös useita teknisiä parannuksia järjestelmän ja tarkastusten toimintaan. Tarkastuksia olisi myös mahdollista nopeuttaa huomattavasti suhteellisen yksinkertaisilla toimenpiteillä.

Parannusehdotuksia käyttöliittymään ja järjestelmän antamaan palautteeseen:

- Järjestelmään voisi lisätä ohjesivun, jolla kerrotaan, kuinka järjestelmä malleja tarkastaa.
- Koko tarkastusprosessista tulisi tallentua tietokantaan lokimerkintöjä, joista käy ilmi, miten tarkastusjärjestelmä mallit tarkastaa. Kyselytutkimuksen perusteella kaikki opiskelijat eivät ymmärtäneet, että tarkastusta varten järjestelmä regeneroi malleja eri parametriyhdistelmillä.
- Järjestelmän tuottaman palautteen tarkastuksista tulisi olla opiskelijoille selkeämpää. Jos malli ei läpäise tarkastusta, pitäisi esittää mahdollisimman tarkasti tietoa siitä, mistä mallissa olevat virheet löytyvät (esimerkiksi parametrit, joilla malli ei täsmää tai compare geometry -työkalulla tuotettu kuva).
- Tarkastuksen tulokset ja virheilmoitukset tulisi mahdollisuuksien mukaan esittää selkokielisinä.
- Järjestelmän käyttöliittymässä tulisi selkeämmin esittää milloin palautettu tehtävä odottaa tarkastusta, on hylätty tai hyväksytty. Prototyypissä ollutta punaista rastia (merkkinä siitä että malli ei ole vielä läpäissyt tarkastusta) ei tulisi näyttää ennen kuin malli todella on tarkastettu virheelliseksi.

Parannusehdotuksia tarkastusjärjestelmän teknisiin ominaisuuksiin:

- Järjestelmän tulisi tukea monipuolisempia tarkastuksia. Erityisesti VBAPI-rajapinnan kautta suoritettavien mapkey-makrojen avulla voisi olla mahdollista toteuttaa monipuolisempia tarkastuksia.
- Oikeiden mallien tarkastustuloksia olisi mahdollista tallentaa välimuistiin sellaisten tarkastusten osalta, jotta jokaista tapausta varten ei tarvitse regeneroida molempia malleja.
- Tarkastusten nopeuttamiseksi mallit voisi avata CAD-ohjelmassa vain yhden kerran ja lukea samalla kaikki tarkastuksessa tarvittavat tiedot. Nyt toteutetussa järjestelmässä malli suljetaan ja avataan uudelleen esimerkiksi tilavuuden ja pinta-alan tarkastusten välissä.
- Useampi Creo-sessio voisi suorittaa tarkastuksia samaan aikaan tehtäväjonon avulla. Sessiot eivät kuitenkaan voisi olla käynnissä samassa käyttöjärjestelmässä sillä VBAPI mahdollistaa vain yhden samanaikaisen yhteyden.
- Järjestelmän tulisi tukea useamman erillisen kurssin tehtävien yhtäaikaista tarkastusta. Järjestelmän aktivointilinkissä voitaisiin aktivoida myös kurssi, jonka alle tehtävät kuuluvat.

7 Yhteenveto

Työssä suoritettiin kirjallisuuskatsaus CAD-opetukseen, kolmiulotteisen tiedon tallentamiseen ja vertailuun sekä olemassa olevien automaattisten tehtävien tarkastusjärjestelmien toimintaan. Osana diplomityötä, kirjallisuuskatsauksen perusteella, toteutettiin toimiva automaattisen tehtävien tarkastusjärjestelmän prototyyppi, jota testattiin 102 opiskelijan kurssilla. Kirjallisuuskatsauksessa jäsennellyn tiedon valossa sekä käyttökokemusten ja kyselytutkimuksessa kerätyn palautteen perusteella on mahdollista suunnata järjestelmän jatkokehitystä.

Suoritettu kirjallisuuskatsaus antaa yleisellä tasolla perspektiiviä CAD-opetuksen ja automaattisten tehtävien tarkastusjärjestelmien toimintaan. Kirjallisuuskatsaus tarjoaa myös katsaukseen niihin menetelmiin, joiden perusteella CAD-ohjelmistoista riippumia kolmiulotteisten mallinnustehtävien tarkastusmenetelmiä voisi lähteä kehittämään.

Toteutetulla prototyypillä suoritetun kokeilun perusteella CAD-mallinnustehtävien automaattinen tehtävien tarkastus on teknisesti mahdollista. Jo yksinkertaisilla tarkastuksilla, kuten mallien tilavuuden ja pinta-alan vertailulla, on mahdollista varmistua siitä, että mallit ovat oikein rakennettuja. Kehitettyä mallinnustehtävien tarkastustyökalua voisi käyttää kokonaisten kurssien tehtävien tarkastamiseen. Automaattista tarkastusta varten tulisi kurssien tehtäväkierrokset ja tehtävänannot suunnitella automaattisen tarkastuksen ominaisuuksia ja rajoituksia silmälläpitäen.

Koska automaattisia tehtävien tarkastusjärjestelmiä on ohjelmoinnin opetuksessa kehitetty jo pitempään, olisi muille aloille niitä kehitettäessä mahdollista välttyä päällekkäisen työn tekemiseltä uudelleenkäyttämällä jo olemassa olevia osakokonaisuuksia muista järjestelmistä tai toteuttamalla CAD-malleja tarkastava järjestelmä osaksi valmista järjestelmää. Jatkokehittämällä prototyyppiä saadun palautteen perusteella on siitä mahdollista saada käyttökelpoinen työkalu konetekniikan parametrisen CAD-mallinnuksen peruskurssien tehtävien automaattiseen tarkastukseen.

Viitteet

[1] Ian Sommerville. *Software Engineering: (Update) (8th Edition) (International Computer Science)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2006.

[2] E. Dimas and D. Briassoulis. 3d geometric modelling based on nurbs: a review. *Advances in Engineering Software*, 30(9):741–751, 1999.

[3] Xiaobo Peng, Prentiss McGary, Michael Johnson, Bugrahan Yalvac, and Elif Ozturk. Assessing novice cad model creation and alteration. 2012.

[4] David A. Field. Education and training for cad in the auto industry. *Computer-Aided Design*, 36(14):1431–1437, 2004.

[5] Ioannis Paliokas. Reinforcing metacognition in cad education using videotutorials. *Computer-Aided Design and Applications*, 6(5):613–623, 2009.

[6] C Werner Dankwort, Roland Weidlich, Birgit Guenther, and Joerg E Blaurock. Engineers' cax education – it's not only cad. *Computer-Aided Design*, 36(14):1439–1450, 2004.

[7] Kaur Jaakma ja Panu Kiviluoma. Keskusteluja diplomityön ohjaajien Kaur Jaakman ja Panu Kiviluoman kanssa.

[8] Tampereen teknillinen yliopisto. Opinto-opas 2016-2017, Konetekniikan DI-tutkinto-ohjelma. 2017.

[9] Aalto-yliopisto. Opinto-opas 2017-2018, kone- ja rakennustekniikka. 2017.

[10] Lappeenrannan teknillinen yliopisto. Opinto-opas, konetekniikka. 2015.

[11] Ilari Laine, Yliopisto-opettaja, Kone- ja tuotantotekniikka, Tampereen teknillinen yliopisto. Sähköpostikeskustelu, 2017.

[12] Oulun yliopisto. Opinto-opas, konetekniikka. 2015.

[13] Tapio Korpela, Yliopisto-opettaja, Konetekniikan ala, Oulun yliopisto. Sähköpostikeskustelu, 2017.

[14] Anna Eckerdal, Päivi Kinnunen, Neena Thota, Aletta Nylén, Judy Sheard, and Lauri Malmi. Teaching and learning with moocs: computing academics' perspectives and engagement. In *Proceedings of the 2014 conference on Innovation & technology in computer science education*, pages 9–14. ACM, 2014.

[15] Paul E. Anderson, Thomas Nash, and Renée McCauley. Facilitating programming success in data science courses through gamified scaffolding and learn2mine. In *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '15, pages 99–104, New York, NY, USA, 2015. ACM.

[16] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Robotics-DL tentative*, pages 586–606. International Society for Optics and Photonics, 1992.

[17] Kenton McHenry and Peter Bajcsy. An overview of 3d data content, file formats and viewers. 2008.

[18] Laakko, T. *Tuotteen 3D-CAD-suunnittelu*. WSOY, Helsinki, 1998. ISBN 951-0-23217-3.

[19] Tuomas Tiainen. Kandidaatintyö: Pintamallinnus. 2015.

[20] STEP Tools. Step schemas: Core data for automotive mechanical design processes. Viitattu 20.10.2015, saatavissa: `http://www.steptools.com/support/stdev_docs/express/ap214/html/index.html`, 2009.

[21] H. Blum. A transformation for extracting new descriptors of shape. In Weinant Wathen Dunn, editor, *Models for the Perception of Speech and Visual Form*, pages 362–381. MIT Press, 1967.

[22] Benjamin Bustos, Daniel A. Keim, Dietmar Saupe, Tobias Schreck, and Dejan V. Vranić. Feature-based similarity search in 3d object databases. *ACM Comput. Surv.*, 37(4):345–387, December 2005.

[23] Martin Reuter, Franz-Erich Wolter, and Niklas Peinecke. Laplace-spectra as fingerprints for shape matching. In *Proceedings of the 2005 ACM Symposium on Solid and Physical Modeling*, SPM '05, pages 101–106, New York, NY, USA, 2005. ACM.

[24] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. *Computer Graphics Forum*, 22(3):223–232, 2003.

[25] Pál Benkő, Ralph R Martin, and Tamás Várady. Algorithms for reverse engineering boundary representation models. *Computer-Aided Design*, 33(11):839–851, 2001.

[26] Andrea Sanna, Fabrizio Lamberti, Gianluca Paravati, and Claudio Demartini. Automatic assessment of 3d modeling exams. *IEEE Transactions on Learning Technologies*, 5(1):2–10.

[27] Michael Guerci and Douglas Baxter. Automating an introductory computer aided design course to improve student evaluation. In *2003 Annual Conference*, Nashville, Tennessee, June 2003. ASEE Conferences.

[28] Ville Karavirta, Petri Ihantola, and Teemu Koskinen. Service-oriented approach to improve interoperability of e-learning systems. In *Advanced Learning Technologies (ICALT), 2013 IEEE 13th International Conference on*, pages 341–345. IEEE, 2013.

[29] Petri Ihantola, Tuukka Ahoniemi, Ville Karavirta, and Otto Seppälä. Review of recent systems for automatic assessment of programming assignments. In *Proceedings of the 10th Koli Calling International Conference on Computing Education Research*, pages 86–93. ACM, 2010.

[30] Otto Seppälä, Yliopisto-opettaja, Tietotekniikan laitos, Aalto-yliopisto. Sähköpostikeskustelu, 2017.

[31] Arto Vihavainen, Thomas Vikberg, Matti Luukkainen, and Martin Pärtel. Scaffolding students' learning using test my code. In *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '13, pages 117–122, New York, NY, USA, 2013. ACM.

[32] Dassault Systemes. Solidworks 2016 help. 2016.

[33] PTC. Creo 3.0 API wizard. 2017.

[34] Keith Rice. Makroesimerkki. 2012.

[35] Siemens PLM Software. Solid edge api reference. 2017.

[36] Tim Bray. The JavaScript Object Notation (JSON) Data Interchange Format. RFC 7159, March 2014.

[37] Celery project. *Celery documentation*. 2017.

Liitteet

Liite A

CreoWrapper-luokan lähdekoodi

```
# -*- coding: utf-8 -*-


from io import StringIO
from win32com.client.dynamic import Dispatch
from win32com.client.gencache import EnsureDispatch
import win32com
from time import sleep
import traceback
from win32com.client import makepy

def isclose(a, b, rel_tol=0.01, abs_tol=0.0):
    return abs(a-b) <= max(rel_tol * max(abs(a), abs(b)), abs_tol)

def compare_res(correct, checked):
    for corr, check in zip(correct, checked):
        if not isclose(corr, check):
            return False
    return True


class CreoWrapperError(Exception):
    def __init__(self, message):
        super(CreoWrapperError, self).__init__(message)

class PythonCreoConnection():
    def __enter__(self):
        print("Trying to form connection")
        self.models = []
        self.asyncconn = Dispatch('pfcls.pfcAsyncConnection')
        self.conn = self.asyncconn.Connect(None, None, None, None)
        self.session = self.conn.Session
        print("Connection formed")

        return self

    def open_file(self, path):
        options = Dispatch('pfcls.pfcRetrieveModelOptions')
        o = options.Create()
        file = Dispatch('pfcls.pfcModelDescriptor')

        f = file.CreateFromFilename(path)
        self.models.append(self.session.RetrieveModelWithOpts(f, o))
        self.session.OpenFile(f)
```

```
    def activate_window(self, model_id):
        self.window = self.session.GetModelWindow(self.models[model_id
            ])
        self.window.Activate()

    def close_window(self):
        self.window.Close()

    def update_massprops(self):
        print("Running macro")
        macro = "~ Activate `main_dlg_cur` `page_Model_control_btn` 1;~
            Command `ProCmdMmParams` ;~ Activate `relation_dlg` `PB_OK
            `;~ Command `ProCmdRegenPart` ;~ Activate `mdlprops_dlg` `
            MASS_PROPS_lay_Controls.push_Info.lay_instance`;~ Close `
            mp_sa_browser_report` `mp_sa_browser_report`;~ Activate `
            mdlprops_dlg` `DlgClose_push`;"
        self.session.RunMacro(macro)

    def set_parameter(self, mdl, param_name, value):
        param = mdl.GetParam(param_name)

        try:
            paramvalue = param.value
        except AttributeError:
            raise CreoWrapperError("Parameter {} not found".format(
                param_name))


        modelitem = Dispatch('pfcls.MpfcModelItem')
        #create boolean if param is not float
        if isinstance(value, bool):
            val = modelitem.CreateBoolParamValue(value)
        elif isinstance(value, (float, int)):
            val = modelitem.CreateDoubleParamValue(value)
        else:
            raise CreoWrapperError("Invalid value type")



        param.SetScaledValue(val, None)
        #for i in range(0, mdl.ListParams().Count):
        #    print(mdl.ListParams().Item(i).Name)
        #param.SetScaledValue()

        #print(param.GetScaledValue().DoubleValue)


        try:
            mdl.Regenerate(None)
            #self.update_massprops()
            self.window.Repaint()
        except:
            raise CreoWrapperError("Model failed to regenerate with {}
```

```
                = {}".format(param_name, value))

    def __exit__(self, exc_type, exc_value, traceback):
        try:
            self.window.Close()
        except:
            pass

        self.conn.Disconnect(2)
        print("Connection closed")
```

Liite B

Kyselytutkimuksen alkuperäiset kysymykset

Background
What is your experience in using parametric CAD programs? (1-5)
Have you used Creo Parametric before the course? (YES/NO)
Have you completed the Lego exercise before? (YES/NO)
Do you have previous experience with automatic exercise assessment systems? (YES/NO)
Describe the automatic assessment systems you have used before. (Freetext)
I found the automatic assessment system useful. (1-5)
I found the automatic assessment system supportive. (1-5)

User experience
Which internet browser did you use? (Freetext)
The exercise instructions were clear. (1-5)
The instructions for the automatic assessment system were clear. (1-5)
How many submissions was required to pass the exercise? (1-5 or more)
I like the possibility to submit exercises time-independently. (1-5)
Where did you submit the exercise (home/university/other).
I like the possibility to submit exercises location-independently. (1-5)
Using the automatic assessment system was easy. (1-5)
I needed assistance from course staff to complete the exercise. (1-5)
I needed assistance from course staff with the automatic assessment system. (1-5)
The response time of the automatic assessment system was fast enough. (1-5).
Feedback and comments about user experience. (Freetext)

Learning experience
The automatic assessment system supported my learning. (1-5)
The feedback provided by the automatic assessment system is sufficient. (1-5)
I understood the feedback provided by the automatic assessment system. (1-5)
I trust that the automatic assessment system assessed my work correctly. (1-5)
Using the automatic assessment system motivated me. (1-5)
Using the automatic assessment system was exciting. (1-5)
I prefer using automatic assessment systems compared to submitting the work to course staff. (1-5)
Feedback and comments about learning experience. (Freetext)