TALLINNA TEHNIKAÜLIKOOL
INSENERITEADUSKOND
Ehituse ja arhitektuuri instituut

TOOTEMUDELITE JA -JOONISTE PROJEKTEERIMISTÖÖ AUTOMATISEERIMINE SKRIPTIMISE ABIL

AUTOMATION OF PRODUCT FABRICATION MODELS AND DRAWINGS DESIGN BY SCRIPT

MAGISTRITÖÖ

Üliõpilane: Albert Notberg

Üliõpilaskood 122579

Juhendaja: Ergo Pikas, abiprofessor

Tallinn 2022

AUTORIDEKLARATSIOON

Olen koostanud lõputöö iseseisvalt.
Lõputöö alusel ei ole varem kutse- või teaduskraadi või inseneridiplomit taotletud.
Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

16. mai 2022

Autor: .....
/ allkiri /

Töö vastab magistritööle esitatud nõuetele.

".....” ..... 20…..

Juhendaja: .....
/ allkiri /

Kaitsmisele lubatud

".....”. .....20… .

Kaitsmiskomisjoni esimees:

.....
/ nimi ja allkiri /

LIHTLITSENTS LÕPUTÖÖ REPRODUTSEERIMISEKS JA LÕPUTÖÖ ÜLDSUSELE KÄTTESAADAVAKS TEGEMISEKS

Mina, Albert Notberg,

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose

Tootemudelite ja -jooniste projekteerimistöö automatiseerimine skriptimise abil

mille juhendaja on Ergo Pikas

1.1 reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

1.2 üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.

3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

16.05.2022

Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. jq 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.

Ehituse ja arhitektuuri instituut

LÕPUTÖÖ ÜLESANNE

Üliõpilane: **ALBERT NOTBERG** Üliõpilaskood **122579**

Õppekava: **EAEI02 Ehitiste projekteerimine ja ehitusjuhtimine**
Peaeriala: Ehitusmajandus ja juhtimine

Lõputöö teema:

TOOTEMUDELITE JA -JOONISTE PROJEKTEERIMISTÖÖ AUTOMATISEERIMINE SKRIPTIMISE ABIL

Automation of product fabrication models and drawings design by script

Juhendaja: **Ergo Pikas** Ergo.pikas@taltech.ee

Lõputöö konsultandid:

Tiitel või ametikoht, Ees- ja Perekonnanimi	Kontakt (e-post või telefon)	Allkiri ja kuupäev

Lõputöö põhieesmärgid:

1. Hinnata parimaid teadmisi ja praktikaid BIM ja automatiseerimisest tootmise täpsusega mudelite ja jooniste loomisel.
2. Välja töötada raamistik ja protsessi mudel tehase tarvis tootmismudelite ja -jooniste koostamise automatiseerimiseks.
3. Välja töötada praktilised skriptid ja testida nende tõhusust ettevõtte kontekstis.

Töö keel: eesti keel

Lõputöö etapid ja ajakava:

	Ülesande kirjeldus	Tähtaeg
1.	Kaaristada parim teadmine ja praktika tootmistäpsusega mudelite ja jooniste automatiseerimisel	17.02.2022
2.	Uurida ja hinnata Saksamaa raudbetoonelementide tootmise ja tehaste eripärasid	03.03.2022
3.	Anda ülevaade BIM-i rollist ja võimalustest tootmiseks vajaliku täpsusega mudelite ja jooniste loomisel	10.03.2022
4.	Anda ülevaade skriptimese keeletest ja võimalustest	17.03.2022
5.	Kirjutada tootemudelite ja -jooniste abiprogramme	24.03.2022
6.	Rakendada abiprogramme praktikas ja hinnata kasu	31.03.2022
7.		
	Kokkuvõte eesti keeles	14.04.2022
	Kokkuvõte inglise keeles	14.04.2022
		….
		….
		….
		….
	Lõputööde 95% ülevaatus, mille läbimine on kaitsmise eelduseks	**04.05.2022**

Lõputöö esitamise tähtaeg: 23. mai 2021

Lõputöö ülesanne välja antud: 13.02.2021

Juhendaja: Ergo Pikas .....

Ülesande vastu võtnud: .....

Avalikustamise piirangu tingimused: puuduvad

SISUKORD

EESSÕNA

Lõputöö eesmärk on optimeerida ja automatiseerida tööd BIM tarkvaras skriptimise abil. Juurdepääsu programmile Strakon ja vajalikud andmed projektide kohta pakus Keskkonnaprojekt OÜ projekteerimisbüroo.

Töö käigus oli tehtud ülevaade Saksa tehaste automatiseerimisprotsessist, BIM programmidest ja skriptimiskeeltest. Töö käigus oli läbi viidud Strakon BIM tarkvara analüüs ja olid pakutud lahendused töö automatiseerimiseks ja optimeerimiseks skriptimise abil. Töö tulemusena õnnestus luua 137 kasulikke skripte Autohotkey abil ja rakendada neid praktikas, mille tõttu oli vähendatud ajakulu rõdu elementidele 11% võrra ja trepi elementidele 14% võrra.

Suur tänu KESKKONNAPROJEKT OÜ ettevõtte meeskonnale, kes aitasid välja selgitada programmi puudused ja osalesid skriptide testimisel.

Suur tänu lõputöö juhendajale Ergo Pikas'le lõputöö hea koordineerimise ja kontrollimise eest.

Võtmesõnad: Automatiseerimine, BIM, skript, Strakon, magistritöö

SISSEJUHATUS

Ehitusmajanduse valdkonnas on aastaid olnud trend ehitusmaterjalide ja üldiselt ehituse hinnatõusule. Pidades seda silmas püüvad ehitusettevõtted vähendada ressursside kulutamist igal võimalikul viisil. Kui väikeprojektide puhul pole rahalises mõttes ressursside kokkuhoid nii oluline, siis suurprojektide puhul on olukord teistsugune. Arhitektuursete, konstruktiivsete ja tehnosüsteemide jooniste koostamisele kuulub palju aega ning tihti neis esinevad vastuolud, kuna need tavaliselt on koostatud erinevate büroode poolt. Suurprojektide elluviimisel on väga oluline õige lähenemine nende haldamisprotsessile. BIM tehnoloogia loob eeldused ehitusprojektide tõhusaks juhtimiseks, ehitusmaterjalide kokkuhoiuks ja ehitusaja vähendamiseks.

Samuti üheks ehitusaja vähendamise võimaluseks on monteeritavate raudbetoonelementide tehases valmistamine. Nende peamiseks eeliseks on võimalus kasutada neid tooteid ehituse industrialiseerimiseks, kuna valmistamine toimub tehases. Lisaks pakub see võimaluse tõsta ehitustööde kvaliteeti ja vähendada tööjõukulusid. Need on transporditavad elemendid, ilma milleta on raske ette kujutada tänapäevast ehitust.

BIM modelleerimistehnoloogia kasutamine suure arvu monteeritavate elementidega projektide haldamiseks on tõhus, kuid kahjuks võtab aega ka modelleerimine ja jooniste koostamine tehaste jaoks. Enamasti osutub projekteerimine, täpsemalt modelleerimine ja joonestamine tootmisprotsessi pudelikaelaks.

Sellest tulenevalt lõputöö eesmärgiks oli seatud BIM-i automatiseerimise parimaid teadmisi ja praktikaid tootmise täpsusega mudelite ja jooniste koostamiseks ning välja töötada raamistik ja protsessi mudel tehase tootmismudelite ja -jooniste koostamise automatiseerimiseks. Samuti eesmärgiks oli välja töötada praktilised skriptid ja katsetada nende tõhusust ettevõtte kontekstis.

Käesolev töö on suunatud projekteerimisbüroo Keskkonnaprojekt OÜ töö optimeerimisele.

Lõputöö eesmärk on optimeerida ja automatiseerida tööprotsessi BIM tarkvaraga skriptimise abil. Uuringu objektiks on Saksamaa turule monteeritavate raudbetoonelementide projekteerimise tehnoloogiline tsükkel. Uuringus on kirjeldatud töö optimeerimine Strakon programmi näitel Autohotkey programmeerimiskeele abil.

Selle eesmärgi saavutamiseks otsitakse vastuseid järgmistele küsimustele:

- Millised on Saksamaa raudbetoonelementide tootmise ja tehaste eripärad?
- Millised BIM modelleerimise programmid on tänapäeval kättesaadavad ja millised on võimalused tootmiseks vajaliku täpsusega mudelite ja jooniste loomise automatiseerimiseks?
- Millised aspektid takistavad tööd valitud instrumentiga?
- Kuidas saab BIM tööriista automatiseerida ja milline on selle kasu?

Antud töö on jagatud mitmeks etapiks. Kõigepealt käsitletakse kirjanduse ülevaates Saksa tehaste eripärasid ja selgitatakse millised töövahendid vastavad tehaste nõuetele. Järgmisena selgitatakse välja, kuidas oleks võimalik nende vahendite abil tööd automatiseerida kasutades skripte ja abiprogramme. Edasi luuakse abiprogrammide süsteemi ülesehitus ning püstitatakse konkreetsed abiprogrammi eesmärgid. Lõpuks arendatakse skriptid ja rakendatakse praktikas, et uurida nendest saadavat kasu.

1. KIRJANDUSE ÜLEVAADE

Probleemide ja ülesannete täpsustamiseks on vaja määratleda tööde kontekst. Selles peatükis vaadeldakse Saksa tehaste automatiseerimise põhiprintsiipe ja nende nõudeid failivormingutele, mis tarnitakse tehasele koos joonistega. Peatükis vaadeldakse ka mõnesid BIM-tööriistu, mida Saksamaa turul sageli kasutatakse projekteerimiseks. Samuti peatükis käsitleti BIM-tööriistades kasutatavaid skriptimiskeele, mida saab kasutada protsessi automatiseerimiseks.

1.1 Saksamaa raudbetoonelementide tootmispraktikad

Järgnevas lõigus käsitletakse Saksa tehaste eripärasid ja monteeritavate raudbetoonelementide tootmise industrialiseerimise põhimõtteid. Artiklis kirjeldatakse ka failivorminguid, mida tehastes vajatakse tootmise läbiviimiseks ja automatiseerimiseks.

1.1.1 Saksamaa tehaste tootmisspetsiifika

Saksa tehaste tunnusteks on tootmise kiirus ja ulatus. Kõik see sai võimalikuks tänu monteeritavate raudbetoonelementide tootmise automatiseeritud protsessi kasutuselevõtule.

Riskide vähendamine on sakslaste prioriteet. Suurtes tehastes projekteerimisjooniseid kontrollitakse mitu korda. Esimest korda kontrollitakse modelleerimise ja joonestamise etapis, pärast seda kontrollitakse jooniseid tehases, seejärel viiakse läbi väline konstruktorite kontroll, kus elementi dokumentaalselt kinnitatakse. Lõplik kooskõlastus toimub paigalduse käigus kohapeal. Samuti element on arhitekti poolt kontrollitud ja dokumenteeritud.

Sisseehitatud osad ostetakse spetsialiseeritud ettevõtetelt, kellel on vastav garantii ja kohustuslik tehniline dokumentatsioon. Näiteks välditakse juhtumeid, kus tõsteaasadena kasutatakse painutatud armatuuri.

Standardsete sisseehitatud osade kasutamine koos minimaalse tugevdusjuhendiga vähendab staatiliste arvutuste mahtu. See oluliselt lihtsustab tööd elemendiga nii modelleerimise kui ka arvutuste ja kontrollide etapis.

Saksa tehased püüavad kasutada uusimaid tehnoloogiaid, eriti kui need muudavad lihtsamaks elemendi kokkupanekut. Üks hea näide on seotud rõdu ja vahelae sõlme

tehnilise lahendusega. Keeviskonstruktsioone ei kasutata töötava osana rõdude ja lodžade toetamiseks, peamiselt kasutatakse Schöck Isokorb ja Halfen Insulated Connection [1][2]. See osa koosneb komposiitsarrusest ja isolatsioonis olevatest surveplokkidest. Komposiitvardad kannavad üle tõmbekoormust ning isolatsioonkihis olemasolev roostevaba teras kaitseb vardad korrodeerumisest. Surveplokid kannavad üle survekoormust. See on hea lahendus, kuna oluliselt vähendab külmasildu ja seda on lihtne kasutada. Selle osa jaoks on olemas minimaalne lubatud lisaarmatuur, mis on juhendis ette nähtud. Selle osa tootja on oma toodetele välja töötanud ka tasuta arvutustarkvara, mis võimaldab vähendada kooste arvutamise ajakulusid.

Teine hea näide on helisummutavate tihendite kasutamine, millel on treppide tugede jaoks määratletud minimaalne geomeetria. Konsoolide tugevdamiseks kasutatakse kõige sagedamini nende tihendite tootjate käsiraamatuid, kuna need määravad konsooli iga tööpiirkonna jaoks minimaalset armeerimist [3]. Antud detail võimaldab vähendada staatika arvutusi.

1.1.2 Tooteprotsesside automatiseerimine

Töös käsitletakse monteeritavate elementide konveieritootmist ühes Saksamaa tehases, mille nimeks on Thomas Prefab [3].

Automatiseerimine betoonitehastes toimub erinevates etappides. Esimene on raketise kokkupanek. Raketise kokkupaneku võib jagada kaheks tüübiks:

- tehase poolt standarditud raketis;
- mittestandardne raketis.

Tehases standardne raketis seab kliendile raamistikku vastavalt elemendi maksimaalsetele mõõtudele ja kujudele. Seda tehakse selleks, et võimalikult palju elemente saaks toota minimaalse raketise materjali ja ajakuluga. Selliste elementide hulka kuuluvad näiteks monteeritavad raudbetoonseinad. Elemendi korduv kuju võimaldab lühendada seadmete ümberkonfigureerimise aega ja vähendada aega laua ettevalmistamise etapis.

Mittestandardset raketist kasutatakse näiteks keerdtreppide valmistamiseks. Nende elementide automatiseerimine toimub valamiskasti kokkupaneku etapis. Puidust kasti kokkupanemiseks tuleb läbi lugeda kogu elemendi geomeetria, jagada vineeri vormideks ning seejärel lõigata soovitud kujunditeks. See on tüüpiliselt keeruline ja pikk protsess. Peamine aspekt on ka materjali vormideks jaotamine selliselt, et minimaalse kärpimisega materjali pealt võimalikult palju kokku hoida. Kuna kogu info

karbi geomeetria kohta inimesed loevad joonise järgi, on inimfaktori risk suur ning sageli kulub keerukate elementide raketise kogumisele oodatust rohkem ressurssi.

Selle probleemi lahenduseks on automatiseeritud masinad, mis loevad faili ja lõikavad raketise komponente säästlikult ja suure täpsusega. Nende masinate sisendiks on tarvis XML-vormingus faili. XML-fail on laiendatav märgistuskeele formaat ja seda kasutatakse andmete organiseerimiseks, talletamiseks ja kommunikeerimiseks [4]. Need failid sisaldavad teavet koordinaatpunktide kohta, mida mööda vineeri edasi lõigatakse.

Lisaks võtavad Saksa tehased üha enam kasutusele LAP (Laser template projection) tehnoloogiat. See lasermasin projitseerib elemendi joonise lauale ning sobivateks sisenditeks on IGES-, DXF-, HPGL-, LPD-, PLY-vormingus failid. Selline meetod säästab aega ja parandab raketise paigaldamise ja sisseehitatud osade paigaldamise täpsust.[5]

Järgmine automatiseerimise samm on ettevalmistatud vardad ja võrgud. Klassikalistes tehastes painutatakse armatuur painutuspinkidel käsitsi. Töötaja saab nimekirja soovitud varrastest koos nende kujuga ja annab need siis edasi raketisse ladumiseks.

Võrkude keevitamine ja painutamine toimub ka käsitsi. See sõltub, millisel kujul on armatuur. Võrgusilma keevitamine tehases on äärmiselt haruldane, kuna see on pikk ja väga kulukas protsess. Teise variandi puhul tellitakse võrgu trumlis, seejärel võrk on lahti painutatud ja kohapeal lõigatud. Protsess on eelmisest odavam, kuid materjali tuleb sirgeks tõmmata, mis omakorda võtab palju aega. Puuduseks on ka lõikamise protsessis tekitavad võrgujäägid, mida tuleb utiliseerida. Kolmas võimalus on tellida sirged võrgud. Neid tarnitakse standardsuuruses ja neid ei ole vaja lahti voltida. Sirgete võrkude puuduseks on kõrge hind ning veel suurem jääkide hulk pärast lõikamist.

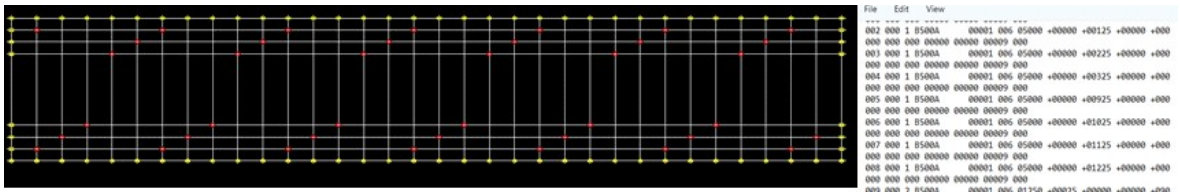
Terase säästmiseks ja protsessi automatiseerimiseks kasutavad Saksa tehased sarrusevardaid kompaktsetes trumlites (Joonis 1.2.1). Masin automaatselt lõikab varda ja painutab soovitud kujule.

Võrkude puhul ajab masin sirgeks ja lõikab vajaliku arvu vardaid. Pärast seda vardad automaatselt laotakse õige sammuga ja keevitatakse aparaadi poolt õigetes kohtades ilma inimese sekkumiseta. Protsessi lõpus transporditakse võrgud suurte magnetite abil soovitud elemendini.

Joonis 1.2.1 Sarrus trumlis, võrgu paigutamine keevitamiseks, võrgu paigutamine raketise magneti abil.

Vajadusel painutab masin ka võrgu soovitud kujule. Painutatud võrku sageli kasutatakse tehastes plaatide vaba serva tugevdamiseks. See võimaldab lühendada U-varraste paigutuse aega.

Masina töötamiseks on üksikvarraste jaoks vaja BVBS-vormingus faili. Disainvõrkude jaoks on vajalik SWP-vorming. BVB - on armatuuride andmekeelvorming, mis sisaldab pikkuste ja soovitud paindenurkade loendeid.[6] SWP - vorming sisaldab teavet iga varda ruumilise paigutuse kohta ruudustikus ja selle keevituskohad (Joonis 1.2.2).



Joonis 1.2.2 SWP fail genereerimine Strakon programmis.

Raudbetoonelementide tootmise automatiseerimise viimaseks etapiks on armatuuri paigaldamine ja elemendi valamine raketisse (Joonis 1.2.3). Valmis armatuurvõrgud laovad robotid lauale vajalikele koordinaatidele, seejärel valatakse elemendid vibrolaual ja transporditakse ruumidesse tahkumiseks.

Joonis 1.2.3 Elemendi valamine vibrolaual.

Tarkvaranõuded põhinevad betoonkonstruktsioonide automatiseeritud tootmisel. Selleks, et masinad saaksid lugeda teavet ja iseseisvalt töötada, on vajalikud teatud failivormingud:

- XML - formaat raketist lõikava masina jaoks;
- BVB - formaat armatuuri painutava ja lõikava masina jaoks;
- SWP - võrke keevitava masina formaat;
- DXF - formaat lasermasina jaoks.

Jooniste koostamiseks ja kontrollimiseks on vaja peamiselt PDF-vormingut. Andmebaasi salvestamiseks kasutatakse PDF- ja DWG-vorminguid. Konstruktori arvutuste jaoks on vaja PDF-, DWG-, IFC- ja mõnikord CON (Strakon mudeli)-vorminguid.

1.2 BIM ehituses

Selles jaotises käsitletakse BIM-i kontseptsiooni ja mõningaid selle tööriistu. Saksa tehastes eelistatakse BIM-programmidest kasutada Eestis vähetuntud modelleerimistööriista Strakonit. Järgmisena antakse ülevaadet Strakon programmist ja kolmest Eestis populaarsest BIM-tööriistast (Revit,Tekla Structures, AutoCad), ning nende automatiseerimise funktsioonidest.

1.2.1 Sissejuhatus BIM-i

BIM on akronüüm sõnadest Building Information Modeling, Building Information Management või Building Information Model. See on ühine keel, mis soodustab ja võimaldab arhitektidel, inseneridel, arendajatel, töövõtjatel, tootjatel ja teistel ehitusspetsialistidel kavandada, projekteerida ja ehitada ehitisi kaks korda. Esimene kord digitaalselt ja teine kord ehitusplatsil. Targal mudelil põhinev ja pilveplatvormi poolt toetatud BIM koondab struktureeritud interdistsiplinaarseid andmeid, et luua vara digitaalset esitust kogu selle elutsüklile alates planeerimisest ja projekteerimisest kuni ehitamise ja käitamiseni. [7] [8]

Sisuliselt on BIM mudel paljude erinevate elementide mudelite kogum, mis omakorda sisaldavad teavet (materjal, omadused, asukoht jne). Neid andmeid saab kasutada täpsuse parandamiseks, projekteerimiskavatsuste edastamiseks kontorist objektile, muudatuste tellimuste ja kohapealse kooskõlastamise probleemide vähendamiseks. Selle planeerimiskäsitluse põhieesmärk on ehitada hoonest digitaalsel kujul kolmemõõtmeline virtuaalne mudel ja kanda täielikku teavet tulevase objekti kohta.

BIM-tehnoloogia kasutamine hoonete projekteerimisel hõlmab hoone tehnoloogilise, arhitektuurse, disaini- ja majandusalase informatsiooni kogumist ja kompleksset töötlemist nii, et ehitusplatsi ja sellega seonduvat teavet käsitletakse tervikuna. 3D-hoonemudel on tihedalt seotud teabeandmebaasiga, nii et isegi ühe ehitusobjekti parameetri muutmine muudab ka kõiki seotud süsteeme ja objekte, sealhulgas jooniseid, spetsifikatsioone, visualiseerimisi ja ajakava. [9] [10]

BIM eelised:

- projekteerimisaja vähendamine;
- projekti elluviimise kulude vähendamine;
- suurenenud tootlikkus tänu kvaliteetsemale teabele;
- ehitusdokumentatsiooni järjepidevuse parandamine;
- konkreetse teabe kättesaadavus materjalide tootjate ja kvantitatiivsete näitajate kohta hindamiseks ja pakkumiseks.

BIM-tehnoloogia tulek on põhjalikult muutnud hoonete ja rajatiste projekteerimise protsessi, võimaldades luua infomudelil põhineva ühtset andmebaasi. See on aidanud parendada ehituse tõhusust, hõlbustades projektis osalejate vahelist suhtlust, vähendades vastuolusid ning tõhustades koostööd. [11]

Edusammud ei seisa paigal. Arengu edenedes leiutatakse töö hõlbustamiseks uusi arendusi. Suurte projektide ilmumisega kulus modelleerimisele ka suur ajakulu. Aja säästmiseks on selle protsessi automatiseerimiseks välja töötatud erinevaid tööriistu.

Tarkvara automatiseerimise tööriistade kasutamine on aga protsess, mis nõuab teatud oskusi, eelkõige programmeerimisoskusi, mis ehitusvaldkonna projekteerijatel sageli puuduvad, mis tekitab raskusi automatiseerimistehnoloogia ja -tööriistade valdamisel. Seda mõistes on tarkvaraarendajad integreerinud BIM-programmidesse visuaalse programmeerimiskeele ehk skriptitud programmeerimise. See võimaldab inseneridel, kellel puuduvad programmeerimise oskused, luua algoritme ülesannete lahendamiseks ja automatiseerimiseks: ühendatud sõlmedena tõlgendatud skriptide jadasid, mis sisaldavad kindlat käskude komplekti. Need komplektid võimaldavad automatiseerida tööd hostprogrammi tööriistade või alla laaditavate tööriistadega. [12]

BIM-programmi töö automatiseerimiseks on praegu mitu võimalust, mis on loodud ülesande erineva keerukuse ja kasutajate koolituse jaoks. Need meetodid hõlmavad järgmist [12]:

- programmeerimiskeeles koodi kirjutamine eraldi programmis koos hilisema laadimisega BIM programmi.
- algoritmi loomine visuaalses programmeerimiskeeles BIM programmi sees sõlmede abil.

1.2.2 Revit

Revit on tuntud ja populaarne BIM-platvorm, mille Autodesk tutvustas 2002. aastal pärast seda, kui Autodesk ostis idufirmalt Reviti tarkvara [11]. Revit on Autodeski peamine parameetriline modelleerimistööriist hoonete projekteerimiseks. See koosneb projekteerimistööriistadest hoone modelleerimiseks 3D-keskkonnas koos automaatselt lingitud 2D-vaadete ja andmeesitustega mitteruumiliste andmete (nt kulu) jaoks. Sellel on kolm töövoogu, mis toetavad projekteerimisprotsessi erinevaid osi. Need on arhitektuur, konstruktsioon ja tehnosüsteemid.[9]

Revitil on lihtne liides, kus on palju vihjeid ja nutikas kursor, mis kohandub aktiivsete käskudega. Selle menüüd on töövoo järgi hästi korraldatud. Selle kavandi genereerimise tugi on väga hea. Selle kavandi genereerimine on väga assotsiatiivne, nii et kavandi väljundit on lihtne hallata. See pakub kahesuunalist redigeerimist joonistest mudeliteni ja tagasi, samuti kahesuunalist redigeerimist uste, ukse riistvara jne spetsifikatsioonides. [11]

Revit võimaldab eksportida XML, BVB, SWP, DXF, DWG, PDF ja IFC failivormingud. Seega vastab Revit kirjeldatud punktis 1.1.2 saksa tehaste failivormingute nõuetele.

REVITi peamised töövoo automatiseerimise tööriistad pakuvad Dynamo visuaalset programmeerimiskeelt. Samuti on sellel tarkvarapaketil rakenduste programmeerimisliides Revit API, tänu millele on võimalik integreerida pluginaid, mis on loodud mistahes .NET platvormi toetavas programmeerimiskeeles nagu VB.NET, C #, Python jt. [12]

Revit API on lühend sõnadest Revit Application Programming Interface (API), rutiinid ja tööriistad, mis võimaldavad kasutajatel kohandada käskude funktsionaalsust. See on loodud selle eesmärgiga, et peegeldada sama kasutaja interaktsiooni paradigmasid kui programmi graafilist kasutajaliidest. Graafilises kasutajaliideses Reviti modelleerimisel kuvatakse objekt 2D või 3D mudelina; Revit API-s objekti väljendatakse tekstina.

Revit API oluline abifail on Revit SDK (tarkvaraarendaja komplekt). Revit SDK pakub Autodesk ja sisaldab näidiskoodi. Revit SDK selgitab ka iga töö süntaksit element Revitis. Iga elementi esindab klass Revit API-s.[13]

Lisaks pakub Reviti API iga eksemplari atribuute, nagu ID, mis on Revitis elemendi unikaalne identifikaator. Seetõttu saavad kasutajad ID abil leida Revitis konkreetset elementi ja seda muuta. Selline funktsioon võimaldab Revitis automaatset juhtimist.

Revit API töötab .NET-i raamistikus, mis on Microsofti välja töötatud tarkvararaamistik peamiselt Microsoft Windowsis ja pakub keele koostalitlusvõimet. On olemas erinevaid tüüpe Revit API-ga töötavaid programmeerimiskeeli, nagu C#, VB.NET, Ruby ja Python. [14]

Nende rakenduste jaoks kõige laialdasemalt kasutatav programmeerimiskeel on C#. Enamik SDK näidistest ja veebinäiteid on C# keeles.

Lisandmoodul või pistikprogramm on tarkvarakomponent, mis lisab olemasolevale arvutiprogrammile kindla funktsiooni. Kui luua auditi lisandmoodul või mistahes .NET-i rakendus, kompilaator teisendab programmi koodi ja kompileerib seda vastavalt vajadusele EXE- või DLL-failiks. Üks C# kompilaatoritest on Visual Studio. See võib sisaldada üht EXE-faili põhitarkvarapaketis ja sellega võib kaasneda üks või mitu DLL-i kogumit.

DLL-i kogum on tarkvarakogu, mis on kasutatavatele programmidele kättesaadav. See tähendab, et kui käivitada programmi (nt EXE) fail, saab programm laadida programmi kompileerimiseks täiendavaid DLL-e. Näiteks Revit.exe jaoks Plugin Manageri pistikprogramm paigutatakse eraldi DLL-i, mille Revit.exe laadib iga kord, kui see käivitub või kui kasutate seda pistikprogrammi. Iga DLL vastab Revit.exe funktsioonile. [15] [16]

Dynamo on visuaalse programmeerimistarkvara. See võimaldab kasutajatel töötada visuaalse programmeerimise protsessis, milles nad ühendavad elemente omavahel, et määratleda seoseid ja toimingute jadasid, mis moodustavad kohandatud algoritme. Algoritm realiseeritakse vajalikus järjekorras ühendatud sõlmede abil, mis tõlgendavad teatud arendajate poolt eelseadistatud käske. Need käsud võivad mudelisse sisse ehitada, teha arvutusi, analüüsida teavet, eraldada andmeid nii mudelist kui ka arvutis asuvatest failidest, samuti neid tagasi väljastada ja palju muud. Koos Revitiga aitab see disaineritel lahendada palju keerulisi ülesandeid, näiteks luua linna 3D-mudelit või ehitada parameetrilisi keerulisi kujundeid.[12] [16]

Teisisõnu pakub see meetod ulatuslikke võimalusi, millest piisab paljude ülesannete teostamiseks. Samal ajal meetod on kättesaadav enamikule kasutajatele, mis meelitab üha rohkem inimesi. Siiski väärib märkimist, et Dynamo sõlmedest küllastunud keeruka

algoritmi töö, eriti kui juurde pääseda failidele väljaspool programmi, on nii arvutile kui ka Reviti tarkvarapaketile suur koormus. Parim lahendus võiks olla see, et kasutada koodi pluginis.

Samuti sellel ei ole kõiki otseseid kodeerimisfunktsioone, näiteks on võimatu seada silmusstruktuure, kordades teatud väärtuste üle, ja standardsõlmede komplekt hõlmab ainult kõige tavalisemaid käske, millest mõnikord ei piisa. [12]

Dynamo pakub erinevaid sõlmesid erinevate funktsioonidega. Kuigi Dynamos on võimalik ehitada ka uus sõlm. [16] [17] See kasutab kodeerimist Pythoni programmeerimiskeeles, kood sisestatakse spetsiaalsetesse Dynamo sõlmedesse, mida nimetatakse "Python Script". Need skriptid võivad olla C#-skriptid, VB.NET-skriptid, (Iron)Python-skriptid ja Revit 2022-le eelnevate väljalasete puhul Ruby-skriptid. [21].

Dynamo kasutajate õnneks on Internetis olemas suur Dynamo jaoks mõeldud erinevate skriptide raamatukogu, mille tulemusena saab programmeerimisalaste teadmisteta kasutaja alla laadida teiste inimeste poolt loodud tööriistade paketi ja nende abil ülesande lahendada, täiendades standardsõlmesid.

1.2.3 Tekla Structures

Tekla Structures on praegu laialdaselt kasutatav konstruktsioonide projekteerimistarkvara. See on aluseks BIM-projekti kontseptsioonile ehituses. Selline lähenemine toob ehitusvaldkonnale suurt kasu ehitusprotsessile kõikides etappides. 3D-teabemudeli loomise protsess toob kaasa uue nõudluse selle protsessi optimeerimiseks. Õnneks kasutab ehitusvaldkond standardimise ja unifitseerimise põhimõtete tõttu piiratud arvu parameetrilisi osi, mida saab kasutada mudeli elementaarsete osadena. See muudab mudelite loomise protsessi palju kiiremaks, kuid nõuab selliste osade loomiseks ettevalmistavat tööd. See lähenemine on eriti rakendatav monteeritavate betoonelementidega projektide puhul. Mudeli monteeritav betoonelement koosneb peamiselt elemendi betooni geomeetriast, armatuurist ja koosteelementidest. Tekla Structures on hea funktsionaalsusega ja sobib erinevate mudelikujude loomiseks. Integreeritud komponendid katavad enamasti armatuuri modelleerimise vajadusi. Sisseehitatud osade jaoks kasutatakse seda sageli tehaste kavandatud valmistoodete jaoks. Need tehased pakuvad sageli oma TS-komponente, et kiiresti ja mugavalt sisestada need osad mudelisse. Kõikide automatiseerimise abitööriistade juurutamine toimub läbi Tekla OPEN API. [18]

Tekla võimaldab eksportida XML, BVB, SWP, DXF, DWG, PDF ja IFC failivormingud. Seega vastab Tekla kirjeldatud punktis 1.1.2 saksa tehaste failivormingute nõuetele.

Tekla Open API on liides, mis võimaldab kasutajatel suhelda 3D-mudelite ja joonistega, kasutades kolmandate osapoolte rakendusi. Teisisõnu võimaldab API välistel programmidel koguda teavet, töödelda seda kasutaja äranägemisel ja seejärel teha muudatusi mudelis ja joonistes. See avab tohutu potentsiaali selle tööriistaga töötamisele, sest Teklas pole info lugemisel ja töötlemisel piiranguid.

OPEN API struktuur sisaldab linke makrodele, salvestatud kohandatud atribuutidele, välistele programmidele ja pistikprogrammidele. Kõik need funktsioonid avavad selles programmis laias valikus töövoo automatiseerimise sätteid.[18] [19] [20]

Pistikprogramm on süsteemikomponent (DLL), mida saab käivitada komponentide kataloogist. Ta töötab Tekla Structures'is. Pluginad muudavad mudeleid, kasutades komponente, mis on mudeliobjektide rühmad. Neid on lihtne mudelisse sisestada ja tervikuna muuta. Võrreldes kohandatud komponentidega on komponentidel sisestusväljad objektide kiiremaks muutmiseks ja mudelite muutustega kohanemiseks. Näiteks komponendi loodud ühendust värskendatakse automaatselt, kui kasutaja muudab selle ühendatavaid osi [20]. Kuid pistikprogrammid luuakse tavaliselt ühekordseks kasutamiseks, näiteks ühe ühenduse tüübi jaoks. Veelgi enam, pistikprogrammide loomisel tuleb kasutada kõrgetasemelist programmeerimiskeelt, näiteks C#, mis nõuab tarkvaraarenduse oskusi, mida ehitusinseneride seas ei leidu.[10] [9]

Makrod/skriptid on tavaliselt toimingud, mida saate Tekla Structuresiga töötades salvestada. Makrod on põhimõtteliselt C# lähtefailid (.cs), mis kompileeritakse käitusajal ja mida saab redigeerida. Käsud on aga tavaliselt lihtsad ja Tekla Structures kasutab neid lihtsate toimingute tegemiseks, näiteks väljundfaili eksportimiseks.[10]

1.2.4 AutoCAD

AutoCAD on üks enim kasutatavaid projekteerimistööriistu. Kahjuks on seda paljude puuduvate funktsioonide tõttu raske BIM-tööriistaks nimetada. Kuigi AutoCAD suudab objekte modelleerida "plokkide" abil, ei säilitada objektide vahel parameetrilisi suhteid ja terviklikkust, nagu seda teeb BIM-platvorm. AutoCAD'il on laiendused tahkude ja pindade modelleerimiseks ning see toetab 3D-geomeetria "plokkide" määratlust. AutoCAD pakub mõningaid parameetriliste tööriistade funktsioone, sealhulgas võimalust luua kohanduva käitumisega kohandatud objekte. AutoCAD'i joonistusruum

on seotud 3D-mudeli mudeliruumiga ja selle praeguses tõlgenduses pakub mudeli ja annoteeritud jooniste vahel ühesuunalisi seoseid. Mudelvaated on lihtsad ortogonaalsed projektsioonid, millel on piiratud vaatekontroll. Liideste hulka kuuluvad DGN, DWG, DWF™, DXF™ ja IFC. Autodesk on julgustanud kolmandaid osapooli kasutama AutoCADi platvormina ja arendama uusi funktsioonikomplekte erinevates AEC domeenides, pakkudes võimsaid API-sid, sealhulgas AutoLISP, Visual Basic, VB Script , C# .NET, ARX (C++) liidesed [21]. See on kaasa toonud ülemaailmset arendajate kogukonna kasvu, kus on palju sõltumatuid ettevõtteid, kes pakuvad konstruktsioonide projekteerimise ja analüüsi, torustike ja seadmete projekteerimise, juhtimissüsteemide, elektrisüsteemide projekteerimise, konstruktsiooniterase, tulekustutussüsteemide, kanalite, puitkarkasside jms pakette. [11]

AutoCAD-põhiste rakenduste tugevad küljed: AutoCADi kasutajate jaoks lihtne juurutamine tänu ühtsele kasutajaliidesele; on lihtne kasutada, kuna need põhinevad hästi tuntud AutoCAD-i funktsioonidel ja liidesel 2D-jooniste tegemiseks.

Kahjuks ei võimalda AutoCad eksportida BVB ja SWP failivormingud. Seega ei vasta AutoCad kirjeldatud punktis 1.1.2 saksa tehaste failivormingute nõuetele.

Mõnede AutoCADi töövoogude automatiseerimine on võimalik AUTOLISPis, Visual Basicus ja SCR skriptide kirjutamisega. [11]

Skripti (.scr) automatiseerimise avaldasid AUTOCADi arendajad esmakordselt versioonis 1.4. Skripti eesmärk on ülesande automaatne simuleerimine käsureal. Kõik, mis AUTOCADi tööaknas on kirjutatud, kuvatakse käsureal ja seda saab lisada skriptifaili. [22]

Skriptifailidel on SCR laiend. Skriptifailid on tavalises ASCII-vormingus. Sel põhjusel ei saa kasutada tekstitöötlusprogrammi, näiteks WordPad, Write või Word. Selle asemel saab skriptide kirjutamiseks kasutada NotePadi. Skriptidel on teatud piirangud, mistõttu esinevad probleemid salvestamisel ning teiste programmisega koostoimetamisel.

Visual Basic for Applications (VBA) on Microsofti poolt loodud programmeerimiskeskkond, mis on sisse ehitatud rakendustesse operatsioonide automatiseerimiseks. VBA pakub tööriistu, mida saab kasutada graafilise kasutajaliidese (GUI) ja programmeerimiskeele loomiseks, mida saab kasutada AutoCADi objektidega suhtlemiseks. [23]

VBA töötab rakendustega, suheldes ja manipuleerides objekte rakenduse objektiteekide kaudu, ilma et tal oleks rakenduse sisemise tööga mingi eriline seos. Kõik, mida peab

teadma on kaasatud objektide nimed, et pääseda nende funktsioonidele VBA-koodi abil juurde. Kui sisestada koodisse objektide nimed, pakuvad VBA integreeritud arenduskeskkonna (IDE) redigeerimisfunktsioonid ripploendeid üksustest, mis vastavad sisestatud objekti tüübile. VBA pakub oma objektide, märksõnade ja konstantide komplekti, mis tagavad programmi täitmist, juhtimist ja silumist.[23]

Makrosid kasutatakse AutoCAD tarkvarakeskkonnaga töötamiseks, need on mõeldud erinevate kasutajate ülesannete automatiseerimiseks, näiteks neid, mida tuleb teha mitu korda.

1.2.5 Strakon

Strakon – Eestis vähetuntud BIM-programm, millel on laialdased võimalused nii inseneridele ja arhitektidele. Programm paistab silma taskukohase hinna poolest. Peamine osa kasutajatest on hetkel Saksamaa projekteerijad. Strakon on 2D/3D/BIM CAD süsteem konstruktsioonide projekteerimiseks. Fookuses on raketise ja armeerimise planeerimine. Programmi on mugav luua 2D või 3D plaane tänu hästi seadistatud tööriistadele. Projekti saab redigeerida BIM-is , 3D- või 2D-vormingus.[24] Projekteerija või projektijuht saab projekti raames otsustada, milline meetod on vajalik või kasulik.

3D-modelleerimine põhineb tahketel elementidel, mida saab lisada plokkidesse ja eraldada kihte, näiteks sorteerida korruste ja tüüpide järgi, samas on võimalik luua eraldi väljad elemendi iseloomustamiseks. See programm sobib nii monteeritavate elementide kui ka monoliitsete konstruktsioonide jaoks. Selles programmis armeerimine on võimalik nii 2D kui ka 3D kujul. Mõlemal juhul on see intuitiivne ja arusaadav isegi algkasutajatele ning 2D armeerimine on ikka seotud mudeliga. Samuti on armeerimise seadistustes sisestatud standardväärtused, mis võimaldavad automaatselt genereerida varraste ja võrkude kattumisi sõltuvalt keskkonnast ja betooniklassist. Mudeli ja jooniste vahelist ühendust saab teostada nii ühes kui ka mõlemas suunas. Vaateid ja lõikeid saab kas mudeliga siduda või nad võivad eksisteerida eraldi.

Tänu sellele, et Strakonis toimub joonise loomine segmentidena (oma mõõtkavaga virtuaalne väli), võimaldab see vajadusel kasutada vaateid ja lõikeid teistest joonistest ja mudelitest. Lisaks saab lõigete värskendamise seada automaatrežiimile, et joonist täielikult ühendada mudeliga.

Samuti pakub see tööriist võimalust luua raketis elemendi valamiseks koos edasise koodi ekspordiga vineeri automaatse lõikamise masinate jaoks. Töörist pakub automaatselt luua raketist ning annab võimalust vajadusel seda muuta.

STRAKON toetab elementide tugevdamist painutatud ja projekteeritud võrkudega koos edasise XML-vormingu ekspordiga varrastest võrgu automatiseeritud keevitamiseks. Võrkude paiknemine, mõõtmed ja üleminekud on seotud Saksa standardite failidega ja programm annab võimalust automaatselt korrigeerida armeeritud elemente.

Strakon võimaldab eksportida XML, BVB, SWP, DXF, DWG, PDF ja IFC failivormingud. Seega vastab Strakon kirjeldatud punktis 1.1.2 saksa tehaste failivormingute nõuetele.

Programmis on sisseehitatud moodulid pöördetreppide, tunnelite, sildade, teede modelleerimiseks viitega maailma koordinaatidele. DICAD-serveri kaudu on mudeliga ühiseks tööks moodul, mis annab võimalust jagada projekti blokke või tervet mudelit teiste partneritega.

Hoolimata asjaolust, et programm vastab kõigile Saksamaa tehaste nõuetele, on programmil tohutu miinus. Puudub API ja muud võimalused makrode rakendamiseks töö automatiseerimiseks. Vaatamata sellele, automatiseerimist saab rakendada Autohotkey skriptide abil.

1.3 Skriptide programmeerimiskeeled

See osa annab ülevaadet skriptide töötamise põhimõttest. Jaotises kirjeldatakse skriptikeeli, mida rakendatakse BIM-programmides automatiseerimiseks. Samuti vaadeldakse Autohotkey skriptide kirjutamise keelt, millega edaspidi tehakse tööd.

1.3.1 Skriptide töötamise põhimõtted

Õnneks on meie ajal juba olemas suur hulk programmeerimis- ja skriptikeeli. Iga kasutaja saab ise valida meetodi, mis on talle mugav ja arusaadav. Tänu suurele keele- ja juhendivalikule on skriptide kirjutamine muutunud algajatele kasutajatele lihtsaks.

Programmeerimiskeel on keel, mis on arvuti töö toimimiseks mõeldud käskude loend. Olenevalt programmeerimiskeelest on igaühel oma kompilaator. Kompilaator on omamoodi programmeerimiskeele tõlkija masintekstiks, et arvuti saaks käske ära tunda. Programmeerimiskeele kood on sageli pikk ja IT-oskusteta inimestele arusaamatu. [25]

Skriptikeel programmeerimiskeele lihtsustatud versioon. Skriptikeeltel on oma kompilaator, mis tõlgib teksti programmeerimiskeelde. See meetod aeglustab töö kiirust, sest selleks et masin saaks käsust aru, peab erineva kompilaatoriga käsu tõlkima vähemalt kaks korda. Kuna tehnoloogia on tänapäeval arenenud, ei ole see aeglustumine eriti märgatav. Tänu kompilaatoritele on kood muutunud lühemaks ja selgemaks.

Skriptid on sisuliselt kood, millele on määratud ülesanded. Need ülesanded võimaldavad automatiseerida tööd, täiendada programmide funktsionaalsust, töödelda ja muuta andmebaasi. Skriptide kirjutamise viis on individuaalne. Kõik sõltub soovitud tulemusest ja programmeerija eelistustest. Skripte saab kirjutada nii programmeerimiskeeles kui ka skriptikeeles.

Skripti võimalused sõltuvad suuresti aluseks olevast programmeerimiskeelest ja kompilaatorist. Ülesande täitmiseks tasub end kurssi viia valitud keele võimalustega, kuna skriptikeeltel on vähem funktsionaalsust ja suuremad piirangud.

Skriptikeeltes skriptide kirjutamise peamised eelised on koodi väike mahukus ja koodi parandamine tänu kompilaatoritele. See funktsioon võimaldab varakult tuvastada viga koodis, mis omakorda vähendab ka soovitud koodi kirjutamisele kuluvat aega.

Samuti tuleks töö automatiseerimiseks skriptikeelt valides tutvuda skripti meetodiga. Skriptid jagunevad vastavalt nende töömeetodile kahte tüüpi: sisemine ja väline suhtlus programmiga.

Sisemine suhtlus tähendab skripti tööd programmi sees. Selleks peab programmil olema avatud juurdepääs andmebaasile, mida saab lugeda ja muuta skriptide abil. [25]

Modelleerija üks peamisi tööriistu on kopeerimisfunktsioon. Kogenud kasutaja koostab enne modelleerimist plaani toimingute arvu vähendamiseks ja kasutab plokke maksimaalseks automatiseerimiseks mudelitega. Kuid töö automatiseerimiseks on ka teisi võimalusi, üks neist on API funktsioon.

Rakenduse programmeerimisliides (API) võimaldab kasutajatel ja arendajatel laiendada olemasoleva rakenduse võimalusi, kirjutades programmi või skripti, mis lisab tarkvarale uusi funktsioone. Näiteks Autodesk Revit API võimaldab programmeerijatel muuta hooneteabe mudeli (BIM) elemente otse või pääseda juurde andmetele, et täita eriülesandeid.[4]

API skriptid töötavad otse BIM programmi sees. Sõltuvalt tarkvara tüübist saab API-sid kirjutada erinevates programmeerimiskeeltes, nagu Dynamo, Python, C#, JAVA.

API kasutamise eelised [5]:

- mudeli loomine korduvate toimingute automatiseerimiseks;
- tootlikkuse suurendamine;
- standardiseerimise tagamine erinevates BIM-projektides;
- analüüsi, arvutuste, planeerimise ja aruandluste teostamine;
- arvutipõhise disaini potentsiaali arendamine.

Välise suhtluse mõistmine tähendab töötamist ilma rakendusega otsese ühenduseta. Programmide jaoks, mis ei toeta API-sid, on veel üks meetod. Väline suhtlus programmiga, kasutades hiire liigutamist ja klaviatuuriga manipuleerimist. See meetod pole professionaalses modelleerimises populaarne. Seda meetodit kasutatakse enamasti rutiinse arvutitöö hõlbustamiseks. Näiteks kiire teabe sisestamine või klahvide ümberjaotamine klaviatuuril. Kuid programmeerimiskeele ja programmi omaduste tundmisega on võimalik oluliselt hõlbustada jooniste modelleerimist ja kujundamist igas BIM-keskkonnas.

1.3.2 C#

C# on objektorienteeritud programmeerimiskeel, avatud lähtekoodiga, lihtne, kaasaegne, paindlik [27]. See programmeerimiskeel sobib hästi programmide loomiseks. Seda saab kasutada ka eraldiseisvate skriptide jaoks, kuid uute kasutajate jaoks võib selle õppimine olla pisut keeruline.

Mõned inimesed on C#-d skriptikeelena laialdaselt kasutanud. C# annab järele sellistele populaarsetele skriptikeeltele, nagu Python ja Java. C#-t saab tulemuslikult kasutada, kuid rakendustes, mis luuakse automatiseerimise võimaldamiseks. Antud skriptikeel ei ole parim valik. Igat tüüpi koodid, sealhulgas põhikoodid, nõuavad segadusttekitavate märksõnadega standardkoodi. Mittespetsialistile pole see väga selge.

C# on kompileeritud, komponentidele orienteeritud ja objektorienteeritud keel, mis kompileeritakse .NET-i vahekeeleks. C#-l on keelekonstruktsioonid, mis otseselt toetavad neid kontseptsioone, muutes selle loomulikuks keeleks, mis on võimas tarkvarakomponentide loomiseks ja kasutamiseks. Peamiselt kasutatakse C#-i .NET-i rakenduste arendamiseks, mis tähendab, et igal C#-rakendust kasutamisel, rakendatakse .NET-i raamistikku. C# on kõrgetasemeline programmeerimiskeel, mida on lihtne lugeda ja kirjutada nendele, kellel on programmeerimiskogemus olemas. C#

abil saab automatiseerida ülesandeid, mis vajavad pikaajalist saavutamist väiksemate tulemustega. Võrreldes teiste programmeerimiskeeltega on C#-l mõnevõrra madalam õppimiskõver, kuigi see pole nii lihtne mõista kui Python neile, kes kasutavad programmeerimiskeelt esimest korda.[28] [29]

1.3.3 Ruby

Ruby on nagu Autohotkey tõlgendatud skriptikeel. See tähendab, et programmi teksti konverteerimine masinkoodiks toimub vahetult selle täitmise protsessis ja programm ise on tekstiskriptifail. Keel ühendab Perli-laadse süntaksi objektorienteeritud lähenemisviisiga. Samuti on mõned funktsioonid laenatud programmeerimiskeeltelt Python, Lisp, Dylan ja CLU. Ruby keeletõlgi platvormide ülene teostus levitatakse avatud lähtekoodiga tarkvara tingimuste alusel.[30] [31] [32]

Ruby on võimas skriptikeel. Skripte saab kasutada selliste toimingute automatiseerimiseks nagu failide loomine ja otsimine ning alamkataloogide haldamine ja korraldamine.[33][17]

Kahjuks ei toetata Rubyt enam Reviti 2022 makrodes, kuid seda saab siiski kasutada varasemates versioonides [34]. Samuti seda saab kasutada ka skriptikeelena programmiga väliseks suhtlemiseks.

Lihtsate skriptide kirjutamise puuduseks on nende käitamine. Kui Autohotkey-is piisab skriptifailil endal klõpsamisest, siis Ruby-failide puhul tuleb need käivitada käsurealt või muude tugiprogrammide kaudu.

1.3.4 Python

Python on aktiivselt arenev skriptikeel, mida kasutatakse kõige erinevamate probleemide ja ülesannete lahendamiseks. Python on kasulik arvuti- ja mobiilirakenduste loomisel, seda kasutatakse suure infohulgaga töötamisel, veebilehtede ja mitmesuguste muude projektide arendamisel ning masinõppes.[35]

Kuna Python on läbimõeldud programmeerimiskeel, sobib see hästi igapäevaste reaalmaailma probleemide lahendamiseks. Sellel on lai rakenduste valik: tööriist muude tarkvarakomponentide haldamiseks ja eraldiseisvate programmide juurutamiseks. Python ei hõlma mitmeotstarbelise programmeerimiskeelena ainult seadmete ja süsteemi programmeerimist, vaid sellel on ka tohutul hulgal tasuta kvaliteetseid mooduleid, mida saab kasutada programmi mistahes osas. Moodulites on juba rakendanud palju vajalikke programmi üksikasju, mis aitavad tööd automatiseerida.

Pythoni lihtsus ja suur arenduskiirus teevad sellest suurepärase GUI (Graphical User Interface) arendustööriista. Pythoniga on kaasas standardne tkinteri moodul, mis võimaldab Pythoni programmidel rakendada kaasaskantavat GUI-d, millel on operatsioonisüsteemi välimus ja tunne.

Python on hea kompromiss üldotstarbeliste keelte ja matemaatikapakettide vahel. Iseenesest sobib "puhas" Python vaid lihtsate arvutuste tegemiseks. [36] [37]

Python sobib suurepäraselt skriptikeele rolli: lihtsate skriptide käskude jada ei pea olema kuidagi vormistatud ja skriptide käitamine on võimalikult lihtne. Selline kasutuslihtsus koos Pythoniga kaasas oleva suure hulga kasulike moodulite ja funktsioonidega teeb Pythonist heaks tööriistaks erinevate toimingute automatiseerimiseks, mida arvutiga töötades käsitsi teha ei tahaks.

Kuigi mõned kasutajad kaebavad erinevaid skriptitõrkeid, on see programmeerimiskeel endiselt väga populaarne.

1.3.5 AutoHotkey

AutoHotkey on tasuta avatud lähtekoodiga skriptikeel Windowsi jaoks, mis võimaldab kasutajatel hõlpsasti luua lihtsamaid kui ka keerukamaid skripte igasuguste toimingute jaoks, nagu vormitäitjad, automaatne klõpsamine, makrod, kuid ei suhtle otseselt programmidega. Antud skriptikeel on kirjutatud C++ baasil. AutoHotkey pakub lihtsat ja paindlikku süntaksit, mis võimaldab keskenduda rohkem käesolevatele ülesandele, mitte igale väiksele tehnilisele asjale. See ei toeta mitte ainult populaarset imperatiiv-protseduurilist paradigmat, vaid ka objektorienteeritud ja käsupõhist programmeerimist. [38]

Sellel keelel on suur juhend koos näidistega iga süntaksi kohta ja koodi loogikast on lihtne aru saada. Iga süntaksi nimi on inglise keeles ja see tähistab enamasti nimele omast tegevust. Näiteks MouseMove käsk nii inglise keelest tõlkes kui ka käsu definitsiooni järgi tähendab sama asja – hiirekursori liigutamist.

Skriptid ei vaja programmiga suhtlemiseks pikki failiteid. AHK suhtleb programmi aknaga ja selleks piisab selle aktiveerimisest soovitud akna nimega käsuga WinActivate. See ei loo töö otsest seost programmiga, kuid oluliselt vähendab koodi pikkust.

Skripti loomiseks piisab sellest, kui kirjutada kood tekstifaili ja salvestada see AHK formaadis. Koodi käivitamiseks peab faili avama. See keel on lihtne ja kergesti meeldejääv. Koodid on suuruselt väiksemad kui näiteks C# keeles.

1.3.6 Koodi võrdlus

Koodi keerukuse mõistmiseks on tabelis 1.4.5.1 toodud erinevate skriptikeeltega koodide näited. Need koodid täidavad sama funktsiooni- notepadi käivitamist.

Tabel 1.3.6.1 Koodi näited

Autohotkey	Run, notepad.exe
C#	System.Diagnostics.Process.Start("Notepad.exe")
Python	import subprocess subprocess.Popen("C:\\Windows\\System32\\notepad.exe")

Ruby	cmd = "/usr/local/bin/code
	#C:\\Windows\\System32\\notepad.exe"
	pid = Process.spawn cmd

Tabelist 1.3.6.1 on näha, et lühim kood on Autohotkey skriptil. Põhjuseks on see, et Autohotkey ei ole programmeerimiskeel vaid on skriptikeel. Autohotkey kompilaator on mõeldud ainult skriptide jaoks ja ei omanda programmeerimiskeelte võimalusi, mistõttu on tal palju piiranguid. Välissuhtlumiseks BIM rakendustega on ta siiski hea alternatiiv, kuna võimaldab klaviatuuri ja hiire manipuleerimist ning aktiivsete akende tuvastamist.

1.4 Lühikokkuvõte

Saksa raudbetoonelementide tehased on ehitusvaldkonnas suured tegijad. Uute tehnoloogiate kasutuselevõtt on võimaldanud tootmisprotsessi automatiseerida, suurendades seeläbi toodangu mahtu ja vähendades inimressursi maksumust.

Saksamaa tehastega töötades peaks meeles pidama, et mitte iga BIM-tööriist võimaldab teha tööd. Projekti ettevõtmiseks tasub oma võimekust hinnata ja ennekõike kokku leppida, millistes vormingus tellija soovib jooniseid saada.

Saksa tehased eelistavad kasutada Strakoni programmi põhjusel, et programm on algselt kirjutatud Saksa standardite jaoks ja suudab toota kõiki tehase jaoks vajalikke failivorminguid. Tekla ja Reviti abil on täiesti võimalik töötada Saksa tehastega. Ainus raskus on tööriista seadistamine tehase nõuetele. Neid programme saad edukalt kasutada, kuigi väikestes projektides nende otstarbekus pisut küsitavam, eriti arvestades seda asjaolu, et Strakon on hinna poolest taskukohasem.

Kuigi Strakon on selle töö jaoks suurepärane programm, pole see ideaalne. Makrode automatiseerimiseks ja kirjutamiseks vajalike tööriistade puudumine muudab selle programmiga töötamise keeruliseks.

Selle probleemi lahendusena kaaluti skriptide kirjutamise võimalust Autohotkey abil. Autohotkey on selleks otstarbeks suurepärane, sest erinevalt C #-st, Pythonist ja Rubyst on see lihtsam analoog ega nõua IT-valdkonna põhjalike teadmisi.

2. LÄHTEMATERJALID JA METOODIKA

Lõputöö eesmärk on hinnata BIM-i automatiseerimise parimaid teadmisi ja praktikaid tootmise täpsusega mudelite ja jooniste koostamiseks. Selleks peab välja töötama raamistik ja protsessi mudel, mida saab kasutada tehase tootmismudelite ja -jooniste koostamise automatiseerimiseks. Lõputöö eesmärk on optimeerida ja automatiseerida tööprotsessi BIM tarkvaraga skriptimise abil.

Selle eesmärgi saavutamiseks otsitakse vastuseid järgmistele küsimustele:

- Millised aspektid takistavad tööd valitud tööriistaga?
- Kuidas saab BIM tööriista automatiseerida ja millist kasu see toob?

Töö on jagatud mitmeks etapiks. Kõigepealt viiakse läbi BIM tarkvara analüüs, et välja selgitada aspekte, mis takistavad tööd valitud tööriistaga. Järgmisel etapil käsitletakse skriptide väljatöötamise metoodikat skriptimiskeele abil. Lõpuks viiakse läbi tulemuste analüüs, kus kirjeldatakse skriptide rakendamist ja testimist ettevõttes koos katsetulemuste andmetöötlusega.

2.1 Tarkvarade analüüsi meetodid

BIM-tööriista automatiseerimise ja optimeerimise edasiseks tööks tuleks välja selgitada programmide piirangud. Puuduste ja piirangute tuvastamiseks valiti välja Strakon tarkvara. Antud tarkvara võimaldab eksportida vajalikud failivormingud saksa tehaste jaoks ning on võetud kasutusele ettevõttes Keskkonnaprojekt OÜ. Skriptikeeleks oli valitud Autohotkey, kuna see võimaldab ühendamist teiste programmidega ning on lihtne kasutada.

2.1.1 Strakon tarkvara analüüs

Strakon BIM-tööriistaga tööprotsessi võib jagada neljaks osaks ning nende osade järgi saab teha tarkvara analüüsi. Tegemist on järgmiste osadega:

- tööriista ergonoomika;
- modelleerimine;
- jooniste vormistamine;
- töö automatiseerimisvõimalused.

Tööriista ergonoomika all mõistakse Strakon programmi mugavust kasutamisel (projektide nimekirjas navigeerimine, käsurea paigutus, kiirklahvid jms.). See on kõige

tähtsam osa, kuna töö kiirus sõltub sellest, kui mugav on tööriista kasutamine ja kas seda on võimalik iga kasutaja jaoks seadistada.

Modelleerimise ja jooniste vormistamise all mõistakse Strakon tarkavas moodulite olemasolu projekti lõpuleviimiseks. See osa võimaldab selgeks teha tööriista võimalusi ning aru saada edaspidiste skriptide kirjutamise vajadusest.

Töö automatiseerimisvõimaluste all mõistakse Strakon tarkvaras moodulite olemasolu, mis võimaldavad viia läbi modelleerimise, armeerimise ja vormistamise. See osa on vajalik olemasolevate automatiseerimistööriistade võimaluste ja nõrkuste väljaselgitamiseks.

Strakon programmi puuduste ja piirangute tuvastamiseks viidi läbi küsitlus Keskkonnaprojekt OÜ ettevõttes. Uuringus osalesid kuus ettevõte töötajad, kellel oli töökogemus Strakon programmi kasutamisel. Kõiki kogemus Strakon programmiga on vähem kui 3 aastat. Vestlused viidi läbi alates 01.09.2021 ning nende raames esitati küsimusi Strakon tarkvara piirangute kohta. Küsimused olid jaotatud neljaks osaks vastavalt tarkvara tööprotsessidele: ergonoomika, modelleerimine, jooniste vormistamine ja töö automatiseerimine.

2.1.2 Autohotkey skriptikeele tutvustus

Autohotkey skriptikeele võimaluste ja piirangute tuvastamiseks oli vaja kõigepealt tutvuda antud skriptikeele juhendiga. Selle keele võimaluste ja piirangute tuvastamiseks tehti Strakon programmi analüüs. Kõikide võimaluste ja piirangute väljaselgitamine ei ole käesoleva uuringu raames otstarbekas suure töömahu tõttu.

Skriptide väljatöötamine toimub Strakon tarkvara analüüsi alusel. Iga tarkvara puudus on fikseeritud ning igaühe jaoks autor pakkus välja lahenduse. Lahendused on formuleeritud skriptide väljatöötamise protsessis, võttes arvesse skriptimiskeele ja tarkvara piirangud.

Pärast skriptide koostamist, koostakse abiprogrammi struktuuri skeem, mille järgi kirjutatakse abiskripte. Skeemi luuakse selleks, et skriptide kasutamine oleks mugav ja arusaadav. Samuti skeemi järgi saab paigutada skripte kaustas, mis lihtsustab edaspidist skriptide rakendamist.

2.2 Katsetulemuste analüüsi metoodika

Selleks, et veenduda väljatöötatud lahenduse tõhususes, testitakse väljatöötatud automatiseeritud lahendusi ettevõttes. Järgnevalt kirjeldatakse katsemeetodit ja tõhususe arvutusmeetodit.

2.2.1 Skriptide testimine

Katse viiakse läbi KESKKONNAPROJEKT OÜ raudbetoonkonstruktsioonide osakonnas. Katse eesmärgiks on rakendada skripte ning hinnata nende mõju tööprotsessile. Katses võtab osa 6 tehnikut (Tabel 2.2.1.1), kes on Strakon tarkvaraga töötanud vähemalt kaks aastat.

Tabel 2.2.1.1 Katses osalevate tehnikute tööstaaž ehitusvaldkonnas ja Strakon tarkvara kogemus.

Tehnik	Tööstaaž ehitusvaldkonnas	Töökogemus Strakon tarkvaraga
1	3 a.	3 a.
2	5 a.	2.5 a.
3	3 a.	3 a.
4	3 a.	3 a.
5	>10 a.	2.5 a.
6	>10 a.	3 a.

Katseks valiti välja kaks erinevat tüüpi monteeritavat raudbetoonelemendi: kolme kaldega rõduplaat ja sirge trepimarss. Tegemist on ühe tehase poolt tehtud betoonelementidega. Elemendid valitakse erinevatest projektidest, aga sama raskusastmega.

Rõduplaatide modelleerimise ülesanne sisaldab plaadi paigutamist 3 kaldega, ühe drenaaztoruga ning ühel küljel asuvate Schöck Isokorb [1] toodetega. Eeldatakse, et ülesandes kajastatakse elementide armeerimine vastavalt Schöck Isokorb juhenditele ja jooniste vormistamine vastavalt tehase nõuetele.

Trepimarssi modelleerimise ülesanne näeb ette sirge trepimarsi modelleerimist ja armeerimist vastavalt staatikale ja Schöck Isokorb juhenditele. Vormistatakse joonis vastavalt tehase nõuetele.

Kõik projektid menetletakse läbi Asana platvormi. Antud platvorm võimaldab koostata projektidest loetelu, kus on kajatatud projekti ülesanded ning nende staatus.

Ülesanne täitmisel tehnik peab muutma projekti staatust ning välja lülitama taimerit. [39]

Kõik ressursikuludega seotud andmed viiakse sisse Everhour süsteemi, mis omakorda arvutab, kui palju aega läks ülesannete lahendamiseks ning töötab Asana platvormiga kooskõlas. [40] Tehtud tööde mõõtühikuks on minut. Kulunud aeg arvutakse iga elemendi jaoks eraldi.

2.2.2 Ajakokkuhoiu hindamine

Ajakokkuhoiu hindamiseks kantakse saadud katsetulemused Tabelisse 2.2.2.1. Elemendid on jaotatud tüübi järgi kahte gruppi. Tabel on jaotatud kahte ossa: enne ja pärast skriptide rakendamist. Ajakokkuhoiu hindamiseks võrreldakse keskmist ajakulu elemendi kohta enne ja pärast skriptide rakendamist.

Tabel 2.2.2.1 Elementide kogus ja ajakulu.

	Enne skriptide rakendamist	Pärast skriptide rakendamist
Rõduelementide arv	…	…
Ajakulu minutites	…	…
Keskmine ajakulu elemendi kohta	…	…
Treppelementide arv	…	…
Ajakulu minutites	…	…
Keskmine ajakulu elemendi kohta	…	…

Tõhususe kinnitamiseks keskmine ajakulu elemendi kohta võrreldakse terve projekti ajakuluga. Saadud tulemused kantakse Tabelisse 2.2.2.2. Tabel on jaotatud kahte ossa: enne skriptide rakendamist ja pärast skriptide rakendamist. Ajakokkuhoiu hindamiseks võrreldakse keskmist ajakulu elemendi kohta enne ja pärast skriptide rakendamist.

Tabel 2.2.2.2 Ajakulud elementidele terve projekti kohta.

	Pärast skriptide kasutamis	Enne skriptide kasutamist
Pojektide arv	…	…
Elementide arv	…	…
Ajakulu tundides	…	…
Keskmine ajakulu elemendi kohta	…	…

3. TULEMUSED JA ARUTELU

Töö järgnevas osas käsitletakse Strakon programmi puudusi ja nende kõrvaldamise võimalusi skriptide abil. Vaadatakse üle ka võimalikke automatiseerimise võimalusi vastavalt Keskkonnaprojekt OÜ soovidele. Uurimisprogrammiks on Strakon ja valitud skriptimiskeeleks on Autohotkey. Selles jaotises antakse ülevaade tehtud töödest ning kirjeldatakse koostatud skriptide funktsioone ja nende rakendusala. Jaotises kirjeldatakse tehtud töö analüüsi tulemust, mis on aluseks tegevuse tõhususe hindamiseks.

3.1 Strakon tarkvara hinnang, puudused ja võimalikud lahendused.

Strakon programmi puuduste ja piirangute tuvastamiseks viidi läbi küsitlus Keskkonnaprojekt OÜ ettevõttes. Uuringus osalesid kuus ettevõte töötajad, kes omavad kuni kolmeaastast kogemust programmi kasutamisel. Vestlused viidi läbi alates 01.09.2021 ning nende raames esitati küsimusi töö takistavatest Strakon tarkvara teguritest. Küsimused olid jaotatud neljaks osaks vastavalt tarkvara abil teostatavale tööprotsessile: ergonoomika; modelleerimine, jooniste vormistamine, töö automatiseerimine.

Järgnevalt käsitletakse Strakon programmi puudusi, nende kõrvaldamise viise, kui ka võimalikke viise töövoo automatiseerimiseks Autohotkey skriptimiskeele abil. Puudused kujunesid välja Keskkonnaprojekt OÜ ettevõtte Strakon kasutajate arvamuste põhjal.

3.1.1 Tarkvara ergonoomika

Küsitluse põhjal tuvastati kolm Strakon tarkvara ergonoomikaga seotud puudust. Nende puuduste loetelu on subjektiivne, kuid oluline ettevõtte vaatevinklist. Tuvastatud puudused kirjeldatakse allpool.

Kiirklahvid

Joonise kujundamine Strakon-is toimub 2D-aknas. Joonise redigeerimise tööriista aktiveerimine toimub ainult hiirega. On võimalik lisada sageli kasutatavaid funktsioone ekraani vasakusse serva, kuid paraku pole sellist võimalust klaviatuuril. Joonise kujundamisel tuleb sageli hiire kursor töödeldavast objektist eemale viia, mis omakorda

vähendab töö kiirust. Lisaks ei saa kõiki käske lisada sageli kasutatavate funktsioonide loendisse. Mõnda neist tuleb otsida menüükaartidelt.

Võimalik lahendus: Skripti loomine, mis võimaldaks klaviatuuri abil funktsioone aktiveerida.

Keel

Strakon oli algselt saksakeelne programm, alles hiljuti ilmus ingliskeelne versioon. Kahjuks pole siiani kõik moodulid tõlgitud. Kasutajatel, kellel on kogemus programmiga töötamisel, pole sellega probleeme, kuid uuele kasutajale võib see tunduda ebamugav.

Võimalik lahendus: Raamatukogu loomine, mis kajastaks teksti juurdepääsetavas keeles, kuid selle sisestamisel asendaks selle saksa keelega. Samuti lahenduseks võiks olla skripti loomine, mis annaks võimaluse tõlkida teksti abiaknas.

Programmil puudub revisjonide nimekiri. Arvestades ettevõtte sihtturgu (Saksamaa), tuleb kõik revisjonid kirja panna käsitsi ning võõrkeeles. See omakorda aeglustab saksa keelt mitteoskavate kasutajate tööd ja viib sageli vigade tegemiseni.

Võimalik lahendus: Skripti loomine, mis aitab simuleerida automaatse täitmise funktsiooniga versioonide teeki.

Projekti seaded ja navigeerimine

Strakon-il on materjaliseaded, kuid kahjuks pole universaalset tööriista, mis võimaldaks luua materjalide teeki. Probleem seisneb selles, et uue projekti loomisel on olemas ainult üks mall materjalide arvutamiseks. Joone värv ja stiil on seotud materjaliga ning kahe projekti vahel liikudes on tõenäoline, et mudeli ja joonise ekraanikuva on valed.

Projektide ja jooniste vahel vahetamine toimub faili otsimisel Strakoni programmi kaudu. Puudub raamatukogu, kus saaks projektide loendeid hoida.

Võimalik lahendus: Installifailide loomine koos lisandmoodulite ja kaustasüsteemiga.

3.1.2 Modelleerimine ja jooniste vormistamine

Järgnevalt kirjeldatakse Strakon programmi modelleerimisega ja jooniste vormistamisega seotud puuduseid ja võimalike lahendusi Autohotkey skriptimiskeele abil. Puuduste loetelu on subjektiivne ja on koostatud vastavalt Keskkonnaprojekti OÜ töötajate arvamustele.

Koosteelemendid

Strakon-is saab koosteelemente paigutada 2D-vormingus joonisele ja 3D-mudelile. 3D puhul ei ole võimalik kiiresti valida sama projekti raames sageli kasutatavaid koosteelemente. Detaili paigutades tuleb otsida vajalik nimi suurest nimekirjast, mis omakorda aeglustab ka töövoogu.

Võimalik lahendus: Teegi loomine koosteelementide kiireks otsimiseks 3D-s.

Sisseehitatud osa paigutamine 2D-sse toimub ploki paigutamisega ja sellele positsiooninumbri määramisega. Probleem on selles, et muutuva pikkusega detaili positsiooninumbri sisestamisel, tuleb iga kord sisestada detaili täisnimi. See protsess ei ole mitte ainult aeganõudev, vaid suurendab ka vea tegemise võimalust.

Võimalik lahendus: Mugava liidese rakendamine koosteelementide loomiseks ja paigutamiseks 2D-vormingus.

Armeerimine

Armeerimine Strakon-is on võimalik nii 2D-s kui ka 3D-s. 2D-s teostatakse lõigete tugevdamine. Selleks peate joonestama varda kontuuri, määrama sammu ja lisama kordusteguri. Probleem on selles, et suure tõenäosusega on seatud vale kordustegur, mis lõpuks mõjutab sarruse kogust. 2D meetod ei ole usaldusväärne, kuna armatuuri ristumisi on võimatu avastada.

Armeerimine 3D aknas võimaldab tuvastada kokkupõrkeid, kuid armeerimisprotsess on pikk, sest iga kord tuleb käsitsi sisestada paigaldatud armatuuri läbimõõt ja samm. Pealegi on mõne varda puhul vaja iga kord sisestada õige ankurduspikkus.

Võimalik lahendus: Armeerimise jaoks liidese loomine, mis võimaldaks salvestada varraste omadusi ja määrata ankurduspikkuse automaatselt.

3.1.3 Automatiseerimisvõimalused

Järgnevalt vaadeldakse Strakon tarkvara automatiseerimisvõimalused, nende puudused ja võimalikud lahendused Autohotkey skriptide abil. Puuduste loetelu on kujunenud vastavalt Keskkonnaprojekt OÜ töötajate arvamustele ja soovidele.

Trepi moodul

Strakon-is on olemas nii sirge kui ka kumera trepi jaoks moodulid. Nende moodulite puudus seisneb nende kasutamise keerukuses. Moodulitel on suur hulk funktsioone, mida ei kasutata. Arhitektuurijoonistest trepi loomiseks tihtipeale ei ole trepi täpseid mõõte või neid tuleb sageli muuta. Mooduli aktiveerimine treppide redigeerimiseks ainult aeglustab tööd. Lisaks pakub moodul ainult treppide mudeli loomist. Joonise kujundamine toimub käsitsi, mis on aeganõudev protsess.

Võimalik lahendus: Lihtsustatud moodulite loomine treppide modelleerimiseks kui ka raketise mõõdistamiseks. Kuna Strakon-is saab objekte modelleerida koordinaatsüsteemi kaudu, on teoreetiliselt võimalik seda protsessi automatiseerida.

Rõdu moodul

Strakon-is puudub võimalus rõdude automaatseks modelleerimiseks. Kuna rõdudel on keeruline kallakutega geomeetria, ei saa iga kasutaja kiiresti aru, kuidas seda tüüpi elemendiga tööd alustada.

Võimalik lahendus: Luua rõdu mooduli. Mooduli loomine on teoreetiliselt võimalik tänu objektide modelleerimisele koordinaatide järgi. Modelleerimise mõte on kokku panna soovitud element primitiivsetest objektidest. Mooduli keerukus seisneb pikkades matemaatilistes arvutustes kõigi primitiiv-objektide loomiseks ja paigutamiseks. Samuti on raskused tilga soone ja rõdupiire sisemise ääre modelleerimisega.

3.2 Lahenduse kontseptsioon ja ülesehitus

Analüüsides Strakon tarkvara puudusi, sai arusaadavaks mille alusel kirjutati Autohotkey skripte. Skriptide mugavaks kasutamiseks koostati struktuur, mis aitab aru saada skriptide tööpiirkonnast ja nende sõltuvusest teistest skriptidest. Järgnevalt kirjeldatakse skriptide kontseptiooni ja põhimõtted.

3.2.1 Struktuur

Täielikuks arusaamiseks, milliseid skripte tuleb kirjutada ja kuidas neid käivitada, koostati skeemi, mis kirjeldab abiprogrammi struktuuri ning skriptide paiknemist süsteemis (Joonis 3.2.1). Kõik ettenähtud skriptid asetakse ettevõtte serverisse, millele on juurdepääs igal programmi kasutajal. Failide paiknemine on tingitud asjaolust, et faili värskendamisel saavad kõik kasutajad selle värskenduse õigeaegselt kätte. Samuti võimaldab see asukoht faile värskendada, ilma uuesti aktiveerimiseta. Kõik skripti

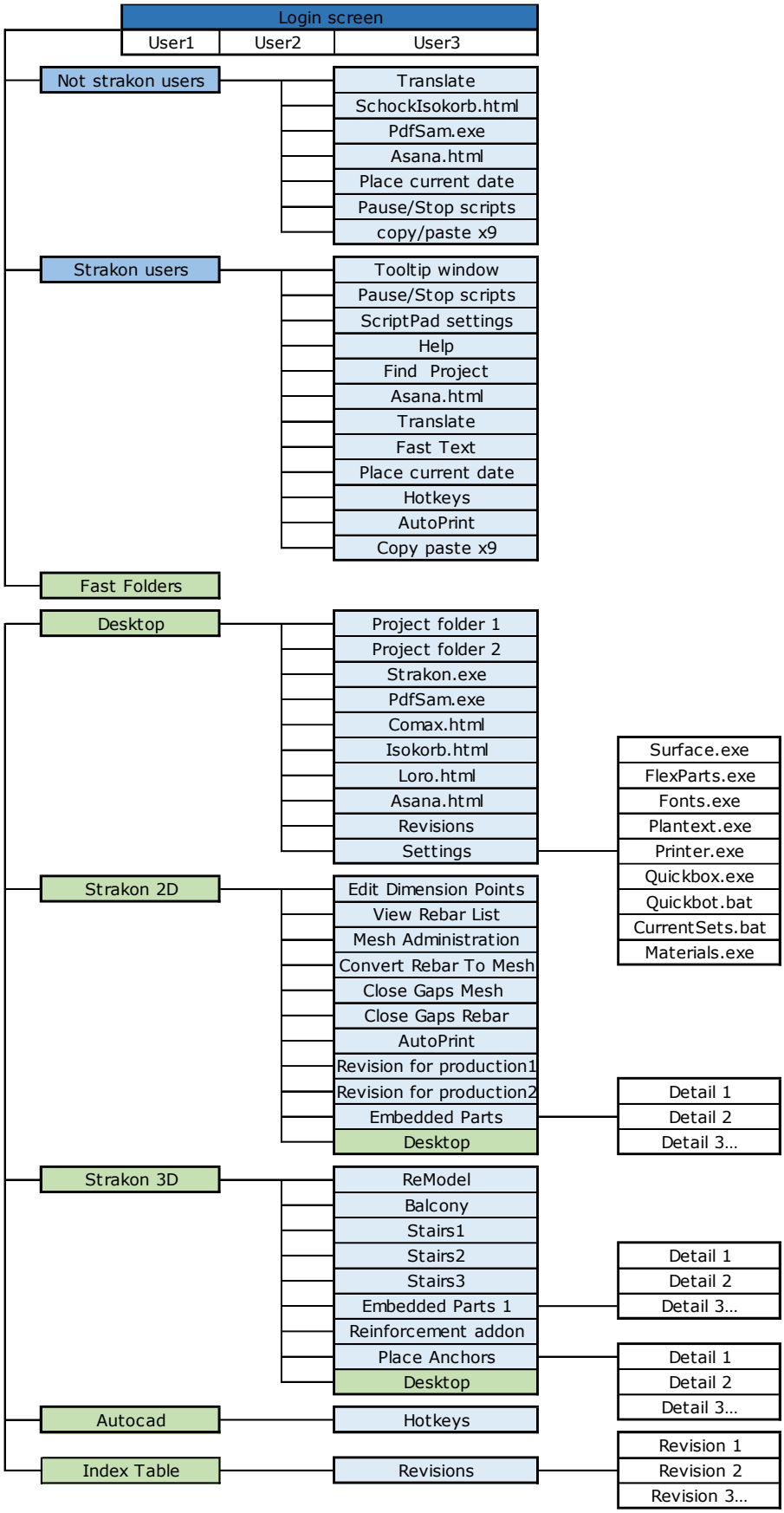
aktiveerimised toimuvad ilma kasutaja sekkumiseta. Samuti failide paiknemise valik on tingitud intellektuaalomandi turvalisse kasutamise kaalutlustest. Selle programmi kopeerimine ja seadmetes kasutamine ei toimi. Programm on loodud töötama ainult ettevõtte serveris. Muidugi saab üksikasjaliku uurimise ja mõne skripti eraldi käivitamisega teatud tulemuse saada, kuid kogu programmi mõte läheb kaotsi.

Programmi ja sellega seotud skriptide automaatne käivitamine toimub serveris, kuid tegelik kooditöötlusega seotud töö toimub kasutaja arvutis. Ühesõnaga, kasutaja arvuti kopeerib käivitamise hetkel koodi ja töötab sellega. See funktsioon võimaldab enamikul skriptidel sujuvalt töötada isegi siis, kui ühendus serveriga on katkenud. Samuti antakse serveriga ühenduse katkemisel eraldi kood, mis uuendab faile teatud sagedusega. See kood võimaldab administraatoril peatada skriptide värskendamist kõigis kasutajate kasutusel olevates arvutites, et vältida kokkujooksmisi.

Programm võimaldab luua kasutaja isiklikku kausta ja käivitada menüü sätete jaoks. Kuna kasutajad saavad kogemata kasutada ja muuta teiste inimeste sätted, pakuti kasutajale parooli loomiseks eraldi moodul. Selle toimingu põhimõtte seisneb selles, et kasutaja jaoks luuakse fail ja see salvestatakse tema arvutisse. Konkreetse kasutaja valimisel kontrollitakse koodiga faili. Kui failis on käivitamiseks vajalik teave, siis on luba olemas.

Abiprogrammi struktuuri (Joonis 3.2.1) loodi lähtudes Strakon tarkvara võimalustest ja Keskkonnaprojekt OÜ kasutajate soovidest. Skeem kirjeldab sõltuvusi ja failide asukohti süsteemis. Iga sektsioon vastutab oma aktiivse akna eest ning laadib oma ülaltoodud diagrammil kirjeldatud skriptide loendi. Struktuuri keskmes on kasutaja valimine skript. Teised jaotised sõltuvad sellest skriptist ja vastutavad skriptide käitamise eest olenevalt tingimustest. Mainitud eraldamine loodi selleks, et oleks võimalik jälgida skripti tööala menüüs ja vähendada üldist funktsioonide loendit.

Kõik skeemis kirjeldatud programmid loodi kasutades Autohotkey skriptikeelt, välja arvatud installiprogrammid, mis loodi Winrari ja Batch tekstidokumenti abil. Skriptid töötavad hiire ja klaviatuuriga manipuleerimise ning tööaknahalduse abil. Skriptide käivitamiseks peab arvutis olema installitud Autohotkey. Skriptide ja teiste installeerite käivitamiseks midagi installima ei pea. Kõik operatsioonid toimuvad arvuti süsteemis ning vajalike faile skript otsib ettevõtte serverist ise. Töölaual võib olla otsetee skriptifailile, või konfigureeritakse automaatne käivitumine pärast arvuti sisselülitamist.



Joonis 3.2.1. Abiprogrammi struktuur

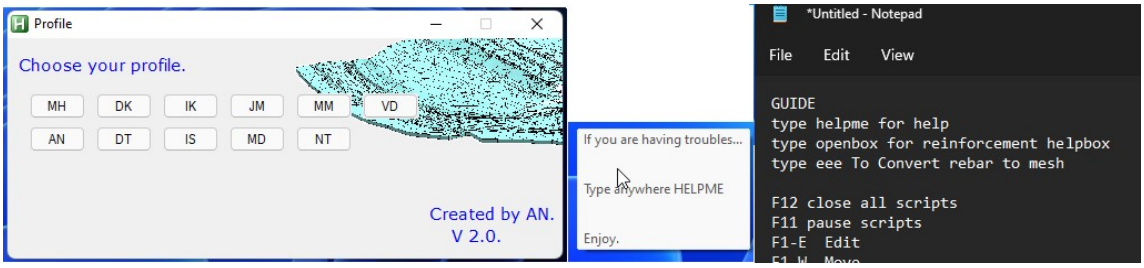
3.2.2 Skriptide ülevaade

Järgnevas lõigus vaadeldakse kirjutatud abiprogrammi skripte ning seletatakse nende tööpõhimõte. Järgnevate skriptide koodidega saab tutvuda lisades.

Kasutaja ja menüü

Pärast skripti (vt Lisa 1) käivitamist, valib kasutaja avatud aknas (Joonis 3.2.2.1 a) oma initsiaalid, mis võimaldab laadida programmil vajalikud sätted. Tänu sellele on programm paindlik ja laseb lisada igale kasutajale abiaknaid ja -programme, arvestades tema soove.

Kuna Keskkonnaprojekt OÜ kasutab modelleerimiseks ka teisi programme, siis skripti kasulikke funktsioone kasutamiseks on ette nähtud kasutajate klassifitseerimine. Pärast kasutaja valimist, programm laadib seaded, lähtudes kasutaja vajadusest. Tegemist on programmidega „Strakon Users" ja „Not Strakon Users".



a.Kasutaja menüü b. Abiaken c. Juhendaken

Joonis 3.2.2.1 Kasutaja menüü, abiaken ja juhendaken

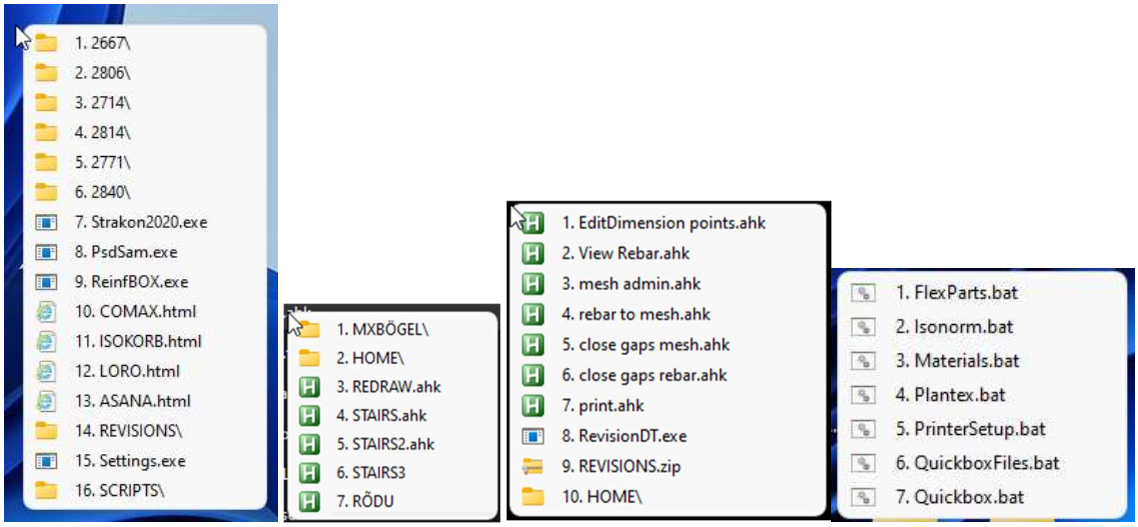
Kasutaja initsiaalide valimisel avaneb abiaken, mis tuletab meelde, et probleemide korral saab tutvuda juhendmaterjaliga."HELPME" kirjutamisel, avaneb Notepad kasutusjuhendi (Joonis 3.2.2.1 c). „HELPME" saab kirjutada kus iganes, skript automaatselt kasutab seda sõna ja teeb juhendmaterjali lahti.

Kausta süsteem

Fast Folders (vt Lisa 2) on kausta süsteemide kogum, mis on iga kasutaja jaoks individuaalne. Selle mooduli eesmärk on lihtsustada projektide ja programmide vahel navigeerimist. Moodulis asub info lemmikkaustadest, programmidest ja veebilehtedest. Soovi korral saab tekstifaili abil muuta programmide loendit, failide asukohta loendis, ikoone, kirjeldust. Igal kasutajal on link oma tekstifailile ja see asub serveris eraldi kaustas, et vältida juhuslikke kustutamisi, skriptimuutusi. Selle jaotise jaoks on ette nähtud paroolifunktsioon.

Samuti Fast Folders on ka kaustaskeem programmide kuvamiseks. Selle mooduli eesmärk on hõlbustada skriptide kasutamist. Suurim osa skripte käivitatakse just Fast Folders-i abil.

Pärast kõigi failide allalaadimist, on igal kasutajal tekkib oma eelmääratletud kiirmenüü nupp. Kiirmenüünupp võib olla mis tahes klahv, klaviatuuri kombinatsioonid, hiirenupud ja muud kolmanda osapoole tarvikute nupud ja klahvid. Kiirmenüü nupp avab programmide ja failide valikuga akna (Joonis 3.2.2.2 a,b,c,d)



a.Desktop menüü b.Strakon 3D menüü c.Strakon 2D menüü d.Seaded

Joonis 3.2.2.2 Menüü aknad

Kui menüünupp on aktiveeritud, hüppab menüü aken kursori alla vastavalt aktiivsele aknale. Olenevalt aktiivsest aknast, näeb kasutaja erinevaid tööriistavalikuid . Selle skripti toimimiseks on olemas käivitatav skript, milles on täpsustatud tingimused. Edasise töö tingimuseks on hetkel aktiivne aken. Kui näiteks Strakoni 3D aken oli tööhetkel aktiivne, siis menüü käivitamisel kuvatakse teine valik (Strakon 3D menüü). Kui tingimuses pole akent määratud, siis käivitatakse vaikimisi esimene valik (Desktop menüü).

Mooduli (Desktop) loendisse on lisatud ettevõttes enamkasutatavad programmid, Interneti leheküljed ja projektikaustad. (Project folder, Strakon.exe, PdfSam.exe, Comax.html, Isokorb.html, Loro.html, Asana.html).

Seaded

Kuna Keskkonnaprojekt OÜ ettevõttes koostatakse projekte erinevate tehaste jaoks, võtab seadete ümberkäivitamine reeglina palju aega. Laadimisseaded veerus Settings

sisaldavad kasutatud materjalide loendeid, koosteelementide loendeid, külgede tähistusi ja täite kvaliteedi sätteid. Samuti võtab Strakon programmi installimine uutesse arvutitesse palju aega. Uute seadmete seadistamise mugavuse huvides on vajalikud ülaltoodud elemendid, samuti jooniste kujundamiseks fontide, printeri, tekstide ja mallide seadistamine. Selle mooduli eesmärk on luua paigaldajate raamatukogu, mis on käepäraselt kättesaadav.

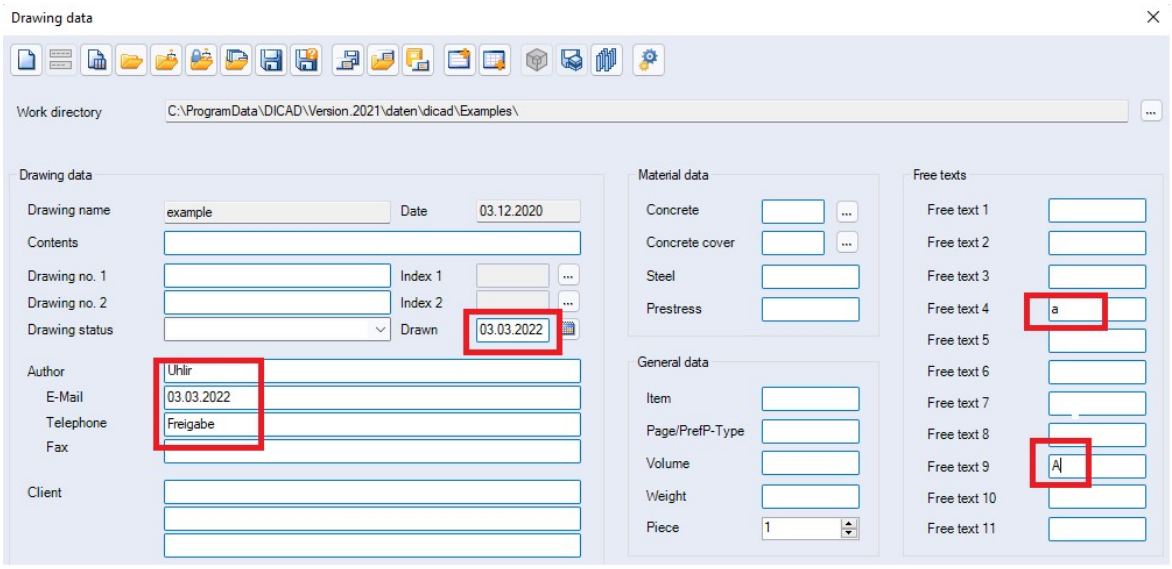
Seadete kausta avamisel avaneb installifailide loend (Seaded)(Joonis 3.2.2.2d). Installifailid loodi WINRAR SFX arhiivi abil [41], kus on määratud konkreetse faili installimise tee. See meetod hõlbustab failide installimist ja võimaldab vajadusel ka tee asendada. Skript sisaldab installifaili käivitamist, kasutades käsku "*Run, [Path].exe*".

Indeksite moodul

Joonise muutmiseks kasutatakse redaktsiooniindeksi numbrid ja redaktsiooni kirjeldus koos initsiaalide ja kuupäevadega. Mõnede tehasest saadud tootmisjooniste kujundamine nõuab palju muid pealdisi ja indekseid. Suures plaanis võimaldab Strakon teha tootmiseks mõeldud jooniste kujundamist eraldi läbivaatamise akna kaudu. Üksikute joonistega väikeste projektide puhul pole see aken praktiline. Seoses sellega, et inimfaktori tõttu esineb suur oht teha redaktsioonide väljakirjutamisel grammatilisi vigu, mugavuse huvides programmi on lisatud eraldi revisjonimoodul Revisions ja revisjoni moodulid tootmiseks (Revision For Production1) (Revision For Production2).

Redaktsioonide moodul (vt Lisa 3) võimaldab aknast valida soovitud redaktsiooni teksti oma emakeeles. Redaktsioon vaikimisi kirjutab saksa keeles. Tootmiseks ettenähtud revisjoni moodul avab joonise üldinfo akna ja täidab nõutud väljad automaatselt.

Skripti käivitamisel avaneb joonise üldteabe aken (Joonis 3.2.2.3) ja redaktsioonide jaoks vajalikud väljad täidetakse automaatselt. See skript on kasulik juhtudel, kui tootmise taotlemisel on vaja täita palju lisavälju. Vajalikud andmed ja väljakirjutamise koht sisestatakse eelnevalt koodi.



Joonis 3.2.2.3 Skripti kaudu sisestatud väljad

Skripti põhimõte:

Selle akna avamiseks peate klõpsama menüü vahekaartidel. Strakon võimaldab menüükaarte avada, kasutades "*Alt+F*" kombinatsiooni. Skripti kaudu sellele vahekaardile pääsemiseks peate kirjutama koodi "*Sendinput, {Alt down}{F down}{F up}{Alt up}*".

Lisaks, kasutades vahekaartidel navigeerimiseks käskude "alla" ja "sisestus" kombinatsioone, avab skript joonistusmenüü. Joonistusmenüüs toimub liikumine läbi *"Tab"* ja väljade täitmine toimub sama käsuga "*Sendinput*". Skript sisestab need kombinatsioonid nii suure kiirusega, et vahekaartide aknaid ei kuvata ja näeme kohe lõpptulemust. Et vältida enneaegset info sisestamist skripti, kasutatakse käsku *"WinWaitActive, Drawing data"*, mis võimaldab oodata akna avanemist ja seejärel toimingut jätkata.

Kiirklahvid

Kuna Strakon programmil puudub võimalus klahvidele joonise redigeerimiskäske seada, regulaarsel käsul ekraanil klõpsamine oluliselt aeglustab programmi tööd. Selle probleemi lahendamiseks võeti kasutusele kiirklahvi moodul Hotkeys ja kiirklahvi moodul lisatarkvarajaoks ScriptPad Settings. (vt Lisa 4)

Kiirklahvide moodul Hotkeys võimaldab valida käsu hiirt kasutamata kasutajale sobiva klahvikombinatsiooniga. Klahvikombinatsioonid valitakse, võttes arvesse kasutajate soove ja meeldejätmise lihtsust.

Tarvikumoodul ScriptPad Settings on mõeldud lisatarkvarana kasutajatele, mis võimaldab lisada klahve F13-F24. Nende abil on võimalik kasutada mis tahes mooduleid kasutaja äranägemisel. Lisatarkvarana saab kasutada skriptpadi või tavalist lisanumpadi. Numpad-i variandi puhul seadistamine võtab rohkem aega, aga seade maksab kordades vähem.

Samuti on moodulite loendisse lisatud sageli kasutatavad funktsioonid (Edit Dimension Points, View Rebar List, Mesh Administration, Convert Rebar To Mesh, Close Gaps Mesh, Close Gaps Rebar), mille abil saab otsida Strakon kaustasüsteemist konkreetset funktsiooni. Skriptid töötavad Revision skripti põhimõttel (vt Lisa 3)

Strakon programmis saab käske aktiveerida vahekaardi menüü kaudu. Kuna kõiki käske ei saa sel viisil kasutada, leiutati alternatiivne võimalus, mis seisneb selles, et võib jätta kursori asukoht meelde, liikuda soovitud koordinaatideni, vajutada nuppu ja viia kursor tagasi algsesse asukohta. Skript käivitatakse järgmiste käskudega. See skript on üks kasulikumaid, kuna see oluliselt hõlbustab rakendusega töötamist. Seda koodi saab kasutusele võtta iga rakendusega, kus pole võimalust klahvide kombinatsioone ümber kaardistada.

Skripti põhimõte:

MouseGetPos x, y - mäletab kursori asukohta

S*endInput, {Click 1130 55}* - liigutab kursor määratud koordinaatidele ja klõpsab hiire nuppu

MouseMove, %x%, %y% - tagasi algasendisse

Selleks, et kasutajal ei tekiks ebamugavusi, on skript seadistatud maksimaalsele kiirusele. See võimaldab käsklusi aktiveerida kohe pärast klahvide vajutamist.

Selle koodiga on seotud ka raskused. Kuna kasutajatel võivad olla erinevad monitoride mõõtmed, võib koordinaatsüsteemiga eksida. Selle probleemi lahendus on järgmine – iga kasutaja jaoks tehakse monitori koordinaatsüsteemi jaoks eraldi seadistus. Need andmed laaditakse kasutaja valimise aknasse ja kasutaja töötab neid kasutades tööseansi lõpuni. Vajadusel saab iga kasutaja soovitud laienduse sätteid muuta.

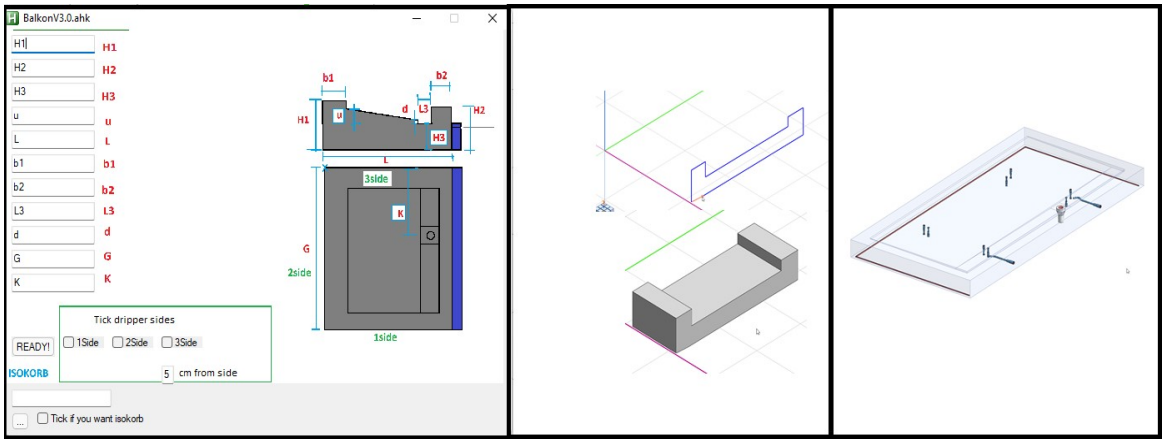
Rõdu moodul

Paraku pole Strakon programmis moodulit rõdude automaatseks modelleerimiseks. Uurides seda küsimust ja analüüsides saabuvate tellimuste rõdutüüpe, selgus, et Keskkonnaprojekt OÜ ettevõttes on umbes pooled tellitud rõdudest ristkülikukujulised

(ühe põhikaldega ja vee äravoolu kanalis ühe või kahe kaldega). Samuti ’nõlvade keeruka geomeetria tõttu võivad esineda vead objekti modelleerimisel programmi abil, mis omakorda ajab kasutajad segadusse. Alati pole selge, millisest elementaarkehast on vaja lõikeid teha, et rõdu edasine geomeetria oleks õige ja vea korral ei pea seda kõike uuesti modelleerima.

Selle probleemi lahendamiseks lisati nimekirja rõdu moodul Balcony, mis võimaldab eelnevalt sisestatud parameetrite järgi modelleerida rõdu korpust automaatselt. Vajadusel saab lisada külgedele tilgasooned ja tõsteankrud kaalu järgi.

Kui antud skript (vt Lisa 5) on aktiveeritud, avaneb aken (Joonis 3.2.2.4 a), kuhu kasutaja sisestab rõdu soovitud parameetrid. Geomeetria konstruktsioon ja koosteelementide valik tuleneb eelnevalt sisestatud parameetritest.



a.Rõdu mooduli aken b.Elemendi modelleerimine c. Lõpptulemus

Joonis 3.2.2.4 Rõdu moodul

Rõdu modelleerimisel võeti aluseks primitiivkujudest modelleerimine, mis teostatakse koordinaatsüsteemi abil. Selle skripti jaoks valiti profiili funktsioon, mis võimaldab joonistada keha ümbermõõtu koordinaatide järgi ja seejärel määrata sügavust. Rõdu geomeetria jaotati primitiivseteks komponentideks. Järgmise sammuna määrati primitiiv kehade paiknemise koordinaadid ja tõmmati kõik perimeetrid individuaalse sügavusega.

Küljed on valmistatud vastavalt Saksa tehaste soovidele, 1,5 cm kaldega, et elementi oleks võimalik raketist välja tõmmata. Need elemendid tehakse ka perimeetri joonistamise teel soovitud koordinaatidesse.

Detailid paigaldati importides koosteelemendid koordinaatide järgi, kasutades klahve *"CTRL+D*". Et arvutil oleks aega antud teelt hüpoteegifaili üles leida, kasutati

"Winwaitopen" koodi. See kood võimaldab peatada edasise modelleerimise, kuni avaneb failivaliku aken.

Drenaažitoru kõrgus valitakse lähtuvalt raketise kõrgusest äravoolupunktis ja LORO toodete saadaolevast kõrgusest. Äravoolu valimine toimub automaatselt raamatukogu kaudu. Koodi on kirjutatud funktsioon «IF», mis vastavalt seadistatud kriteeriumitele valib ülespoolse ümardamisega detaili tüübi.

Ankrud asetatakse elementi koordinaatide kaudu ja valitakse tulenevalt elemendi kaalust. Selleks, et valida õigeid ankruid, kirjutati skripti valem, mis arvutab elemendi kaalu vastavalt eelnevalt sisestatud parameetritele. Funktsiooni "IF" abil valitakse õige ankru tüüp ja asetatakse see teatud koordinaatidesse. Tulenevalt asjaolust, et koos ankru geomeetriaga modelleeriti täiendav armeering, ei ole ankrute edasine armeerimine vajalik.

Tilgasoon asetatakse külgedele, mis olid määratud esialgsete parameetrite järgi. Modelleerimine toimub perimeetri joonestamisel. Tilgasoone jaoks valiti negatiivne materjal, et geomeetria sobiks rõdu korpusega.

Selle koodi keerukus seisnes nõutavate koordinaatide matemaatilises arvutamises nõlvade ja kalde modelleerimisel. Seda tüüpi rõdude moodul sisaldab enam kui 1700 koodirida, mis muutis ka selle skripti kirjutamise keeruliseks. Inimfaktor oli suureks probleemiks kehade perimeetri joonistamisel. Andmete sisestamine Strakon programmi toimub klaviatuuri või hiire abil. Hiire liikumist peetakse prioriteediks ja see muudab kõik eelnevalt sisestatud punktiandmed. Hiire liikumishäirete vältimiseks skripti töötamise ajal kasutati koodi, mis blokeerib kasutaja klaviatuuri ja hiire sisendi kuni skripti lõpuni. Arvesse võeti ka muid skripti ebaõnnestumise võimalusi ja lisati kood, mis peatab kõik skriptid (klahv F11)

Trepi moodul

Strakon programmis on olemas valikuvõimalusega trepimoodul, kus tuleb määrata kõrgused, materjalid, kõik toed ja astmete kõrgused. Mõned kasutajad peavad seda keeruliseks ja ebapraktiliseks juhtudel, kui treppide geomeetria ei ole täielikult teada ja on vaja kohandada treppide toed platvormiga.

Töö hõlbustamiseks võeti kasutusele 3 treppide moodulit (Stairs1), (Stairs2), (Stairs3). Trepi moodul (vt Lisa 6) võimaldab kiiresti modelleerida sirge trepi korpust, kulutamata aega suurte tabelite täitmisele.

- (Stairs1) võimaldab modelleerida trepi korpust, teades eelnevalt astmete kõrgust.
- (Stairs2) Võimaldab modelleerida treppide korpust, teades eelnevalt, millisel kõrgusel trepp peaks asuma.
- (Stairs3) Võimaldab modelleerida trepi korpust konsoolidega ning ekspordib vajalikud vaadet 2d akna ja paigutab vajalikke mõõtjoone.

Treppide moodul töötab ka etteantud sügavusega kere perimeetri joonestamisel. Skripti käivitamisel ilmuvad aknad küsimustega trepi parameetrite kohta, mida kasutatakse trepi korpuse edasiseks modelleerimiseks.

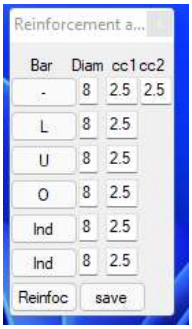
Treppide modelleerimiseks kirjutati kolm erinevat koodi. Esimest kasutatakse juhul, kui on teada iga astme kõrgus. Teine on treppide jaoks, mille alguse ja lõpu kõrgused on teada. Kõrgus teisel juhul arvutatakse koodis.

Kolmas skript modelleerib etteantud väärtuste järgi trepi ja tugede korpust. Pärast seda eksporditakse vaated Strakon 2D-moodulisse. Vaaded paigutatakse automaatselt lehe keskele määratud koordinaatidesse. Seejärel toimub mõõtmejoonte automaatne kinnitamine. Mõõtmejoonte seadmine sai võimalikuks tänu sellele, et Strakon võimaldab neid määrata punktide koordinaatide täpsustamisega. Samuti kõik kinnitatud mõõtmejooned on rangelt seotud geomeetriaga. See tähendab, et geomeetria edasise reguleerimise korral muutuvad ka kõik jooned õigeks.

Armeerimise abimoodul

Armeerimine Strakon programmis on intuitiivne ja arusaadav, kuid paraku nõuab varda läbimõõdu ja kuju valik iga kord vajaliku info sisestamist hüpikaknasse. Armeerimiselementide töö hõlbustamiseks on nimekirja lisatud abimoodul Reinforcement addon (vt Lisa 7). See võimaldab teil salvestada varraste läbimõõtu ja kuju, samuti arvestada varraste paigaldamisel ankurduspikkust. Abiaken peaks olema pidevalt nähtav ainult siis, kui kasutatakse modelleerimiseks 3D-akent ja seda ei varja teised programmid.

Armeerimise lisamoodulil on aken, kuhu saate salvestada sageli kasutatavate varraste seadistusi. Helpbox-i aken (Joonis 3.2.2.5) on alati paikneb Strakon 3D-akna peal. Seda saab teisaldada mis tahes mugavasse kohta. Aknas käsu aktiveerimisel lülitub sahtel armeerimisrežiimile, valib vajaliku varda, paneb maha läbimõõdud ja kaitsekihid. Kasutajal tuleb varda vaid õigesse kohta panna, säästes andmesisestuse aega. Skript automaatselt määrab ette ankurdamiseks vajalikud pikkused.



Joonis 3.2.2.5 Armeerimise abimoodul

Teksti moodul

Kuna Keskkonnaprojekt OÜ ettevõte töötab Saksamaa turul, siis on ettevõte probleemiks mõnede töötajate puudulik saksa keele oskus. Selle probleemi lahendamiseks on skriptide loendisse lisatud kiirtekstid (Fast type) ja tõlke moodul (Translate). Need moodulid aitavad säästa aega õige teksti otsimisel ja lühendada nende kirjutamiseks kuluvat aega. Moodulitel on emakeelse tekstiga aken, mis aitab täita nõutud väljad. Moodulid sisaldavad ka kiire teksti kirjutamise võimalust, kasutades selleks eelkonfigureeritud lühendeid. Tähemärkide kombinatsiooni kirjutamisel tunneb skript need ära ja asendab need soovitud saksakeelse tekstiga. Näiteks "S1" kirjutamisel kirjutatakse "Schnitt 1-1". Lühendite täieliku loendi leiate juhendist. Moodul sisaldab ka võimalust hõlpsasti tõlkida saksakeelse teksti Internetis.

Antud skript (vt Lisa 8) on kirjutatud saksakeelse teabe sisestamise hõlbustamiseks. Selle mooduli eelised seisnevad eelkõige kasutusmugavuses ja aitab vältida grammatilisi vigu skripti kirjutamisel. Tekst sisestatakse automaatselt kursori kohale. Redaktsioonid valitakse nutinuppu vajutades ja sobiva teksti valimisega. Tekst on kasutaja jaoks ette nähtud tema emakeeles.

Koosteelementide moodul

Strakon programmi liides teeb otsingu aeganõudvaks. Probleem muutub eriti aktuaalseks, kui koosteelementide loendist kasutatakse vaid mõnda tükki. Eriti problemaatiline on muutuva pikkusega detailid. Selle probleemi lahendamiseks lisati skriptide loendisse koosteelementide (Embedded Parts) (Anchors) (Embedded Parts1) moodulid. Need moodulid on suunatud koosteelementide otsimise ja paigutamise lihtsustamisele. Arvestades erinevate tehaste nõudeid, koostatakse iga tehase või projekti kohta mitu moodulit.

3D koosteelementide osade moodul (vt Lisa 9) võimaldab kasutajal vältida nimekirjast detailide pikki otsimisi. Tihti kasutatavad koosteelemendid saab kirja panna, vajutades nutikat nuppu ja valides soovitud tehast. Samuti sellel moodulil on eraldi hüpikaken ankrute ja tavaliselt kasutatava geomeetriaga elementide loendiga. Skripti olemus seisneb selles, et kutsutakse välja koosteelemendi otsingukasti ja sisestatakse kiirelt vajalik nimi.

2D koosteelementide moodul (vt Lisa 10) võimaldab lihtsustada nimede sisestamist. Menüüst tuleb valida õige detail ja sisestada selle pikkust. Järgmisena jääb ainult klikkiga paigaldada koosteelement õigesse kohta.

Mudeli ümberkujundamine

Kuna Keskkonnaprojekt OÜ ettevõte impordib 3D-objekte Strakon programmi, ei suuda Tekla mõnikord keerulise geomeetriaga elementidega toime tulla ja ekspordib katkised 3D-objektid. See probleem ei võimalda elementi renderdamisel õigesti joonestada. Selle probleemi lahendamiseks on skriptide loendisse lisatud moodul (ReModel). See moodul on mõeldud elemendi geomeetria ümberkujundamiseks, kasutades Strakoni programmi 3D-moodulis saadaolevat käsku Intersection mass.

Skripti (vt Lisa 11) põhimõtte on modelleerida suurt blokki ja kasutades Strakon-i käsu Intersection mass ja säilitada ainult plokiga lõikuva mudeli.

3.3 Skriptide rakendus ja mõju hinnang

Skriptide rakendamine toimus Keskkonnaprojekt OÜ serveris alates 09.09.2021. Skriptidega tutvustamine ja kasutusele võtmine toimus samal päeval. Abiprogrammi kasutamiseks paigaldati skriptimiskeele kompilaatorid tööarvutitele.

Töö käigus vaadati läbi 44 projekti, mis olid tehtud Keskkonnaprojekt OÜ ettevõttes ajavahemikul 27.03.2021 kuni 27.03.2022. Sihtmärgiks oli leida sarnase geomeetriaga ning vormistamisega elemente. 15 projekti sobisid ning sellest võeti arvesse 139 raudbetoonelemendi (99 treppelemendi ja 40 rõduelemendi). Kõige esimene skript oli kasutusele võetud 09.09.2021. Selle kuupäeva järgi olid elemendid jagatud kahte ossa (enne skriptide kasutamist ja pärast skriptide kasutamist).

Tabel 3.3.1.Tulemused: Elementide kogus ja ajakulu.

	Enne skriptide rakendamist	Pärast skriptide rakendamist
Rõduelementide arv	32	8
Ajakulu minutites	6394	1417
Keskmine ajakulu elemendi kohta minutites	199.82	177.13
Treppelementide arv	38	61
Ajakulu minutites	4930	6809
Keskmine ajakulu elemendi kohta minutites	129.74	111.62

Tabelist 3.3.1 johtub, et rõduelementide puhul ajakulu vähenes 11.35% võrra ning treppelementide puhul vähenes 13,97% võrra. Kuna pärast skriptide kasutamist sai arvesse võtta ainult 8 elemendi, ei pruugi rõduelementide arvutus olla statistiliselt esinduslik.

Lisaks analüüsitakse kogu projekti tulemuslikkust. Kokku vaadati 44 projekti. Neist 20 võeti arvesse, kuna osa neist oli välja töötatud Teklas või sisaldas muud tüüpi ülesannet.

Tabel 3.3.2 Tulemused: Ajakulud elementidele terve projekti kohta.

	Pärast skriptide kasutamis	Enne skriptide kasutamist
Projektide arv	8	12
Elementide arv	110	176
Ajakulu tundides	525.1	909
Keskmine ajakulu elemendi kohta	4.77	5.16

Vastavalt ajaraamile projektid jaotatakse kahte ossa, enne ja pärast skriptide rakendamist. Tabeli 3.3.2 järgi on näha, et 12 projekti tehti enne skriptide rakendamist ja 8 projekti tehti pärast skriptide rakendamist.

Kokku võeti arvesse 286 elemendi. Tabeli 5.2.4 järgi on näha, et enne skriptide kasutamist loodi 12 projektile 176 elemendi. See on keskmiselt 13,53 elemendi projekti kohta. Pärast skriptide kasutamist sai valmis 8 projekti 110 elemendiga kokku. See on keskmiselt 13,75 projekti kohta. Kuna 13.53 ja 13.75 on väga lähedased numbrid, siis võib eeldada, et projektide maht lõpptulemusi väga ei mõjutanud.

Lõplikud tulemused näitavad, et keskmine kulu elemendi kohta vähenes 0,39 tunni võrra (23,4 minuti võrra). Nende tulemuste põhjal võib eeldada, et projektile kuluv aeg on vähendanud 7,55% võrra. See tulemus erineb oluliselt varasematest näitajatest 11% ja 14%, seetõttu, et projekti ajakulu arvutamisel võeti arvesse projekti kontrollimise ja analüüsimise aega.

Tabel 3.3.3 Skriptide mõju tööprotsessile.

Abiprogrammid	Rõdu			Trepielement		
	M	A	V	M	A	V
Kiirklahvid	*	*	*	*	*	*
Kausta süsteem ja seaded	*	*	*	*	*	*
Rõdu elementide moodul	*	*				
Trepi elementide moodulid				*		*
Armeerimise abimoodul		*			*	
Keele moodul			*			*
Koosteelementide moodulid	*			*		
Indeksite moodulid			*			*

M-Mudel, A-Armeering, V- vormistamine

Tabeli 3.3.3 järgi on näha, millised on skriptide mõjud tööprotsessile. Kõige rohkem mõju on Kiirklahvide ja Kausta süsteemi skriptidel. Võttes arvesse, et antud skriptid olid rakendatud kõige esimesena, võib eeldada, et suurim osa ajakokkuhoiust on tulnud just tarkvara ergonoomika optimeerimisest. Uuringu käigus vähem kasutati rõdu ja trepi moodulid, sest neid rakendati viimasena. Samas on positiivne see, et isegi täismahus skriptide kasutamiseta õnnestus ajakulu vähendada 11% võrra rõduelementide jaoks ja 14% võrra treppelementide jaoks.

Aasta jooksul olid tehtud 99 treppelemendi, mis sobisid antud töö analüüsiks. Võrreldes keskmiseid ajakulusid võib eeldada, et skriptide kasutamine võiks kokku hoida 30 tundi aastas ainult konkreetsete ülesannete täitmiseks. Arvutades analüüsis vaadeldavate projektidele ajakulud võib eeldada, et loodud skriptide kasutamine võiks kokku hoida vähemalt 108 tundi aastas.

3.4 Lühikokkuvõte

Töö käigus õnnestus Strakon programmi jaoks arendada abimooduleid töö optimeerimiseks ja automatiseerimiseks. Skriptide kirjutamisel ei õnnestunud luua moodulit, mis võimaldaks tõlkida teksti brauserit avamata. Kaaluti võimalust tõlkida teksti brauseri taustal, kuid paraku pole see võimalik erinevate brauseritele kasutamisel ja kulutab palju operatsioonimälu. Rakendatud skriptide versioonid ei ole lõplikud, kuid neid on võimalik lihtsustada ja nad vajavad edasiarendamist.

Uuringu tulemusena õnnestus Strakon tarkvaras monteeritavate raudbetoonelementidega tööaega vähendada 11% võrra rõduelementide puhul ja 14% treppelementide puhul arvestades, et kõiki skripte ei kasutatud täismahus. Õnnestus vähendada projektidele kuluva aega 7,53% võrra, mis omakorda kinnitab skriptide positiivset mõju .Täpsemad andmed nõuavad analüüsimiseks rohkem tööaega ja projekte.

Töö tulemusena õnnestus luua ja rakendada praktikas 137 skripti. Kõige mahukamaks osutus rõdu moodul, mis sisaldab enam kui 1700 koodirida. Igal skriptil on oma funktsionaalsus ja tänu ühisele struktuurile on nende kasutamine andnud positiivse tulemuse.

4. ARUTELU

Selles töös kirjeldati ebatraditsioonilist meetodit BIM-tööriista töö automatiseerimiseks ja optimeerimiseks.

Kirjanduse ülevaates kirjeldati töö automatiseerimise meetodeid, kasutades skripte sisemise interaktsiooni tööpõhimõtteid. Selle funktsiooni jaoks BIM programmil peab olema avatud lähtekood, mida saab lugeda ja muuta kasutades programmeerimiskeeli.

Selles töös kasutati automatiseerimismeetodit programmi jaoks, kus avatud juurdepääs koodile puudub. Kahjuks ei pakuta nende programmidele programmeerimiskeeli automatiseerimist. Selle probleemi lahendamiseks loodi abiprogrammi skriptikeeles, mis võimaldab tööd automatiseerida ja seadistada vajaliku tööriista iga kasutaja jaoks. Loodud abiprogrammi tööpõhimõte seisneb algandmete sisestamisel programmi aknasse ja tulemuse väljastamisel kursori ja klaviatuuri manipuleerides, justkui inimene teeks seda ise.

See meetod on rakendatav kõikidele BIM-tööriistadele, kuna seal ei esine erinõudeid. Pealegi pole Autohotkey ainuke skriptikeel, mis võib anda sarnase tulemuse. Keelevalik sõltub otseselt programmi välja töötava inimese oskustest ja eelistustest.

Ainus takistus selle meetodiga automatiseerimisel on iga BIM-tööriista unikaalsus. Iga BIM-tööriist vajab oma lähenemist, kuna neil on erinevad liidesed ja funktsioonid, mistõttu ei saa selle töö skripte kasutada teiste rakenduste jaoks.

Selles töös kirjeldati skripte rakendamist praktikas ning saadud tulemusi. Töö kaigus õnnestus automatiseerida raudbetoonist monteeritavate treppide ja rõdude modelleerimist, samuti parandada programmi Strakon ergonoomikat. Rakendatud skriptide versioonid ei ole lõplikud, kuid neid on võimalik lihtsustada ja nad vajavad edasiarendamist. Täpsemad andmed ajakokkuhoiu hindamiseks nõuavad analüüsimiseks rohkem tööaega ja projekte.

KOKKUVÕTE EESTI KEELES

Lõputöö eesmärgiks oli hinnata BIM-i automatiseerimise parimaid teadmisi ja praktikaid tootmise täpsusega mudelite ja jooniste koostamiseks ning välja töötada eesmärgi saavutamiseks vajalik raamistik ning tööprotsessis kasutatav mudel.

Selleks oli vaja töötada välja praktilised skriptid ja testida nende tõhusust ettevõtte kontekstis. Uuringu objektiks on Saksamaa turule ettenähtud monteeritavate raudbetoonelementide projekteerimise tehnoloogiline tsükkel. Uuringus kirjeldati töö optimeerimist Strakon programmi näitel Autohotkey programmeerimiskeele abil. Töö oli suunatud projekteerimisbüroo Keskkonnaprojekt OÜ töö optimeerimisele konkreetsete nõuete järgi.

Selle eesmärgi saavutamiseks otsiti vastuseid järgmistele küsimustele:

- Millised on Saksamaa raudbetoonelementide tootmise ja tehaste eripärad?
- Millised BIM modelleerimise programmid on tänapäeval kättesaadavad ja millised võimalused tootmiseks vajaliku täpsusega mudelite ja jooniste loomise automatiseerimiseks?
- Millised aspektid takistavad tööd valitud instrumentiga?
- Kuidas saab BIM tööriista automatiseerida ja milline on selle kasu?

Uuringu käigus tehti punktis 1.1 ülevaade betoonelementelementide automatiseeritud tootmisprotsessist saksa tehastes. Uuringu käigus tuvastati selleks vajalikud failivormingud: XML, BVB, SWP, DXF, DWG, PDF ja IFC.

Punktis 1.2 uuriti milline BIM tarkvara vastab saksa tehaste nõuetele ning käsitleti nende automatiseerimis võimalusi. Vaadeldavatest BIM tarkvaradest selgus, et saksa nõuetele vastavad Tekla, Revit ja Strakon. Töö käigus selgus, et automatiseerimine BIM tarkvaras toimub programmeerimiskeelte sisese interaktsioon läbi avatud lähtekoodi.

Punktis 1.3 uuriti programmeerimiskeeli, millega automatiseerimisprotsessi BIM tarkvarades läbi viiakse. Samuti punktis 1.3.1 kirjeldati alternatiivmeetodit töö automatiseerimiseks BIM tarkvaras, juhul kui puudub avatud lähtekood. See meetod on rakendatav välissuhtluses BIM tarkvaraga skriptide abil.

Peatükkis 2 kirjeldati lähtematerjalid ja metoodikat, mille järgi analüüsiti BIM tarkvara ning loodi vajalikke skripte. Peatükis on kirjeldatud skriptide rakendamine praktikas ning saadud tulemuste analüüs. Punktis 2.1 on põhjendatud Strakon tarkvara ja Autohotkey skriptimiskeele valik uuringu jaoks.

Peatükkis 3 kirjeldati saadud tulemusi. Töö käigus õnnestus läbi viia BIM-tööriista Strakon hinnangu ning välja selgitada tarkvara nõrkusi. Strakon programmi tuvastatud puuduste alusel tehti ettepanekud ja loodi skripte Autohotkey skriptimiskeele abil. Loodud skriptid kirjeldati punktides 3.2.1 ja 3.2.2.

Punktis 3.3 kirjeldati skripte rakendamist praktikas ning saadud tulemusi. Töö kaigus õnnestus automatiseerida raudbetoonist monteeritavate treppide ja rõdude modelleerimist, samuti parandada programmi STRAKON ergonoomikat. Uuringu tulemusena õnnestus rõduelementidele kuluvat tööaega vähendada 11% võrra ja treppelementide puhul 14% võrra. Keskmist projektile kuluvat aega arvutades selgus, et aega kulutati keskmiselt 7,5% võrra vähem, mis omakorda kinnitab skriptide positiivset mõju tööprotsessile.

Lõputöös kirjeldati ebatraditsioonilist automatiseerimise meetodit, kasutades klaviatuuri ja hiire manipuleerimist skriptide abil. Need skriptid ei ole lõplik versioon ja nõuavad edasiarendamist. Samuti on see meetod rakendatav teistes BIM-tööriistades.

KOKKUVÕTE INGLISE KEELES

The aim of the thesis was to evaluate the best knowledge and practices of BIM automation for the precast concrete models and workshop drawings, and to develop a framework and process model for the automation of production models and drawings for the factory. Aim was to develop practical scripts and test their effectiveness in a business context. The object of the study is the technological cycle of the precast concrete elements production on the German market. The study described the optimization of the work with the example of the Strakon program using the Autohotkey programming language. The work was aimed at optimizing the worklow of the design bureau Keskkonnaprojekt OÜ according to specific requirements.

To achieve this goal, answers were sought to the following questions:

- What are the German factories specifics for the production of precast concrete elements?
- What BIM modeling programs are available today and what are the ways to automate the creation of models and drawings with the precision needed for production?
- What aspects prevent from working with the chosen application?
- How can the BIM tool be automated and what are the benefits?

In the course of the study, an overview of the automated production process of precast concrete elements in German factories was provided in section 1.1. The study identified the necessary file formats: XML, BVB, SWP, DXF, DWG, PDF and IFC.

Section 1.2 examined which BIM software meets the requirements of German factories and their automation possibilities. The BIM software in question showed that Tekla, Revit and Strakon met German requirements. In the course of the work, it was found that automation in BIM software takes place through the interaction of programming languages using open source.

Section 1.3 investigated the programming languages used to perform the automation process in BIM software. Section 1.3.1 also described an alternative method for automating work in BIM software that does not have open source. This method is external communication with BIM scripts.

Chapter 2 described the source materials and the methodology according to which the BIM software was analyzed and the necessary scripts were created. The chapter

describes the implementation of the scripts in practice and the method of analyzing the results. Section 2.1 explains the choice of Strakon software and Autohotkey scripting language for the study.

Chapter 3 described the results obtained. In the course of the work, the evaluation of the BIM tool Strakon was carried out, and based on the criticism and wishes of the company's employees, the weaknesses of the software were identified. Based on the shortcomings identified by the Strakon program, suggestions were made and scripts were created using the Autohotkey scripting language. The generated scripts were described in sections 3.2.1 and 3.2.2.

Section 3.3 describes the implementation of the scripts in practice and its results. During the work, the modeling of reinforced concrete stairs and balconies was automated, as well as the ergonomics of the STRAKON program were improved. As a result of the study, the working time for balcony elements was reduced by 11% and for stair elements by 14%. Calculating the average time spent on a project, it was found that time was spent on average 7.5% less, which in turn confirms the positive impact of scripts on the work process.

The thesis described non-traditional automation method using mouse and keyboard manipulation throw scripts. These scripts are not the final version and are subject to further development. This method is also applicable to other BIM tools, as it has no special requirements.

KASUTATUD KIRJANDUS

[1] Schöck (28.02.2020)"Isokorb® - Schöck Bauteile GmbH."[WWW] https://www.schoeck.com/en/isokorb

[2] HALFEN (2022) "HALFEN HIT Insulated Connection." [WWW] https://www.halfen.com/uk/769/product-ranges/construction/reinforcement-systems/halfen-hit-insulated-connection/introduction/

[3] Thomas Rraefab (2022) "Thomas Rraefab" [WWW] https://www.thomas-praefab.de/video/

[4] Indeed (23.02.2021)."What is an XML." [WWW] https://www.indeed.com/career-advice/career-development/xml-file

[5] LAP (2022) "Laser Projection, Laser Measurement and QA." [WWW] https://www.lap-laser.com/

[6] Trimble (2019)."BVBS | Tekla User Assistance." [WWW] https://support.tekla.com/doc/tekla-structures/2019/int_bvbs_overview

[7] Trimble (06.04.2022) "What is BIM?" [WWW] https://constructible.trimble.com/construction-industry/what-is-bim-building-information-modeling

[8] " Autodesk (2022) "What Is BIM | Building Information Modeling | Autodesk." [WWW] https://www.autodesk.com/industry/aec/bim

[9] K. Daria (2020) "Kosavchenko Daria BIM Geometry Creation from Point Clouds"
LAB University of Applied Sciences, Civil and Construction engineering, Lappeenranta

[10] Joonas Helminen (06.2019) "Automated generation of steel connections of BIM by machine learning."
[WWW] Tampere University https://core.ac.uk/reader/250167800

[11] Chuck Eastman, Paul Teicholz, Rafael Sacks, Ghang Lee (14.08.2014)"BIM Handbook : A Guide to Building Information Modeling for Owners, Designers, Engineers, Contractors, and Facility Managers." Publisher: John Wiley & Sons [WWW]

https://ebookcentral.proquest.com/lib/tuee/reader.action?docID=5447327

[12] N. v Divin (2020) " BIM by using Revit API and Dynamo. A review" Peter the Great St. Petersburg Polytechnic University, St. Petersburg, Russian Federation ,AlfaBuild

[13] Autodesk (07.12.2020) "About the Revit SDK." [WWW] https://knowledge.autodesk.com/support/revit/learn-explore/caas/CloudHelp/cloudhelp/2021/ENU/Revit-Customize/files/GUID-D7E8694D-7DB3-41CA-A0F3-AF64DC2DA015-htm.html

[14] Autodesk (18.04.2014).“Supported Programming Languages , Revit .” [WWW] https://knowledge.autodesk.com/support/revit/learn-explore/caas/CloudHelp/cloudhelp/2014/ENU/Revit/files/GUID-FEF0ED40-8658-4C69-934D-7F83FB5D5B63-htm.html

[15] FileInfo (2022)“DLL File Extension” [WWW] https://fileinfo.com/extension/dll

[16] L. Huang (08.2018) “Revit Plugins for Electrical Engineering Improvements in Buildings: Lighting Power Density and Electrical Equipment Placement” School of Architecture, University of Southern California

[17] Autodesk (2022) “Beyond Dynamo: The Powerful Automation Potential of Forge and the Revit API” [WWW] https://www.autodesk.com/autodesk-university/article/Beyond-Dynamo-Powerful-Automation-Potential-Forge-and-Revit-API-2022

[18] M. Koshelev (2021) “Tekla Structures Custom Components and Open API in the design of precast elements ” LAB University of Applied Sciences

[19] Trimble (18.03.2020).“How Tekla Open API works | Tekla Developer Center.” [WWW] https://developer.tekla.com/tekla-structures/documentation/how-tekla-open-api-works

[20] Trimble (13.05.2019). “5 ways to enhance and extend Tekla Structures.” [WWW] https://developer.tekla.com/tekla-structures/documentation/5-ways-enhance-and-extend-tekla-structures

[21] Autodesk (2010) “AUTOCAD .Net C#”, [WWW] http://docs.autodesk.com/ACD/2010/ENU/AutoCAD%20.NET%20Developer%27s%20Guide/index.html

[22] Mahesh Chandra Luintel (18.04.2019) “ USE OF SCRIPT FILES IN AUTOCAD.” [WWW] https://www.researchgate.net/publication/332493976_USE_OF_SCRIPT_FILES_IN_AUTOCAD

[23] Татьяна Климачева (2007) “Трехмерная компьютерная графика и автоматизация проектирования на VBA в AutoCAD” ДМК Пресс [WWW] https://books.google.ee/books?hl=ru&lr=&id=uQfRAAAAQBAJ&oi=fnd&pg=PA9&dq=autocad+%D0%B0%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F&ots=TpYxX0Z0K8&sig=55G2cxdz7OHLICZjzhjZ2Udv1xA&redir_esc=y#v=onepage&q=autocad%20%D0%B0%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F&f=false

[24] DICAD Systeme GmbH (2022) “2D/3D/BIM CAD Systems for Structural Engineering and Precast Construction.” [WWW] https://www.strakon.com/products/products.html

[25] ФГБОУ ВО "АГПУ" (2021) "Современные информационно-коммуникационные технологии" [WWW] http://www.agpu.net/modernIT/Archive/Volume8_2021/8.pdf#page=19

[26] Bimlab (2022) "Программирование для BIM: Revit API, C#, Python, Forge." [WWW] https://bimlab.ru/bim-programming.html

[27] Mahesh Chand (07.03.2020) "What Is C#."[WWW] https://www.c-sharpcorner.com/article/what-is-c-sharp/ (07.03.2020).

[28] Luis Gillman (2018) "Is C# a scripting language or other type of language?." [WWW] https://www.compsuccess.com/is-c-a-scripting-language/

[29] Эндрю Троелсен, Филипп Джепикс (2019)."Язык программирования C# 7 и платформы .NET и .NET Core." Гарнитура Times [WWW] https://books.google.ee/books?hl=ru&lr=&id=wDrBDwAAQBAJ&oi=fnd&pg=PA34&dq=%D0%A1+sharp&ots=ofOkqCgfuk&sig=WOb7g6s59CMytdsCDHKPD6N3QaA&redir_esc=y#v=onepage&q&f=true

[30] Ivan Bondarenko (2018) "Ruby для начинающих: чем интересен этот язык и как его эффективно изучать." [WWW] https://dou.ua/lenta/articles/ruby-for-beginners/

[31] Ruby-lang (2022) "О Ruby." [WWW] https://www.ruby-lang.org/ru/about/

[32] Хэл Фултон (2007) "Программирование на языке Ruby."ДМК Пресс [WWW] https://books.google.ee/books?hl=ru&lr=&id=hi-bAgAAQBAJ&oi=fnd&pg=PA12&dq=ruby+%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5+&ots=shoYJC00Di&sig=BzebKsTcaoOT4Yz-CXDoKmfq2Qg&redir_esc=y#v=onepage&q=ruby%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&f=false

[33] Learnhowtoprogram (23.03.2022) "Basic Ruby Scripting." [WWW] https://www.learnhowtoprogram.com/ruby-and-rails/basic-ruby/basic-ruby-scripting

[34] Autodesk (04.02.2022) "Upgrade Information, Revit 2022 ." [WWW] https://knowledge.autodesk.com/support/revit/learn-explore/caas/CloudHelp/cloudhelp/2022/ENU/Revit-WhatsNew/files/GUID-720338F0-4495-49FF-A6E0-418AA1A3876B-htm.html

[35] Mchost (07.12.2020) "Что такое Python и для чего нужен этот язык "[WWW] https://mchost.ru/articles/chto-takoe-python/

[36] С. И. В. Сысоева М. В. (2018)., "Программирование для «нормальных» с нуля на языке Python," [WWW] https://lib.samtuit.uz/uploads/files/61e52d532d3477.07081707.pdf

[37] Пол Бэрри (2017) “Изучаем программирование на Python .” Издательство “Э” [WWW] https://books.google.ee/books?hl=ru&lr=&id=rs81DwAAQBAJ&oi=fnd&pg=PA5&dq=python+%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&ots=xjKGlIsphS&sig=I7aUSPzPu8ebE1bYl80a_ky2Kak&redir_esc=y#v=onepage&q=python%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&f=false

[38] Autohotkey (2022)“AutoHotkey.” [WWW] https://www.autohotkey.com/.

[39] Asana (2022) “Asana.” [WWW] https://app.asana.com/0/home/1199943173474911

[40] Everhour (2022)“Everhour” [WWW] https://everhour.com/

[41] Documentation (2022) “Console RAR SFX module (WinCon.SFX)” [WWW] https://documentation.help/WinRAR/HELPConsoleSFX.htm

LISAD

Lisa-1 Menüü akna skript

```
#NoEnv
#Warn
SendMode Input
SetWorkingDir %A_ScriptDir%  ;
#SingleInstance, Off
#NoTrayIcon
Gui, Add, Button, x20 y50 w50 gMH, MH
Gui, Add, Button, x20 y80 w50 gAN, AN
Gui, Add, Button, x80 y50 w50 gDK, DK
Gui, Add, Button, x80 y80 w50 gDT, DT
Gui, Add, Button, x140 y50 w50 gIK, IK
Gui, Add, Button, x140 y80 w50 gIS, IS
Gui, Add, Button, x200 y50 w50 gJM, JM
Gui, Add, Button, x200 y80 w50 gMD, MD
Gui, Add, Button, x260 y50 w50 gMM, MM
Gui, Add, Button, x260 y80 w50 gNT, NT
Gui, Add, Button, x320 y50 w50 gVD, VD
Gui, Font, s11 cBlue, Verdana
Gui, Add, Text, x380 y150, Created by AN.
Gui, Font, s11 cBlue, Verdana
Gui, Add, Text, x400 y170, V 2.0.
Gui, Font, s11 cBlue, Verdana
Gui, Add, Text, x10 y15, Choose your profile.
Gui, Show, w500 h200, Profile
Gui, Add, Picture, x0 y0 w500 h200 +0x4000000, „Desired Path“
Return
AN:

{

Run, „Desired Path“
Run, „Desired Path“
Exitapp
GuiClose:
Exitapp
Return
```

Lisa 2 Kausta süsteemi skript (baas)

```
CreateMenu:
Menu, %rootdir%, add, Open %rootdir%, OpenFolder
ExtractIcon(rootdir, "Open " . rootdir)
Loop, %rootdir%\*.*, 2, 1 ; Folders only
{
StringGetPos, pos, A_LoopFileLongPath, \, R
if A_LoopFileAttrib contains H,R,S
continue
if (pos <> -1)
StringLeft, ParentFolderDirectory, A_LoopFileLongPath, %pos%
if (pos = -1)
ParentFolderDirectory := rootdir
Menu, %A_LoopFileLongPath%, add, Open %A_LoopFileName%, OpenFolder
ExtractIcon(A_LoopFileLongPath, "Open " . A_LoopFileName)
Menu, %ParentFolderDirectory%, add, %A_LoopFileName%, :%A_LoopFileLongPath%
ExtractIcon(ParentFolderDirectory, A_LoopFileName)
}
Loop, %rootdir%\*.*, 0, 1 ; Files only
{
;msgbox % A_LoopFileLongPath
if A_LoopFileAttrib contains H,R,S
continue
StringGetPos, pos, A_LoopFileLongPath, \, R
if (pos <> -1)
StringLeft, ParentFolderDirectory, A_LoopFileLongPath, %pos%
if (pos = -1)
ParentFolderDirectory := rootdir
Menu, %ParentFolderDirectory%, add, %A_LoopFileName%, OpenFile
ExtractIcon(ParentFolderDirectory, A_LoopFileName)
}
Return
```

Lisa-3 Indeksite skript

```
#SingleInstance, Off
#NoTrayIcon
#NoEnv
SetDefaultMouseSpeed, 1
SendMode Input
#NoEnv
SetTitleMatchMode, 2
SetWorkingDir %A_ScriptDir%
Sendinput,{ESC}
sleep 200
Sendinput,
!f{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Do
wn}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{ENTER}
WinWaitActive, Drawing data
sleep 300
;info
FormatTime, CurrentDateTime,, dd.MM.yyyy
Sendinput, {TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}Uhlir
Sendinput, {TAB}{TAB}{TAB}%CurrentDateTime%
Sendinput, {TAB}Freigabe
Sendinput,
{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{T
AB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB
}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}{TAB}a
Sendinput, {TAB}{TAB}{TAB}{TAB}{TAB}A
Exitapp
```

Lisa 4 Kiirklahvide skript

```
#SingleInstance, Off
#NoTrayIcon
#NoEnv
#MaxHotkeysPerInterval 99000000
#HotkeyInterval 99000000
#KeyHistory 0
ListLines Off
Process, Priority, , A
SetBatchLines, -1
SetKeyDelay, -1, -1
SetMouseDelay, -1
SetDefaultMouseSpeed, 0
SetWinDelay, -1
SetControlDelay, -1
SendMode Input

MouseGetPos x, y
SendInput {Click 1156 55}
MouseMove %x%, %y%
Return
```

Lisa-5 Rõdu mooduli skript (lühendatud)

```
SetTitleMatchMode, 2
#SingleInstance Force
SetWorkingDir %A_ScriptDir%
if not A_IsAdmin
Run *RunAs "%A_ScriptFullPath%"
#NoEnv  ; Recommended for performance and compatibility with future AutoHotkey releases.
; #Warn  ; Enable warnings to assist with detecting common errors.
SendMode Input  ; Recommended for new scripts due to its superior speed and reliability.
SetWorkingDir %A_ScriptDir%  ; Ensures a consistent starting directory.
SendMode, Event
Gui, Add, Edit, vOutputVarH1 w100, H1 ;Result input in variable "Input"
Gui, Add, Edit, vOutputVarH2 w100, H2 ;Result input in variable "Input"
Gui, Add, Edit, vOutputVarH3 w100, H3 ;Result input in variable "Input"
Gui, Add, Edit, vOutputVaru w100, u ;Result input in variable "Input"
Gui, Add, Edit, vOutputVarL w100, L ;Result input in variable "Input"
Gui, Add, Edit, vOutputVarb1 w100, b1 ;Result input in variable "Input"
Gui, Add, Edit, vOutputVarb2 w100, b2 ;Result input in variable "Input"
Gui, Add, Edit, vOutputVarL3 w100, L3 ;Result input in variable "Input"
Gui, Add, Edit, vOutputVard w100, d ;Result input in variable "Input"
Gui, Add, Edit, vOutputVarG w100, G ;Result input in variable "Input"
Gui, Add, Edit, vOutputVarK w100, K ;Result input in variable "Input"
Gui, Add, Button, 100 y+50 gPerform, READY! ;Will launch "Perform" routine
Gui, Add, checkbox, x+10 vKante1, 1Side
Gui, Add, checkbox, x+10 vKante2, 2Side
Gui, Add, checkbox, x+10 vKante3, 3Side
Gui, Add, Edit, vOutputVarKante y+20  , 5
Gui, Add, Edit  , x10 200 h20 vInputfile
Gui, Add, Button, x10 220 h20 vSource gSelectFile, ...
Gui, Add, checkbox, x+10 vISOKORB, Tick if you want isokorb
Gui, Add, Picture, x0 y0 w600 h400 +0x4000000, C:\Users\alber\Desktop\Scripts-all\balkon\balkon.png
Gui, Show
return
SelectFile:
  FileSelectFile, SelFile,3 , %A_WorkingDir%, Select a SRT File, Subtitles (*.ifc;*.Ini)
  IfEqual,SelFile,, Return
  SplitPath, SelFile,, SelDir
  GuiControl,,Inputfile,%SelFile%
  FileRead, Text, %SelFile%
  GuiControl, ,EText, %Text%
Return
Perform:
Gui, Submit,  ; Passes completed text into variable
BlockInput, On
SendInput, C
sleep 300
SendInput, {ESC}{ESC}
```

```
SendInput, {Click 1000 500}
sleep 200
; ((OutputVarL*OutputVarH3+(OutputVarH2-
OutputVarH3)*OutputVarb2+(OutputVarH1-
OutputVarH3)*OutputVarb1+OutputVaru*(OutputVarL-OutputVarb1-OutputVarb2-
OutputVarL3)-(OutputVaru-OutputVard)*(OutputVarL-OutputVarb1-
OutputVarb2)/2)*OutputVarG+2*(OutputVarL-OutputVarb1-
OutputVarb2)*OutputVarb1*(OutputVarH1-OutputVarH3))/1000000*2.5

VES := ((OutputVarL*OutputVarH3+(OutputVarH2-
OutputVarH3)*OutputVarb2+(OutputVarH1-
OutputVarH3)*OutputVarb1+OutputVaru*(OutputVarL-OutputVarb1-OutputVarb2-
OutputVarL3)-(OutputVaru-OutputVard)*(OutputVarL-OutputVarb1-
OutputVarb2)/2)*OutputVarG+2*(OutputVarL-OutputVarb1-
OutputVarb2)*OutputVarb1*(OutputVarH1-OutputVarH3))/1000000*2.5
Sendinput {ESC}
Sendinput D
Sendinput {F6}
Sendinput 0
Sendinput {F8}
Sendinput %OutputvarG%
Sendinput {Enter}
Sleep 50
Sendinput 0,0,0
Sendinput {Enter}
sleep 50
Sendinput, 0,
Sendinput, 0,
Sendinput,  %OutputVarH1%
Sendinput, {Enter}
sleep 50
Sendinput, % OutputVarb1 - 1.5 ;X
Sendinput, ,0,  ;Y
Sendinput, 0   ;Z
Sendinput, {Enter}
sleep 50
Sendinput, 1.5,  ;X
Sendinput, 0,-  ;Y
Sendinput % OutputVarH1 - OutputVarH3 - OutputVaru   ;Z
Sendinput, {Enter}
sleep 50
Sendinput % OutputVarL - OutputVarb1 - OutputVarb2 - OutputVarL3   ;X
Sendinput, ,0,-  ;Y
Sendinput % OutputVaru  - OutputVard   ;Z
Sendinput, {Enter}
sleep 50
Sendinput, 0   ;X
Sendinput, ,0,-  ;Y
Sendinput %OutputVard%  ;Z
Sendinput, {Enter}
sleep 50
```

```
Sendinput, %OutputVarL3%  ;X
Sendinput, ,0,  ;Y
Sendinput, 0  ;Z
Sendinput, {Enter}
sleep 50
Sendinput, 1.5  ;X
Sendinput, ,0,  ;Y
Sendinput % OutputVarH2 - OutputVarH3    ;Z
Sendinput, {Enter}
sleep 50
Sendinput, % OutputVarb2 - 1.5   ;X
Sendinput, ,0,  ;Y
Sendinput, 0   ;Z
Sendinput, {Enter}
sleep 50
Sendinput, 0,  ;X
Sendinput, 0,-  ;Y
Sendinput, %OutputVarH2%   ;Z
Sendinput, {Enter}
sleep 50
  ; ENDPOINT
Sendinput {Enter}
sleep 50
Sendinput {Enter}
; GEOMETRY ok
Sendinput {ESC}
Sendinput D
Sendinput {F6}
Sendinput 0
Sendinput {F8}
Sendinput %OutputvarL%
Sendinput {Enter}
Sleep 50
Sendinput %OutputvarL%,0,0
Sendinput {Enter}
sleep 50
Sendinput, 0, ;X
Sendinput, 0,  ;Y
Sendinput, %OutputVarH1%  ;Z
Sendinput, {Enter}
sleep 50
Sendinput, 0,- ;X
Sendinput, % OutputVarb1 - 1.5  ;Y
Sendinput, ,0  ;Z
Sendinput, {Enter}
sleep 50
Sendinput, 0, ;X
Sendinput, -1.5,-  ;Y
Sendinput, % OutputVarH1 - OutputVarH3  ;Z
Sendinput, {Enter}
sleep 50
```

```
Sendinput, 0, ;X
Sendinput, 0,-  ;Y
Sendinput, %OutputVarH3%  ;Z
Sendinput, {Enter}
sleep 50
 ; ENDPOINT
Sendinput {Enter}{Enter}
sleep 50
Sendinput {Enter}
;BORT 1 OK
Sleep 50
Sendinput {ESC}
Sendinput D
Sendinput {F6}
Sendinput 0
Sendinput {F8}
Sendinput %OutputvarL%
Sendinput {Enter}
Sleep 50
Sendinput %OutputvarL%
Sendinput, ,-
Sendinput % OutputvarG - Outputvarb1
Sendinput, ,0
Sendinput {Enter}
sleep 50
Sendinput, 0, ;X
Sendinput, 0,  ;Y
Sendinput, %OutputVarH3%  ;Z
Sendinput, {Enter}
sleep 50
Sendinput, 0, ;X
Sendinput, -1.5,  ;Y
Sendinput, % OutputVarH1 - OutputVarH3   ;Z
Sendinput, {Enter}
sleep 50
Sendinput, 0,- ;X
Sendinput, % OutputVarb1 - 1.5  ;Y
Sendinput, ,0  ;Z
Sendinput, {Enter}
sleep 50
Sendinput, 0, ;X
Sendinput, 0,-  ;Y
Sendinput, %OutputVarH1%  ;Z
Sendinput, {Enter}
sleep 50
 ; ENDPOINT
Sendinput {Enter}{Enter}
sleep 50
Sendinput {Enter}
;BORT 2 OK
Sendinput {ESC}
```

```
Sendinput D
Sendinput {F6}
Sendinput 0
Sendinput {F8}
Sendinput % OutputvarL3 + 5
Sendinput {Enter}
Sleep 50
Sendinput % OutputvarL - Outputvarb2 - OutputvarL3  ;X
Sendinput, ,-
Sendinput % Outputvarb1 - 2
Sendinput, , ;Y
Sendinput % OutputvarH3 + OutputVard  ;Z
Sendinput {Enter}
sleep 50
Sendinput, 0, ;X
Sendinput, 0,-  ;Y
Sendinput, %OutputVard%  ;Z
Sendinput, {Enter}
sleep 50
Sendinput, 0,- ;X
Sendinput, % OutputVarK - OutputVarb1 - OutputVarL3 / 2 + 2  ;Y
Sendinput, %OutputVarH1%  ;Z
Sendinput, {Enter}
sleep 50
 ; ENDPOINT
Sendinput {Enter}{Enter}
sleep 50
Sendinput {Enter}
;sliv 1 qOK
Sendinput {ESC}
Sendinput D
Sendinput {F6}
Sendinput 0
Sendinput {F8}
Sendinput % OutputvarL3 + 5
Sendinput {Enter}
Sleep 50
Sendinput % OutputvarL - Outputvarb2 + 5  ;X
Sendinput, ,-
Sendinput % OutputvarG - Outputvarb1 + 2   ;Y
Sendinput, ,
Sendinput % OutputvarH3 + OutputVard  ;Z
Sendinput {Enter}
sleep 50
Sendinput, 0, ;X
Sendinput, 0,-  ;Y
Sendinput, %OutputVard%  ;Z
Sendinput, {Enter}
sleep 50
Sendinput, 0, ;X
Sendinput, % OutputVarG - OutputVarK - OutputVarb1- OutputVarL3 / 2 + 2  ;Y
```

```
Sendinput, %OutputVarH1%  ;Z
Sendinput, {Enter}
sleep 50
 ; ENDPOINT
Sendinput {Enter}{Enter}
sleep 50
Sendinput {Enter}
;sliv 2 qOK
Sleep, 400
;ANKERA
If VES between 0 and 1.4
{
;RD12
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\PHILIPP
Schraubanker RD12.smd
SendInput, {Enter}
sleep 300
;3anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4  ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
Sendinput, xx
;3anker
SendInput, {F2}
```

```
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 - 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 - 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4  ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
Sendinput, {ESC}
}
If VES between 1.41 and 2.3
{
;MsgBox, RD14
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\PHILIPP
Schraubanker RD14.smd
SendInput, {Enter}
sleep 300
;3anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
```

```
Sendinput, {Enter}
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4  ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
Sendinput, xx
;3anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 - 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 - 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4  ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
Sendinput, {ESC}
}
If VES between 2.31 and 3.4
{
;MsgBox, RD16
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
```

```
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\PHILIPP
Schraubanker RD16.smd
SendInput, {Enter}
sleep 300
;3anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4  ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
Sendinput, xx
;3anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 - 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 - 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
```

```
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4  ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
Sendinput, {ESC}
}
If VES between 3.41 and 4.6
{
;MsgBox, RD18
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\PHILIPP
Schraubanker RD18.smd
SendInput, {Enter}
sleep 300
;3anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4  ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
```

```
Sendinput, xx
;3anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 -10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 - 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4  ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
Sendinput, {ESC}
}
If VES between 4.61 and 5.7
{
;MsgBox, RD20
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\PHILIPP
Schraubanker RD20.smd
SendInput, {Enter}
sleep 300
;3anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
```

```
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4  ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
Sendinput, xx
;3anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 - 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 - 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4  ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
Sendinput, {ESC}
}
If VES between 5.71 and 7.1
{
```

```
;MsgBox, RD24
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\PHILIPP
Schraubanker RD24.smd
SendInput, {Enter}
sleep 300
;3anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4  ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
Sendinput, xx
;3anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 - 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 - 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
```

```
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4   ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
Sendinput, {ESC}
}
If VES between 7.11 and 11.4
{
;MsgBox, RD30
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\PHILIPP
Schraubanker RD30.smd
SendInput, {Enter}
sleep 300
;3anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4  ;X
Sendinput, ,-
```

```
Sendinput, % OutputVarG * 1 / 4 ;Y
Sendinput, ,%OutputVarH3%
Sendinput, {Enter}
Sendinput, xx
;3anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 - 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;4anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 3 / 4 - 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;2anker
SendInput, {F2}
Sendinput, % OutputVarL / 4 * 3 ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
;1anker
SendInput, {F2}
Sendinput, % OutputVarL / 4  ;X
Sendinput, ,-
Sendinput, % OutputVarG * 1 / 4 + 10 ;Y
Sendinput, ,0
Sendinput, {Enter}
Sendinput, {ESC}
}

if Ves > 11.4
{
MsgBox, cannot lift, try ohter anchors
}
VES1 := ((OutputVarL*OutputVarH3+(OutputVarH2-
OutputVarH3)*OutputVarb2+(OutputVarH1-
OutputVarH3)*OutputVarb1+OutputVaru*(OutputVarL-OutputVarb1-OutputVarb2-
OutputVarL3)-(OutputVaru-OutputVard)*(OutputVarL-OutputVarb1-
OutputVarb2)/2)*OutputVarG+2*(OutputVarL-OutputVarb1-
OutputVarb2)*OutputVarb1*(OutputVarH1-OutputVarH3))/1000000*2.5
sleep, 1100
;ANKERA TOREC
If VES1 between 0 and 0.7
{
;RD12
SendInput, {CTRL down}{d down}{d up}{CTRL up}
```

```
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-
all\5mouse\cube\ankera\Gewindetransportanker RD12.smd
sleep 100
SendInput, {Enter}
sleep 300
;1anker
SendInput, {F2}
sleep 100
SendInput, y
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter
;2anker
SendInput, {F2}
sleep 100
SendInput, x
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 * 3 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
Sendinput, {ESC}{ESC}
}
If VES1 between 0.71 and 1.1
{
;MsgBox, RD14
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-
all\5mouse\cube\ankera\Gewindetransportanker RD14.smd
sleep 100
SendInput, {Enter}
sleep 300
;1anker
SendInput, {F2}
sleep 100
SendInput, y
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
```

```
Sendinput, ,-
Sendinput, % OutputVarG / 4 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
;2anker
SendInput, {F2}
sleep 100
SendInput, x
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 * 3 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
Sendinput, {ESC}
}
If VES1 between 1.11 and 1.7
{
;MsgBox, RD16
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-
all\5mouse\cube\ankera\Gewindetransportanker RD16.smd
sleep 100
SendInput, {Enter}
sleep 300
;1anker
SendInput, {F2}
sleep 100
SendInput, y
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
;2anker
SendInput, {F2}
sleep 100
SendInput, x
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 * 3 ;Y
```

```
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
Sendinput, {ESC}
}
If VES1 between 1.71 and 2.3
{
;MsgBox, RD18
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-
all\5mouse\cube\ankera\Gewindetransportanker RD18.smd
sleep 100
SendInput, {Enter}
sleep 300
;1anker
SendInput, {F2}
sleep 100
SendInput, y
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
;2anker
SendInput, {F2}
sleep 100
SendInput, x
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 * 3 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
Sendinput, {ESC}
}
If VES1 between 2.31 and 2.9
{
;MsgBox, RD20
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-
all\5mouse\cube\ankera\Gewindetransportanker RD20.smd
sleep 100
```

```
SendInput, {Enter}
sleep 300
;1anker
SendInput, {F2}
sleep 100
SendInput, y
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
;2anker
SendInput, {F2}
sleep 100
SendInput, x
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 * 3 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
Sendinput, {ESC}
}
If VES1 between 2.91 and 3.6
{
;MsgBox, RD24
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-
all\5mouse\cube\ankera\Gewindetransportanker RD24.smd
sleep 100
SendInput, {Enter}
sleep 300
;1anker
SendInput, {F2}
sleep 100
SendInput, y
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
```

```
;2anker
SendInput, {F2}
sleep 100
SendInput, x
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 * 3 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
Sendinput, {ESC}
}
If VES1 between 3.61 and 5.7
{
;MsgBox, RD30
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-
all\5mouse\cube\ankera\Gewindetransportanker RD30.smd
sleep 100
SendInput, {Enter}
sleep 300
;1anker
SendInput, {F2}
sleep 100
SendInput, y
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
;2anker
SendInput, {F2}
sleep 100
SendInput, x
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 * 3 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
Sendinput, {ESC}
}
```

```
If VES1 between 5.71 and 9
{
;MsgBox, RD36
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-
all\5mouse\cube\ankera\Gewindetransportanker RD36.smd
sleep 100
SendInput, {Enter}
sleep 300
;1anker
SendInput, {F2}
sleep 100
SendInput, y
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
;2anker
SendInput, {F2}
sleep 100
SendInput, x
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 * 3 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
Sendinput, {ESC}
}
If VES1 between 9.1 and 11.4
{
;MsgBox, RD42
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-
all\5mouse\cube\ankera\Gewindetransportanker RD42.smd
sleep 100
SendInput, {Enter}
sleep 300
;1anker
SendInput, {F2}
sleep 100
```

```
SendInput, y
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
;2anker
SendInput, {F2}
sleep 100
SendInput, x
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 * 3 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
Sendinput, {ESC}
}
If VES1 between 11.41 and 17.9
{
;MsgBox, RD52
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-
all\5mouse\cube\ankera\Gewindetransportanker RD52.smd
sleep 100
SendInput, {Enter}
sleep 300
;1anker
SendInput, {F2}
sleep 100
SendInput, y
SendInput, x
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
;2anker
SendInput, {F2}
sleep 100
SendInput, x
SendInput, x
```

```
sleep 100
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 * 3 ;Y
Sendinput, ,
Sendinput, % OutputVarL3 / 2
Sendinput, {Enter}
Sendinput, {ESC}
}
if VES1 > 17.91
{
MsgBox,Too heavy, try another anchor type.
}
sleep 1200
;LORO
TRUBKA := (OutputvarH3*1)
If TRUBKA between 0 and 9.5
{
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\LORO95.smd
SendInput, {Enter}
sleep 300
SendInput, {F2}
sleep 100
Sendinput, % OutputVarL - OutputVarb2 - OutputVarL3 / 2 ;X
Sendinput, ,-
Sendinput, %OutputVarK% ;Y
Sendinput, ,
Sendinput, %OutputVarH3%
Sendinput, {Enter}
Sendinput, {ESC}
}
If TRUBKA between 9.51 and 12
{
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\LORO120.smd
SendInput, {Enter}
sleep 300
SendInput, {F2}
sleep 100
Sendinput, % OutputVarL - OutputVarb2 - OutputVarL3 / 2 ;X
Sendinput, ,-
Sendinput, %OutputVarK% ;Y
Sendinput, ,
Sendinput, %OutputVarH3%
Sendinput, {Enter}
Sendinput, {ESC}
```

```
}
If TRUBKA between 12.01 and 14
{
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\LORO140.smd
SendInput, {Enter}
sleep 300
SendInput, {F2}
sleep 100
Sendinput, % OutputVarL - OutputVarb2 - OutputVarL3 / 2 ;X
Sendinput, ,-
Sendinput, %OutputVarK% ;Y
Sendinput, ,
Sendinput, %OutputVarH3%
Sendinput, {Enter}
Sendinput, {ESC}
}
If TRUBKA between 14.01 and 16
{
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\LORO160.smd
SendInput, {Enter}
sleep 300
SendInput, {F2}
sleep 100
Sendinput, % OutputVarL - OutputVarb2 - OutputVarL3 / 2 ;X
Sendinput, ,-
Sendinput, %OutputVarK% ;Y
Sendinput, ,
Sendinput, %OutputVarH3%
Sendinput, {Enter
Sendinput, {ESC}
}
If TRUBKA between 16.01 and 18
{
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\LORO180.smd
SendInput, {Enter}
sleep 300
SendInput, {F2}
sleep 100
Sendinput, % OutputVarL - OutputVarb2 - OutputVarL3 / 2 ;X
Sendinput, ,-
Sendinput, %OutputVarK% ;Y
Sendinput, ,
```

```
Sendinput, %OutputVarH3%
Sendinput, {Enter}
Sendinput, {ESC}
}
If TRUBKA between 18.01 and 20
{
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\LORO200.smd
SendInput, {Enter}
sleep 300
SendInput, {F2}
sleep 100
Sendinput, % OutputVarL - OutputVarb2 - OutputVarL3 / 2 ;X
Sendinput, ,-
Sendinput, %OutputVarK% ;Y
Sendinput, ,
Sendinput, %OutputVarH3%
Sendinput, {Enter}
Sendinput, {ESC}
}
If TRUBKA between 20.01 and 100
{
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\LORO220.smd
SendInput, {Enter}
sleep 300
SendInput, {F2}
sleep 100
Sendinput, % OutputVarL - OutputVarb2 - OutputVarL3 / 2 ;X
Sendinput, ,-
Sendinput, %OutputVarK% ;Y
Sendinput, ,
Sendinput, %OutputVarH3%
Sendinput, {Enter}
Sendinput, {ESC}
}
;KANTE
sleep, 1300
If Kante2 = 1
{
Sendinput {ESC}
Sendinput D
Sendinput {F4}
Sendinput {F6}
Sendinput 0
Sendinput {F8}
Sendinput % OutputvarG - OutputVarKante - OutputVarKante
```

```
Sendinput {Enter}
Sleep 50
Sendinput, % OutputVarKante -0.75 ;X
Sendinput, ,-
Sendinput, % OutputVarG - OutputVarKante ;Y
Sendinput, ,0 ;Z
Sendinput {Enter}
sleep 50
Sendinput, 1.5 ;X
Sendinput, 0, ;Y
Sendinput, 0, ;Z
Sendinput, {Enter}
sleep 50
Sendinput, -0.75, ;X
Sendinput, 0, ;Y
Sendinput, 1 ;Z
Sendinput, {Enter}
sleep 50
Sendinput, {Enter}
sleep 50
Sendinput, {Enter}
sleep 50
}
sleep, 1300
If Kante1 = 1
{
Sleep 1100
Sendinput {ESC}
Sendinput D
Sendinput {F4}
sleep 50
Sendinput {F6}
Sendinput 0
Sendinput {F8}
Sendinput % OutputvarL - OutputVarKante - OutputVarKante
Sendinput {Enter}
Sleep 100
Sendinput, % OutputvarL - OutputVarKante  ;X
Sendinput, ,-
Sendinput, % OutputVarG - OutputVarKante + 0.75 ;Y
Sendinput, ,0 ;Z
Sendinput {Enter}
sleep 100
Sendinput, 0, ;X
Sendinput, 1.5, ;Y
Sendinput, 0, ;Z
Sendinput, {Enter}
sleep 100
Sendinput, 0, ;X
Sendinput, -0.75, ;Y
Sendinput, 1 ;Z
```

```
Sendinput, {Enter}
sleep 50
Sendinput, {Enter}
sleep 50
Sendinput, {Enter}
sleep 50
}
If Kante3 = 1
{
Sleep 1200
Sendinput {ESC}
Sendinput D
Sendinput {F4}
sleep 50
Sendinput {F6}
Sendinput 0
Sendinput {F8}
Sendinput % OutputvarL - OutputVarKante - OutputVarKante
Sendinput {Enter}
Sleep 100
Sendinput, % OutputvarL - OutputVarKante  ;X
Sendinput, ,
Sendinput, % 0 - OutputVarKante - 0.75 ;Y
Sendinput, ,0 ;Z
Sendinput {Enter}
sleep 100
Sendinput, 0, ;X
Sendinput, 1.5, ;Y
Sendinput, 0, ;Z
Sendinput, {Enter}
sleep 100
Sendinput, 0, ;X
Sendinput, -0.75, ;Y
Sendinput, 1 ;Z
Sendinput, {Enter}
sleep 50
Sendinput, {Enter}
sleep 50
Sendinput, {Enter}
sleep 50
}
if ISOKORB = 1
{
sleep 1500
;MsgBox, ISOKORB
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
SendInput, %Inputfile%
SendInput, {Enter}
SendInput, {Enter}
sleep 300
```

```
sleep 200
;OK
ControlGetText, v_OK, OK, A
if (v_OK = "")
  ControlGetText, v_OK, &OK, A
if (v_OK <> "")
  ControlClick, %v_OK%, A
sleep 200
;OK
ControlGetText, v_OK, OK, A
if (v_OK = "")
  ControlGetText, v_OK, &OK, A
if (v_OK <> "")
  ControlClick, %v_OK%, A
sleep 100
SendInput, {Click 1000 500}
sleep 100
Sendinput, {F4}
sleep 50
Sendinput, z
sleep 50
Sendinput, z
sleep 50
Sendinput, z
Sendinput, %OutputvarL%,0,0
sleep 100
Sendinput, {Enter}
sleep 100
Sendinput, {ESC}
Sendinput, {ESC}
sleep 300
Sendinput, {ESC}
Sendinput, {ESC}
sleep 50
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\Z.smd
sleep 100
SendInput, {Enter}
sleep 300
SendInput, {F4}
Sendinput, %OutputVarL% ;X
Sendinput, ,-
Sendinput, % OutputVarG / 4 ;Y
Sendinput, ,22
Sendinput, {Enter}
sleep 500
Sendinput, {ESC}
sleep 50
Sendinput, {ESC}
```

```
sleep 50
SendInput, {CTRL down}{d down}{d up}{CTRL up}
sleep 300
WinWaitActive, Open
SendInput, C:\Users\alber\Desktop\Scripts-all\5mouse\cube\ankera\Z.smd
sleep 100
SendInput, {Enter}
sleep 500
SendInput, {F4}
Sendinput, %OutputVarL% ;X
sleep 50
Sendinput, ,-
sleep 50
Sendinput, % OutputVarG / 4 * 3 ;Y
sleep 50
Sendinput, ,22
sleep 50
Sendinput, {Enter}
}
BlockInput, Off
Msgbox, DONE
Exitapp
```

Lisa-6 Trepimooduli skript (baas)

```
#NoEnv
#Warn
SendMode Input
SetWorkingDir %A_ScriptDir%  ; Ensures a consistent starting directory.
SetTitleMatchMode, 2
SendMode, Event
SetKeyDelay, 50
F1 & q::
SendMode Event
Gui, Add, Edit, voutputvarn y35 x230, 10 ;Result input in variable "Input"
Gui, Add, Edit, voutputvara y110 x240, 18 ;Result input in variable "Input"
Gui, Add, Edit, voutputvarb y150 x180, 25 ;Result input in variable "Input"
Gui, Add, Edit, voutputvarc y140 x350, 20 ;Result input in variable "Input"
Gui, Add, Edit, voutputvard y0 x470, 35 ;Result input in variable "Input"
Gui, Add, Edit, voutputvare y25 x575, 12.5 ;Result input in variable "Input"
Gui, Add, Edit, voutputvarf y40 x545, 30 ;Result input in variable "Input"
Gui, Add, Edit, voutputvari y110 x505, 13 ;Result input in variable "Input"
Gui, Add, Edit, voutputvark y220 x75, 35 ;Result input in variable "Input"
Gui, Add, Edit, voutputvarg y300 x5, 30 ;Result input in variable "Input"
Gui, Add, Edit, voutputvarh y320 x30, 12.5 ;Result input in variable "Input"
Gui, Add, Edit, voutputvarj y380 x63, 13 ;Result input in variable "Input"
Gui, Add, Edit, voutputvarl y500 x10, 100 ;Result input in variable "Input"
Gui, Add, Button, y600 x530 gPerform, READY! ;Will launch "Perform" routine
Gui, Add, Picture, x0 y0 w600 h661 +0x4000000, C:\Users\alber\Desktop\treppe.jpg
Gui, Show
return
Guiclose:
Exitapp
Perform:
WinActivate, STRAKON Cube
WinWaitActive, STRAKON Cube
Gui, Submit,  ; Passes completed text into variable
;Null point
SendInput {C}{Esc}
sleep 300
Sendinput D
Sleep 300
Sendinput, {F6}
sleep 300
Sendinput, 0
sleep 300
Sendinput, {tab}
sleep 200
Sendinput, %outputvarL%
Sendinput {Enter}
sleep 200
Sendinput 0,0,0
Sendinput {Enter}
sleep 300
;1stpoint
```

```
Sendinput 0,0,
Sendinput % outputvarg - outputvarh
Sendinput {Enter}
;2nd point
Sendinput, -%outputvarj%,0,0
Sendinput {Enter}
;3rd point
Sendinput 0,0,%outputvarh%
Sendinput {Enter}
;4th point
Sendinput %outputvark%,0,0
Sendinput {Enter}
;stairs
sleep 300
LOOP % OutputVarn - 1
{
Sendinput 0,0,%outputvara%
Sendinput {Enter}
Sendinput %outputvarb%,0,0
Sendinput {Enter}
}
;Last step point
Sendinput 0,0,%outputvara%
Sendinput {Enter}
;5th point
Sendinput %outputvard%,0,0
Sendinput {Enter}
;6th point
Sendinput 0,0,-%outputvare%
Sendinput {Enter}
;7th point
Sendinput, -%outputvari%,0,0
Sendinput {Enter}
;8th point
Sendinput 0,0,-
Sendinput % outputvarf - outputvare
Sendinput {Enter}
;9th point
Sendinput, -
Sendinput % outputvara * ( outputvarf - outputvare ) / outputvarb
Sendinput, ,0,0
Sendinput {Enter}
;10th point
Sendinput, -
Sendinput, % ( outputvark - outputvarj + outputvarb * outputvarn - outputvarb +
outputvard - outputvari - ( outputvara * ( outputvarf - outputvare ) / outputvarb ) - (
outputvark - outputvarj + outputvarc / outputvara * sqrt(outputvara * outputvara +
outputvarb * outputvarb) - outputvarg * outputvarb / outputvara))
Sendinput, ,0,-
Sendinput, % ( outputvarg + outputvara * outputvarn - outputvarf )
Sendinput {Enter}
```

```
Sendinput {Enter}
Sendinput {Enter}
sleep 300
Sendinput {Enter}
;end of modelling--------
;setting 3d views---------
sleep 500
Sendinput {Esc}
Sendinput {Esc}
sleep 200
Sendinput {Esc}
sleep 200
Sendevent, {shift down}{5 down}{5 up}{shift up}
sleep 200
Sendinput 0,0,0
Sendinput {Enter}
sleep 200
Sendinput 200,400
Sendinput {Enter}
sleep 200
Sendinput 200,100
Sendinput {Enter}
; dimensioning top stair landing--
sleep 1000
WinActivate, STRAKON Cube
WinWaitActive, STRAKON Cube
WinClose, STRAKON Cube
WinActivate, STRAKON premium
WinWaitActive, STRAKON premium
xstart := 200 - outputvarj
ystart := 100
x1start := 200 - outputvarj
y1start := 400
sleep 200
Sendevent, {alt down}{2 down}{2 up}{alt up}
sendinput,
{down}{down}{down}{down}{down}{down}{down}{down}{down}{down}{enter}
sleep 200
Sendinput, % xstart
Sendinput, ,
Sendinput, % ystart
Sendinput, {enter}
;0 point
;1 point
Sendinput % xstart + outputvark
Sendinput, ,
Sendinput % ystart
Sendinput, {enter}
;prostanovka linii-----------------------
Sendinput % x1start
Sendinput, ,
```

```
Sendinput % y1start + outputvarg + outputvara * outputvarn + 50
Sendinput, {enter}
;stupeni ----------------------
Sendinput % xstart + outputvarb + outputvark
Sendinput, ,
Sendinput % ystart
Sendinput, {enter}
stupenki := outputvarb
LOOP % OutputVarn - 2
{
Stupenki := Stupenki + outputvarb
Sendinput % xstart + stupenki + outputvark
Sendinput, ,
Sendinput % ystart
Sendinput, {enter}
}
;last landing----------
Sendinput % xstart + outputvark + outputvarb * ( outputvarn - 1) + outputvard
Sendinput, ,
Sendinput % ystart
Sendinput, {enter}
Sendinput {F7}
;dimensioning stair landings2 -----
sleep 200
Sendevent, {alt down}{2 down}{2 up}{alt up}
sendinput,
{down}{down}{down}{down}{down}{down}{down}{down}{down}{down}{enter}
sleep 200
Sendinput, % xstart
Sendinput, ,
Sendinput, % ystart
Sendinput, {enter}
;v nule
;1 point
Sendinput % xstart + outputvark
Sendinput, ,
Sendinput % ystart
Sendinput, {enter}
;prostanovka linii----------
Sendinput % x1start
Sendinput, ,
Sendinput % ystart + outputvarl + 30
Sendinput, {enter}
;stupeni ----------------------
Sendinput % xstart + outputvarb + outputvark
Sendinput, ,
Sendinput % ystart
Sendinput, {enter}
stupenki := outputvarb
LOOP % OutputVarn - 2
{
```

```
Stupenki := Stupenki + outputvarb
Sendinput % xstart + stupenki + outputvark
Sendinput, ,
Sendinput % ystart
Sendinput, {enter}
}
;last stair landing----------
Sendinput % xstart + outputvark + outputvarb * ( outputvarn - 1) + outputvard
Sendinput, ,
Sendinput % ystart
Sendinput, {enter}
Sendinput {F7}
;setting max dim line------
sleep 200
Sendevent, {alt down}{2 down}{2 up}{alt up}
sendinput,
{down}{down}{down}{down}{down}{down}{down}{down}{down}{down}{enter}
sleep 200
Sendinput, % xstart
Sendinput, ,
Sendinput, % ystart
Sendinput, {enter}
;0 point
;1point
Sendinput % xstart + outputvark + outputvarb * ( outputvarn - 1 ) + outputvard
Sendinput, ,
Sendinput % ystart
Sendinput, {enter}
;prostanovka linii-----------
Sendinput % x1start
Sendinput, ,
Sendinput % ystart + outputvarl + 50
Sendinput, {enter}
Sendinput {F7}
;Prostanovka gabaritov--2 verh------
sleep 200
Sendevent, {alt down}{2 down}{2 up}{alt up}
sendinput,
{down}{down}{down}{down}{down}{down}{down}{down}{down}{down}{enter}
sleep 200
Sendinput, % xstart
Sendinput, ,
Sendinput, % ystart
Sendinput, {enter}
;v nule
;1 point
Sendinput % xstart + outputvark + outputvarb * ( outputvarn - 1 ) + outputvard
Sendinput, ,
Sendinput % ystart
Sendinput, {enter}
;prostanovka linii-----------------------
```

```
Sendinput % x1start
Sendinput, ,
Sendinput % y1start + outputvarg + outputvara * outputvarn + 70
Sendinput, {enter}
Sendinput {F7}
;setting hor lower dim line-----upper view---

sleep 200
Sendevent, {alt down}{2 down}{2 up}{alt up}
sendinput,
{down}{down}{down}{down}{down}{down}{down}{down}{down}{down}{enter}
sleep 200
Sendinput, % xstart
Sendinput, ,
Sendinput, % ystart
Sendinput, {enter}
;v nule
;1 point
Sendinput % xstart + outputvarj
Sendinput, ,
Sendinput % ystart
Sendinput, {enter}
;setting lines--------------
Sendinput % xstart
Sendinput, ,
Sendinput % y1start - 50
Sendinput, {enter}
Sendinput % xstart + outputvarj + ( outputvark - outputvarj + outputvarc /
outputvara * sqrt(outputvara * outputvara + outputvarb * outputvarb) - outputvarg *
outputvarb / outputvara)
Sendinput, ,
Sendinput % ystart
Sendinput, {enter}
Sendinput % ( outputvark + outputvarb * ( outputvarn - 1 ) + outputvard - outputvari
- (( outputvark - outputvarj + outputvarc / outputvara * sqrt(outputvara * outputvara
+ outputvarb * outputvarb) - outputvarg * outputvarb / outputvara )) -( outputvara *
( outputvarf - outputvare ) / outputvarb ))
Sendinput, ,
Sendinput % ystart
Sendinput, {enter}
Sendinput {F7}
return
```

```
SetTitleMatchMode, 2 ;
#SingleInstance, Off
#NoTrayIcon
;OPTIMIZATIONS START
#NoEnv
#MaxHotkeysPerInterval 99000000
#HotkeyInterval 99000000
#KeyHistory 0
ListLines Off
Process, Priority, , A
SetBatchLines, -1
SetKeyDelay, -1, -1
SetMouseDelay, -1
SetDefaultMouseSpeed, 0
SetWinDelay, -1
SetControlDelay, -1
SendMode Input
#NoEnv
; #Warn
SendMode Input  y.
SetWorkingDir %A_ScriptDir%

#Persistent
#SingleInstance, Force
SetTitleMatchMode, 2
SetWinDelay, 50
SendMode Input
SetBatchLines, 1
Gui, +Border +ToolWindow +AlwaysOnTop

Gui, Add, Text, x15 y10, Bar
Gui, Add, Text, x45 y10, Diam
Gui, Add, Text, x75 y10, cc1
Gui, Add, Text, x95 y10, cc2

Gui, Add, Button, x0 y25 w50 gStr, -
Gui, Add, Button, x0 y50 w50 gL, L
Gui, Add, Button, x0 y75 w50 gU, U
Gui, Add, Button, x0 y100 w50 gO, O
Gui, Add, Button, x0 y125 w50 gii, Ind
Gui, Add, Button, x0 y150 w50 gii2, Ind

Gui, Add, Button, x50 y175 w50 gsave, save
Gui, Add, Button, x0 y175 w50 greinf, Reinfoc

Gui, Add, Edit, x50 y25  vOutputVarStr, 8
Gui, Add, Edit, x50 y50  vOutputVarL, 8
Gui, Add, Edit, x50 y75  vOutputVarU, 8
Gui, Add, Edit, x50 y100  vOutputVarO, 8
Gui, Add, Edit, x50 y125  vOutputVarii, 8
Gui, Add, Edit, x50 y150 vOutputVarii2, 8

Gui, Add, Edit, x70 y25  vOutputVarStrcc1, 2.5
```

```
Gui, Add, Edit, x95 y25  vOutputVarStrcc2, 2.5

Gui, Add, Edit, x70 y50  vOutputVarLcc1, 2.5

Gui, Add, Edit, x70 y75  vOutputVarUcc1, 2.5

Gui, Add, Edit, x70 y100  vOutputVarOcc1, 2.5

Gui, Add, Edit, x70 y125  vOutputVariicc1, 2.5

Gui, Add, Edit, x70 y150  vOutputVarii2cc1, 2.5

Gui, Show, w120 h200,Helpbox
SetTimer, CheckActive, 100
return
ii:
{
WinActivate, STRAKON Cube
MouseGetPos x, y
SendInput {Click 35 90}

Sendinput {F4}
Sendinput %OutputVarii%
Sendinput, .0
Sendinput {Tab}
Sendinput {Tab}
Sendinput %OutputVariicc1%
Sendinput {Enter}
MouseMove %x%, %y%
return
}
ii2:
{
WinActivate, STRAKON Cube
MouseGetPos x, y
SendInput {Click 35 90}

Sendinput {F4}
Sendinput %OutputVarii2%
Sendinput, .0
Sendinput {Tab}
Sendinput {Tab}
Sendinput %OutputVarii2cc1%
Sendinput {Enter}
MouseMove %x%, %y%
return
}
save:
Gui, Submit, Nohide
CheckActive:
IfWinActive, STRAKON Cube
{
WinSet, AlwaysOnTop, ON, Pedrobox ahk_class AutoHotkeyGUI
}
Else
```

```
WinSet, AlwaysOnTop, Off, Pedrobox ahk_class AutoHotkeyGUI
Return
Reinf:
{
WinActivate, STRAKON Cube
MouseGetPos x, y
SendInput {Click 300 35}
MouseMove %x%, %y%
return
}
GuiClose:
Exitapp
Str:
{
WinActivate, STRAKON Cube
 MouseGetPos x, y
Sendinput {Esc}
Sendinput {Esc}
SendInput {Click 100 100}
sleep 100
Sendinput {F3}
Sendinput str
Sendinput {Enter}
sleep 100
Mousemove, 500, 500
SendInput {MButton}
Sendinput {F7}
Sendinput %OutputVarStr%
Sendinput, .0
Sendinput {Tab}
Sendinput %OutputVarStrcc1%
Sendinput {Tab}
Sendinput %OutputVarStrcc2%
Sendinput {Enter}
MouseMove %x%, %y%
return
}
L:
{
WinActivate, STRAKON Cube
 MouseGetPos x, y
Sendinput {Esc}
Sendinput {Esc}
SendInput {Click 100 100}
sleep 100
Sendinput {F3}
Sendinput L
Sendinput {Enter}
sleep 100
Mousemove, 500, 500
SendInput {MButton}
Sendinput {F7}
Sendinput %OutputVarL%
Sendinput, .0
Sendinput {Tab}
```

```
Sendinput %OutputVarLcc1%

if OutputVarL = 8
{
Sendinput {Tab}
Sendinput 40
Sendinput {Tab}
Sendinput 40
}
if OutputVarL = 10
{
Sendinput {Tab}
Sendinput 50
Sendinput {Tab}
Sendinput 50
}
if OutputVarL = 12
{
Sendinput {Tab}
Sendinput 60
Sendinput {Tab}
Sendinput 60
}
if OutputVarL = 14
{
Sendinput {Tab}
Sendinput 70
Sendinput {Tab}
Sendinput 70
}
if OutputVarL = 16
{
Sendinput {Tab}
Sendinput 80
Sendinput {Tab}
Sendinput 80
}
if OutputVarL = 18
{
Sendinput {Tab}
Sendinput 90
Sendinput {Tab}
Sendinput 90
}
if OutputVarL = 20
{
Sendinput {Tab}
Sendinput 100
Sendinput {Tab}
Sendinput 100
}
if OutputVarL = 25
{
Sendinput {Tab}
Sendinput 120
```

```
Sendinput {Tab}
Sendinput 120
}
Sendinput {Enter}
MouseMove %x%, %y%
return
}
U:
{
WinActivate, STRAKON Cube
 MouseGetPos x, y
Sendinput {Esc}
Sendinput {Esc}
SendInput {Click 100 100}
sleep 100
Sendinput {F3}
Sendinput U
Sendinput {Enter}
sleep 100
Mousemove, 500, 500
SendInput {MButton}
Sendinput {F7}
Sendinput %OutputVarU%
Sendinput, .0
Sendinput {Tab}
Sendinput %OutputVarUcc1%
if OutputVarU = 8
{
Sendinput {Tab}
Sendinput 40
}
if OutputVarU = 10
{
Sendinput {Tab}
Sendinput 50
}
if OutputVarU = 12
{
Sendinput {Tab}
Sendinput 60
}
if OutputVarU = 14
{
Sendinput {Tab}
Sendinput 70
}
if OutputVarU = 16
{
Sendinput {Tab}
Sendinput 80
}
if OutputVarU = 18
{
Sendinput {Tab}
Sendinput 90
```

```
}
if OutputVarU = 20
{
Sendinput {Tab}
Sendinput 100
}
if OutputVarU = 25
{
Sendinput {Tab}
Sendinput 120
}

Sendinput {Enter}
MouseMove %x%, %y%
return
}
O:
{
WinActivate, STRAKON Cube
 MouseGetPos x, y
Sendinput {Esc}
Sendinput {Esc}
SendInput {Click 100 100}
sleep 100
Sendinput {F3}
Sendinput Stir
Sendinput {Enter}
sleep 100
Mousemove, 500, 500
SendInput {MButton}
Sendinput {F7}
Sendinput %OutputVarO%
Sendinput, .0
Sendinput {Tab}
Sendinput %OutputVarOcc1%
Sendinput {Enter}
MouseMove %x%, %y%
return
}
```

Lisa-8 Kiirteksti ja Translate skript (baas)

```
; Google translate
F1 & F2::
{
 Send, ^c
 Sleep 50
 Run, https://translate.google.com/?sl=de&tl=ru&text=%clipboard%&op=translate
 Return
}

::S1::Schnitt 1-1
…
Return
```

Lisa-9 3D-Koosteelementide skript (baas)

```
NoEnv  ; Recommended for performance and compatibility with future AutoHotkey releases.
; #Warn  ; Enable warnings to assist with detecting common errors.
SendMode Input  ; Recommended for new scripts due to its superior speed and reliability.
SetWorkingDir %A_ScriptDir%  ; Ensures a consistent starting directory.
SetTitleMatchMode, 2
#SingleInstance, Off
#NoTrayIcon
SendInput, {CTRL down}{e down}{e up}{CTRL up}
SendInput, {F1}
SendInput, 2200086
SendInput, {Enter}
Exitapp
```

Lisa-10 2D-Koosteelementide skript (baas)

```
#NoEnv  ; Recommended for performance and compatibility with future AutoHotkey releases.
; #Warn  ; Enable warnings to assist with detecting common errors.
SendMode Input  ; Recommended for new scripts due to its superior speed and reliability.
SetWorkingDir %A_ScriptDir%  ; Ensures a consistent starting directory.
SetTitleMatchMode, 2
#SingleInstance, Off
InputBox, OutputVar1, L=...m?
Sendinput, !a
Sendinput,{down}{down}{down}{down}{down}{enter}
Blockinput, On
MouseGetPos x, y
Sendinput, {Click 200 1000}
MouseMove %x%, k%y%, 0
Sendinput, 860-%OutputVar1%
Sendinput, {enter}
KeyWait, LButton, D
sleep 100
Sendinput, {enter}
Sendinput, {Click 200 1000}
Sendinput, Flex-Kunststroffrohr 32/24.5mm
Sendinput, {enter}
sleep 100
Sendinput, L=%OutputVar1%Lfdm
Sendinput, {enter}

return
```

```
#NoEnv
; #Warn
SendMode Input
SetWorkingDir %A_ScriptDir%
SendInput {C}{Esc}
sleep 300
Sendinput q
SendInput {Click 400 200}
sleep 200
SendInput 10000{Enter}
sleep 100
SendInput -10000{Enter}
sleep 100
SendInput 10000{Enter}
sleep 100
SendInput {Esc}
SendInput {Click 410 210}
sleep 100
SendInput A
sleep 100
SendInput {Click 1200 700}
sleep 100
SendInput {Enter}
Sendinput {ctrl}A
Sleep 200
Sendinput h
sleep 300
Sendinput q
SendInput {Click 400 200}
sleep 200
SendInput 10000{Enter}
sleep 100
SendInput -10000{Enter}
sleep 100
SendInput 11000{Enter}
sleep 100
SendInput {Esc}
Sendinput {Click 530 35}
sleep 100
Sendinput {Click 850 90}
Sleep 200
SendInput {Esc}{Esc}
SendInput {Click 1100 400}
sleep 300
SendInput A
sleep 300
SendInput {Click 1100 400}
SendInput {Click 1100 400}
sleep 200
```

```
SendInput {Enter}

sleep 100
SendInput {Esc}{Esc}
SendInput f
sleep 100
SendInput {Click 1200 700}
sleep 100
SendInput {Enter}
SendInput {Click 1200 700}
sleep 100
SendInput 1000
SendInput {Enter}
SendInput 1000
SendInput {Enter}
Return
```