Implementacija računalnih simulacija pri planiranju proizvodnih procesa

Bulić, Martin

SVEUČILIŠTE U RIJECI

TEHNIČKI FAKULTET

Diplomski sveučilišni studij strojarstva

Diplomski rad

IMPLEMENTACIJA RAČUNALNIH SIMULACIJA PRI PLANIRANJU PROIZVODNIH PROCESA

Martin Bulić

0069076493

Rijeka, ožujak 2024.

SVEUČILIŠTE U RIJECI

TEHNIČKI FAKULTET

Diplomski sveučilišni studij strojarstva

Diplomski rad

IMPLEMENTACIJA RAČUNALNIH SIMULACIJA PRI PLANIRANJU PROIZVODNIH PROCESA

Mentor: prof. dr. sc. Mladen Perinić

doc. dr. sc. David Ištoković

Martin Bulić

Rijeka, ožujak 2024. 0069076493

IZJAVA

Sukladno članku 12. i 13. Pravilnika o diplomskom radu i završnom ispitu na diplomskom sveučilišnom studiju i stručnom studiju Tehničkog fakulteta Sveučilišta u Rijeci, izjavljujem da sam samostalno izradio diplomski rad pod nazivom „IMPLEMENTACIJA RAČUNALNIH SIMULACIJA PRI PLANIRANJU PROIZVODNIH PROCESA“. Rad sam izradio iz kolegija „Računalna simulacija proizvodnih procesa“ prema zadatku Povjerenstva za diplomske ispite diplomskog sveučilišnog studija strojarstva, pod vodstvom mentora prof. dr. sc. Mladena Perinića i doc. dr. sc. Davida Ištokovića. Zadatak je zadan 21.3.2022.

ZAHVALA

Ovim putem se želim zahvaliti mentorima, prof. dr. sc. Mladenu Periniću i doc. dr. sc. Davidu Ištokoviću, koji su mi pomogli u razumijevanju i svladavanju potrebne literature, koja mi je bila potrebna za izradu diplomskog rada te omogućili potrebnu opremu koja je bila od nominalne važnosti za izradu istog. Također želim zahvaliti obitelji, na svoj podršci i pomoći tijekom izrade rada, te firmi LTH Metal Cast Benkovac na pruženoj pomoći u vidu korištenja potrebnih informacija i podataka o dijelu pogona za izradu rada.

SADRŽAJ

1. UVOD

U dinamičkom okruženju moderne proizvodnje, ostati korak ispred svih, zahtijeva neumoljivu potrebu za efikasnosti, preciznosti i prilagodljivosti. Integracija najsuvremenijih tehnologija postala je imperativ za suočavanje s izazovima industrije koji se stalno razvijaju. Jedan takav tehnološki predvodnik koji je privukao veliku pozornost jest primjena računalnih simulacija u području planiranja proizvodnog procesa. Kako industrija sve više potražuje optimizaciju proizvodnje, smanjenje troškova te reduciranje vremena potrebnog za plasiranje proizvoda, računalne simulacije se pojavljuju kao moćan alat koji nudi neprocjenjive mogućnosti predviđanja i nadzora.

Ovaj rad zadire u raznolikost računalnih simulacija i njihovu stratešku integraciju u planiranje raznih procesa. Proizvodni sektor suočava se s kompleksnom problematikom i izazovima, od zamršenih proizvodnih procesa do zahtjeva za brzom izradom prototipa i fleksibilnim proizvodnim linijama. U tom kontekstu, računalne simulacije pojavljuju se kao transformacijska sila, pružajući virtualno okruženje za inženjere i planere za analizu, ponavljanje i usavršavanje procesa prije fizičke implementacije.

U nastavku rada slijedi teoretski dio koji vodi kroz osnovne koncepte simulacijskih modela, njihovih načela, prednosti i izazova povezanih s implementacijom istih, a posebno kroz simulaciju diskretnih događaja. Razlog zbog kojeg se posebno prolazi kroz simulaciju diskretnih događaja je taj što je ona korištena u istraživačkom dijelu rada. Potom slijedi opis software-a koji nudi stvaranje modela računalnih simulacija za razne procese a ponajprije proizvodne.

Nakon teoretskog dijela dolazi istraživački dio rada. U prvom dijelu objašnjen je dio proizvodnog pogona za kojeg je potrebno izraditi potpunu automatizaciju istog te odrediti optimalan broj namjenskih paleta. Potom prolazimo kroz objašnjenje izrade osnovnog simulacijskog modela dijela proizvodnog pogona, odnosno za trenutno stanje, te simulacijskog modela za isti dio proizvodnog pogona koji je potpuno automatiziran. Zadnji dio rada je provođenje simulacijskih eksperimenata s pomoću kojih će se odrediti najbolji moguć broj namjenskih paleta za ovaj proces.

Cilj je pružiti sveobuhvatno razumijevanje računalnih simulacija, načina izrade odnosno načina na koji ove simulacije preoblikuju budućnost planiranja proizvodnog procesa.

2. KONCEPT SIMULACIJE

Postoji puno definicija simulacije ali ne i opća. Sve su one točne i u globalu prate isti koncept a to je da simulacija oponaša rad sustava odnosno da je imitacija nekog sustava. Ona se sastoji od svih komponenti sustava s postavkama istog u stvarnom vremenu. Ovakva simulacija omogućava praćenje i razumijevanje ponašanja sustava tijekom vremena.[1]

U današnjem sve bržem i kompleksnijem svijetu, simulacija je postala neizostavan alat za razumijevanje, planiranje i rješavanje različitih problema i situacija. Simulacija je koncept koji se primjenjuje u različitim područjima kao što su inženjerstvo, ekonomija, medicina, vojska i drugi.

Simulacija je proces kreiranja modela stvarnog sustava ili situacije i njegovo eksperimentalno proučavanje bez potrebe za intervencijom u stvarnom okruženju. Ova tehnologija omogućuje stvaranje virtualnih modela stvarnog sustava, koji se ponašaju slično stvarnima, omogućavajući analizu njihova ponašanja, predviđanja budućih događaja i testiranja različitih scenarija. [2]

Svrha simulacijskog modela je višestruka. Prvo, ona omogućuje istraživanje i razumijevanje sustava i događaja bez potrebnog rizika i utjecaja na okruženje. Na primjer, u poslovnom okruženju menadžeri mogu simulirati različite poslovne strategije kako bi identificirali najbolji ishod situacije. Također, vojska koristi simulacije za provođenje vojnih vježbi. Dakle, drugi jako važan benefit korištenja simulacija je predviđanje ponašanja sustava i donošenje važnih odluka bez potrebe za djelovanjem u stvarnom svijetu. [3]

Primjene simulacije su brojne. Neke smo već nabrojili, a koriste se još u inženjeringu za projektiranje i ispitivanje različitih komponenti pa tako i cijelog sklopa (automobili, avioni, i dr.), u ekonomiji za analizu financijskih tržišta i trendova, u urbanističkom planiranju kako bi se predvidjeli efekti različitih urbanističkih projekata na gradsko okruženje, u meteorologiji za predviđanje kretanja ciklona, drugim riječima prognoze, itd.

2.1. Povijest simulacije

Simulacija kakvu danas poznajemo nije postojala sve do početka dvadeset prvog stoljeća. Na razvoj tehnologije a ujedno i simulacije uvelike je utjecao drugi svjetski rat. Povijesno gledajući postoje četiri studije koje se mogu smatrati začetcima simulacije kakvu danas poznajemo. Prva studija bila je rješavanje matematičkih problema s pomoću ENIAC računala (Električno

numerički integrator i računalo) 1940-ih, drugi je predstavljanje pokusa od strane Stanislawa Ulama i drugih znanstvenika koristeći se Monte Carlo metodom na računalu iz područja praktične fizike 1960-ih, treća je razvoj Paris – Durham šok modela na području astrofizike 1980-ih te povijest digitalnog modeliranja iz područja dizajna i arhitekture 1980-ih i 1990-ih.[4]

2.2. Koncepti modeliranja

Danas poznajemo 4 osnovna koncepta modeliranja, a to su: Monte Carlo metoda, modeliranje bazirano na agentima, modeliranje bazirano na diskretnim događajima i dinamičko modeliranje sustava.

2.2.1. Monte Carlo metoda

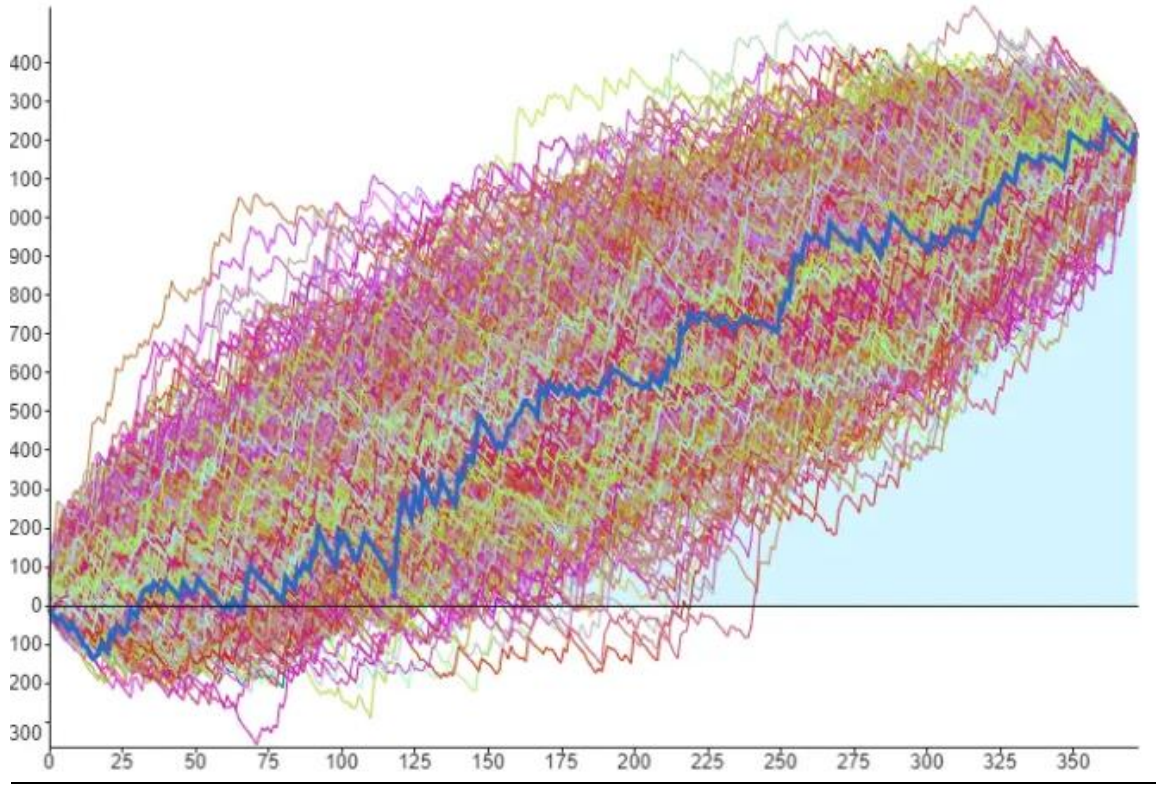
Metoda je dobila svoje ime prema kockarskoj destinaciji u Monacu zato što su šanse i slučajni ishodi srž ove metode kao što su za razne igre na sreću. Ova metoda je inicijalno razvijena od strane Stanislawa Ulama koji je, po struci matematičar, radio na tajnom projektu Manhattan razvijanja prvog atomskog oružja. Nakon što je svoju ideju podijelio sa svojim kolegom s projekta John Von Neumannom, zajedno su razvili simulaciju Monte Carlo.

Ova metoda se koristi u slučajevima kada je broj izračuna za promatrani problem velik i kada je sam izračun kompleksan. Drugim riječima, ovom metodom gradimo simulaciju koja pruža moguća rješenja korištenjem distribucije vrijednosti, kao što su uniformna ili normalna distribucija, za bilo koju varijablu koja ima inherentnu nesigurnost. Zatim ta simulacija iznova izračunava rezultate koristeći se svaki put različitim skupom nasumičnih vrijednosti između zadane minimalne i maksimalne granice vrijednosti. Uobičajeno je da se ovaj eksperiment ponovi i više tisuća puta, što je veći broj izvedenih pokusa to će biti veća točnost izlaza. Na kraju, model nam kao izlaz daje niz mogućih ishoda s vjerojatnošću ostvarenja svakog rezultata (slika 2.1).

Tri osnovna principa na kojima se temelji Monte Carlo simulacija:

1. Nasumičnost: Ključni dio ove simulacije je upravo nasumičnost. Mogućnost generiranja slučajnih brojeva omogućuje simulaciji modeliranje nesigurnosti i varijabilnosti u sustavu.
2. Ponavljanje eksperimenta: Monte Carlo simulacija funkcionira tako da se eksperiment ili eksperimenti povaljaju velik broj puta, pri čemu se koriste različite nasumične vrijednosti, a sve kako bi se dobila statistika i distribucija rezultata.

3. Procjena rezultata: Na osnovi velikog broja ponovljenih eksperimenata mogu se izračunati vjerojatnosti, srednje vrijednosti, varijacije i druge relativistički podatci.[5]



Slika 2.1 Dijagram kao rezultat Monte Carlo simulacija

Korištenje ove metode se svodi na tri osnovna koraka:

1. Postaviti predvidiv model gdje je potrebno utvrditi zavisne varijable koje treba predvidjeti te nezavisne (ulazne podatke) koje će utjecati na predviđanje.
2. Navesti distribucije vjerojatnosti nezavisnih varijabli. Mogu se upotrijebiti iskustveni podatci i subjektivna prosudba kako bi se definirao raspon vjerojatnih vrijednosti i dodijelila vjerojatnost za svaku.
3. Provoditi simulaciju više puta, generirajući nasumične vrijednosti nezavisnih varijabli. Ovaj korak je potrebno ponavljati više puta kako bi se akumulirao dovoljan broj rezultata kojim se stvara uzorak koji predstavlja beskonačan broj mogućih kombinacija.

Prednost korištenja ovog modela su brojni. Prije svega, pruža odgovore na probleme koji su analitički ne rješivi. Mogućnost modeliranja složenih i stohastičkih sustava, rukovanja nesigurnostima i varijacijama u analizi, preciznog procjenjivanja vjerojatnosti i statističkih parametara. Također, ako zadamo točne parametre, Monte Carlo metoda može u potpunosti istražiti raspon parametara promatranog problema.

Najveći nedostatak ove metode je da za korištenje iste je potrebno računalo. Čak i za male simulacije potrebno je i po nekoliko sati vremena da računalo obradi sve zadane parametre. Dodatni problem je neispravno rukovanje ovom tehnikom, ako se krivo zadaju ulazni podatci, izlazni će u najmanju ruku biti beskorisni.[6]

Danas je korištenje ove metode jako rasprostranjeno. Koristi se u:

- inženjerstvu za osjetljive analize i analize kvantitativne vjerojatnosti u procesu modeliranja,
- teoretskoj fizici za razne proračune pri modeliranju,
- računalne grafike za nasumično praćenje uzoraka mogućih kretanja svjetlosti,
- primjenjivoj statistici,
- umjetnoj inteligenciji za razvoj igara,
- financijama,
- medicini za simuliranje biomedicinsko nuklearne sinteze slika s visokim realizmom,
- analizama klimatskih promjena, itd.

2.2.2. Modeliranje simulacije zasnovane na agentima

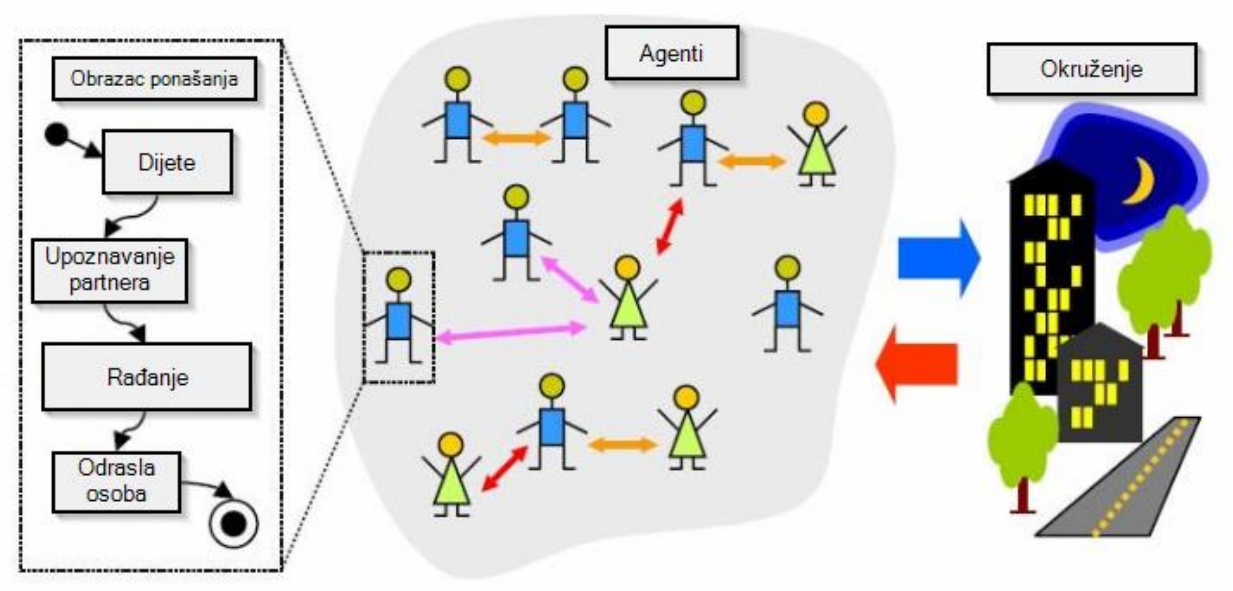
Modeliranje bazirano na agentima je danas jako moćan alat koji je pustio korijenja u raznim područjima. Simulacija zasnovana na agentima je vrsta simulacije koja se koristi za modeliranje i analizu složenih sustava uz pomoć autonomnih agenata koji djeluju u određenom okruženju te se integriraju u istom. Prvi put kad se koristila riječ agent i same definicije riječi kakva se danas koristi je teško odrediti. Prema povijenim izvorima pretpostavlja se da se radi o Johnu Hollandu i njegovom članku iz 1991 „Artificial Adaptive Agents“ u teoriji ekonomike. Upravo ta riječ čini ključ ove metode simuliranja.

Ne postoji univerzalna definicija termina „Agent“. Jedan dio konstruktora smatra da je agent bilo koja nezavisna komponenta koja je sposobna donositi odluke od primitivnih pa sve do kompleksnih. Drugi dio konstruktora smatra da ponašanje komponente mora biti prilagodljivo da bi se smatrala agentom. [6]

Kod modeliranja simulacije korištenjem ove metode, sustav je načinjen kao skupina autonomnih entiteta koji su sposobni sami donositi odluke. Svaki agent, entitet, ima svoje karakteristike, ciljeve, sposobnosti te svojstveno pristupa situaciji te donosi odluke na račun postavljenih pravila. Te odluke mogu biti razne a ovise o klasi koju agenti predstavljaju. Također, entiteti mogu i međusobno biti u interakciji a radnje koje obavljaj mogu biti ponavljajuće ili pak

natjecateljskog duha. Uz to, entiteti se mogu razvijati čime se otvara mogućnost pojave novog načina ponašanja entiteta.

Jednostavan primjer ovog modela se sastoji od sustava, agenata i njihova odnosa (slika 2.2). Čak i najjednostavniji modeli koncipirani ovom metodom mogu ispitati kompleksne uzorke ponašanja te pružiti vrijedne informacije o dinamici sustava stvarnog svijeta kojeg ona oponaša.



Slika 2.2 Simulacija bazirana na agentima

Većina modela temeljenih na ovoj metodi se sastoji od nekoliko elemenata:

1. Veći broj agenata posloženih po razinama (tzv. granularnost agenata).
2. Svojevoljno heurističko donošenje odluka.
3. Pravila prema kojima agenti uče ili se razvijaju.
4. Neprekidna interakcija kojom agenti međusobno komuniciraju razmjenjujući informacije čime utječu na razvoj jedni drugih.
5. Sustav (okolina) čini prostor u kojem agenti djeluju što može biti virtualno ili fizičko okruženje.
6. Modeliranje vremena: ABS simulacija dozvoljava modeliranje vremenskog toka događaja i promjena u sustavu. [6]

Prednosti ovakvog pristupa je modeliranje heterogena sustava. Pod heterogeni model smatra se da su statističke postavke distribucije jednake u različitim dijelovima distribucije. Ovaj način simuliranja omogućuje diskretno modeliranje više nego kontinuirano. Još jedna prednost je da osoba koja konstruira model ne mora biti upoznata do u detalj s mehanikom rada. Ovaj princip

nudi nasumičnost modela pošto se ne zna koje uzorke treba očekivati. Posljednja prednost je pregled razvoja modela tijekom vremena jer se podatci skupljaju tijekom simulacije.

Jedna od prednosti je ujedno i mana a ta je da nema svrhe koristiti ovu metodu kod homogenih sustava jer nema nasumičnog odnosno promjenjivog ponašanja. Korištenje ove metode je izrazito skupo po računalnu opremu jer se za svakog agenta zasebno rade velike kalkulacije. [6]

Ovakav način modeliranja simulacija se koristi u raznim područjima, kao što su:

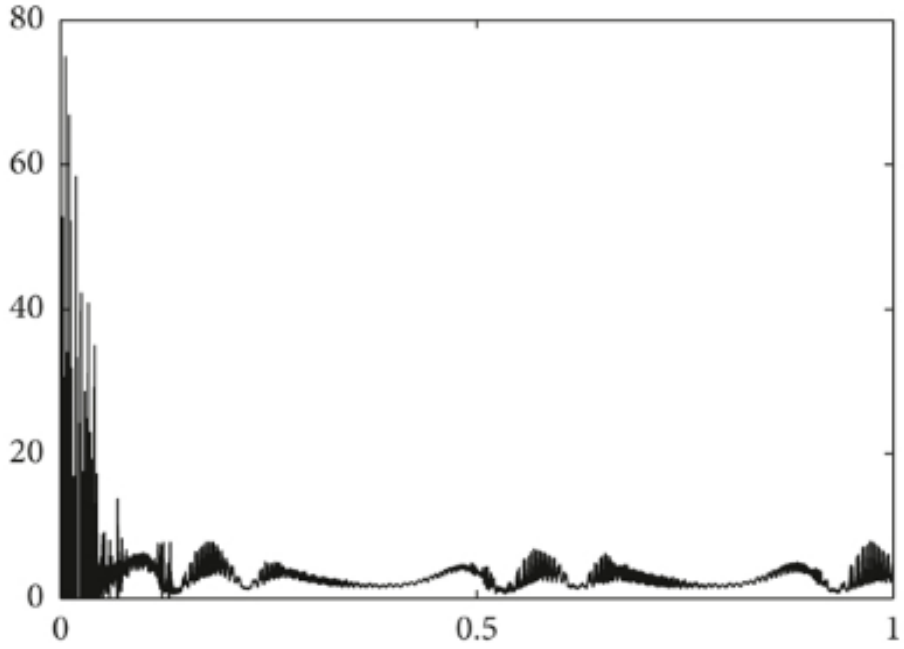
- Organizacija i poslovanje (Proizvodnja, opskrbni lanci, osiguranja, …)
- Ekonomija (Umjetna inteligencija u prodaji, Internet prodaja, ...) – tu agenti mogu predstavljati firme, pojedince ili tržišta kako bi simulirali ekonomske procese, tržište cijena, ponašanje potrošača. Time se omogućuje analiza utjecaja ekonomske politike kao i razumijevanje ponašanja tržišta u različitim scenarijima.
- Kultura i društvo (drevne civilizacije, građanski neposluh, …) – u ovom slučaju agenti predstavljaju drevne civilizacije koje uz nametnute zakone interakcijom prikazuju djelovanje istih, čime se može pokušati razumjeti razvoj ljudi kroz povijest.
- Vojska (Vođenje i upravljanje, …) – agenti u ovom slučaju mogu predstavljati vojsku, teroriste ili neprijateljsku vojsku čijom interakcijom se može predvidjeti strategija suprotne strane.
- Biologija (modeliranje i analizu bioloških sustava kao što su populacije životinja, širenje bolesti, …) – ovdje agenti mogu predstavljati životinje, mikroorganizme ili čak dijelove ekosustava čime se pospješuje razumijevanje dinamike ekosustava i ekološke promjene.
- Društvene znanosti (širenje informacija, društvene mreže, interakcije između ljudi) – tu agenti predstavljaju pojedince u društvu i njihove interakcije čime se omogućuje proučavanje socio-psiholoških procesa.
- Transport (analiza saobraćaja, analize gužvi u prometu, planiranje transportnih sustava) – u ovom slučaju agenti mogu predstavljati vozila, vozače ili putnike. Ideja je analizom ponašanja pojedinačnih vozila i njihovih interakcija možemo razumjeti kako dolazi do gužvi, optimizirati rute i planirati izgradnju novih puteva i sustava javnog prijevoza.

2.2.3. Dinamičko modeliranje

Dinamičko modeliranje procesa je donedavno bilo nerazvijeno. Razlog tome je kompleksnost izračuna. Razvoj dinamičkog modeliranja je rastao paralelno s razvojem tehnologije zbog koje se danas ovakav pristup sve češće koristi. Prva uporaba računalne simulacije ovakvog

tipa bila je u zrakoplovnoj industriji, a danas se koristi na području matematike, mehanike fluida, zdravstva, itd.

Dinamičko modeliranje predstavlja vremenski aspekt sustava pri čemu nadgleda kontrolne elemente kroz koje se ponašanje objekata može razumjeti tijekom vremena. Dinamički model uzima u obzir ne samo ulazne podatke već i izlazne u svim prethodnim točkama u proteklom vremenu. Prema tome, izlaz dinamičkih modela ovisi o vremenu, odnosno, ulazni podatci (varijable) su u funkciji vremena. Ovakve modele treba koristiti ako je poznato da se proces mijenja tijekom vremena, zbog mogućih trošenja i vijeka trajanja. Drugim riječima, treba ga koristiti za modeliranje samo za one objekte s određenim životnim ciklusom ili one koji pokazuju takvo ponašanje. Izlaz je predstavljen skupom dijagrama stanja (Grafikoni stanja). Svaki od tih dijagrama stanja (i njihovi pod-dijagrami) modelira sve moguće sekvence stanja i događaja dopuštenih za jednu klasu objekta kao odgovor na vanjske i unutarnje događaje i može se promatrati kao predložak ponašanja za klasu objekata koju on modelira. Drugim riječima, dijagram stanja prikazuje stanja i prijelaze između stanja kroz koja objekt prolazi tijekom svog životnog vijeka, zajedno sa njegovim odgovorima na vanjske utjecaje i događaje (slika 2.3).



Slika 2.3 Odnos dva parametra u dinamičkoj simulaciji

Prednosti ovakvog pristupa su:

1. Modeliranje promjena u vremenu: Dinamičko modeliranje se fokusira na promjene i evoluciju sustava tijekom vremena. Ovo je ključno za razumijevanje kako se sustavi razvijaju, rastu, opadaju i kako se ponašaju u različitim fazama razvoja.

2. Proučavanje uzročnih veza: Dinamičko modeliranje omogućuje identifikaciju uzročnih veza i interakcija između različitih komponenti sustava. Ova prednost pomaže u razumijevanju načina na koji promjene u jednom dijelu sustava mogu utjecati na ostatak sustava.
3. Proaktivno planiranje: Dinamički modeli omogućuju simuliranje različitih scenarija i predviđanje budućih stanja sustava. Ova prednost je od važnog značaja za donošenje odluka koje su potkrijepljene informacijama i razvoj strategija.[7]
4. Optimizacija i simulacija: Dinamičko modeliranje se često koristi za optimizaciju resursa, planiranje projekata i simulacije različitih strategija. Tako se omogućuje bolje iskorištavanje resursa i efikasnije upravljanje.
5. Kvalitetna osnova za donošenje odluka: Modeli koji su modelirani po ovom pristupu pružaju kvalitetne informacije koje podržavaju donošenje ispravnih odluka i omogućuju identifikaciju najboljih pristupa za postizanje ciljeva.

Nedostaci ovog pristupa su:

1. Složenost modeliranja: Dinamičko modeliranje može biti jako složeno, posebno kada se razmatraju veliki i kompleksni sustavi. Razvoj i održavanje ovih modela zahtijeva značajnu stručnost i resurse.
2. Potreba za podatcima: Efikasno dinamičko modeliranje zahtijeva točne i izdašne podatke za validaciju i kalibraciju modela. Nedostatak kvalitetnih podataka može ograničiti upotrebu ovog pristupa.
3. Ograničena preciznost: Dinamički modeli su pojednostavljenje reprezentacije stvarnih sustava. U nekim slučajevima, ova pojednostavljenja mogu rezultirati manjom preciznošću i točnošću modela.
4. Vremenski zahtjevi: Razvoj i izvođenje dinamičkih modela može zahtijevati značajno vrijeme i računalne resurse, posebno ako se radi o kompleksnijim sustavima.
5. Ograničenje validnosti modela: Modeli dinamičkog modeliranja su validni samo za period i uvjete za koje su kreirani. Promjene u okruženju ili parametrima sustava mogu značajno utjecati na istinitost modela.[7]

Kao i prethodni pristupi modeliranja i ovaj pristup se primjenjuje na raznim područjima. To su sljedeća područja:

1. Ekonomija: Omogućuje analizu efekta ekonomske politike kao i predviđanje budućih promjena na tržištima.
2. Inženjering i proizvodnja: Koriste se u proizvodnji za modeliranje i analizu procesa proizvodnje, upravljanjem lancem opskrbe te planiranje resursa. Ovakav pristup

omogućuje identifikaciju efikasnijeg proizvodnog procesa, optimizaciju resursa i planiranju kapaciteta.

3. Zalihe i logistika: Ovakav pristup koristan je za upravljanje zalihama, praćenje i optimizaciju logističkih lanaca te distribuciju proizvoda. On omogućuje bolje razumijevanje potreba za zalihama, praćenje rokova isporuke i reagiranje na promjene u potražnji.
4. Epidemiologija i javno zdravstvo: Dinamičko modeliranje je ključno u modeliranju širenja bolesti, procjenu efikasnosti preventivnih mjera i planiranje odgovora na epidemije.
5. Urbanizam i planiranje gradova: Dinamičke simulacije se koriste za planiranje razvoja gradova, upravljanje prometom i ocjenu potrebe za razinom infrastrukture.
6. Obrazovanje i obuka: Ovakve simulacije omogućavaju učenicima i studentima interaktivno učenje i eksperimentiranje s različitim scenarijima.[6]

3. SIMULACIJA DISKRETNIH DOGAĐAJA

U modernog svijetu gdje se procesi složenog sustava ne mogu lako razumjeti samo analizom i matematičkim modeliranjem, simulacija diskretnih događaja uzima ključnu ulogu za proučavanje složenih dinamičkih sustava. Simulacija diskretnih događaja kvantitativno predstavlja stvarni svijet simulirajući njegovu dinamiku prema događaj-po-događaj tehnici te generira detaljan izvještaj o njegovoj izvedbi. S razvojem računalne tehnologije ova metoda postaje jedna od glavnih računalnih alata. Ova metoda omogućuje modeliranje i analizu složenih procesa koji se mijenjaju tijekom vremena, a česta je primjena ove metode u područjima proizvodnje, logistike, telekomunikacije i epidemiologije.[6]

Simulacija diskretnih događaja je relativno nova tehnologija koja se koristi za modeliranje sustava gdje se promjene događaju u diskretnim trenucima vremena, obično kao reakcija na određene događaje ili akcije. Ovaj metoda modeliranja se razlikuje od kontinuiranih modela gdje se vrijednosti varijabli mijenjaju neprekidno tijekom vremena. U simulaciji diskretnih događaja, sustav se promatra u trenutcima kada se događaju relevantni događaji, što nam omogućuje detaljnije proučavanje interakcije entiteta i efekata u složenim sustavima. [8]

U procesu izrade simulacijskog modela diskretnih događaja definiramo modele sustava koristeći događaje, entitete i resurse. Događaji su ključni trenutci u sustavu koji pokreću promjene, entiteti su objekti koji prolaze kroz promjenu u sustavu, a resursi su oni koji obavljaju određene aktivnosti nad entitetima. Tijekom simulacije moguće je pratiti kako se entiteti kreću kroz sustav, integriraju s resursima te pokreću događaje.

Ova metoda modeliranja simulacija se razvijala polako kroz povijest a od 1960-ih progresivno napreduje. Od te godine razvili su se razni sustavi od strane industrije i znanosti kako bi se lakše suočavali s problemima i preprekama u proizvodnji. Kroz povijest su razvijena 4 software-a modeliranja simulacije:

1. Generacija (kasne 1960-e): Programiranje u programskim jezicima kao što je „Fortian“. Princip rada ove prve generacije je bio da je potrebno programirati model, logiku rada te kod koji će upravljati događajima i aktivnostima, odnosno kod koji će predstavljati srce modela.
2. Generacija (kasne 1970-e): Razvijeni su simulacijski jezici koji imaju naredbe poput gore navedenog „srca“ modela za kontrolu događanja i aktivnosti. Model se sastavlja u simulacijskom jeziku te se povezuje s potprogramima za proizvodnju kompletnog modela.

3. Generacija (rane 1980-e): Simulacijski jezici već sadržavaju u sebi „srce“ modela koje povezuje i tvori završni oblik model. Ovim postignućem je smanjeno potrebno vrijeme za izradu modela ali je i dalje bilo potrebno da korisnik ovlada sa svim aspektima mehanizma sustava.
4. Generacija (kasne 1980-e): Interaktivan paket simulacijskog software-a koji pruža stvaranje onog što vidiš, također dopušta modifikaciju istog u bilo koje vrijeme, ubrzana izrada, ubrzana analiza, itd. Ovakav software je bio razumljiv i lak za uporabu svima kojima je bio dostupan.

Od tada ne postoji značajan razvoj software-a u usporedbi s 4. generacijom.[9]

3.1. Primjena simulacije diskretnih događaja u proizvodnji

Simulacija diskretnih događaja se tradicionalno koristi u proizvodnji. Za vrijeme četvrte generacije dolazi do rapidnog razvoja napredne proizvodne tehnologije u industrijaliziranim državama kao što su: CAD (Računalom potpomognuto projektiranje), CAM (Računalom potpomognuta proizvodnja), AGV (Automatski vođenja vozila), Robotics (Robotika), FMS (Fleksibilni proizvodni sustavi) te CIM (Računalno integrirana proizvodnja). Na isti način se razvijala i tehnologija simulacije diskretnih događaja. Mnoge velike kompanije su uvelike ulagale u razvoj novih tehnologija kako bi si pospješili proizvodnju i učinili ju što fleksibilnijom. Software-i za izradu simulacija diskretnih događaja je bio glavni alat voditeljima koji je pomagao pri donošenju važnih odluka. Svaki od njih želi pospješiti proizvodnju u vidu većeg protoka, kraćeg vremena izrade, manje WIP-a (*work in process*), itd. Kroz simulaciju ovakvog pristupa mogu se pretpostaviti ponašanja proizvodnog procesa kroz različite postavljene početne uvjete. Simulacija će provoditi „what-if“ analize scenarija a sve kako bi se identificirao bolji prostorni raspored strojeva i omogućilo jednostavnije donošenje odluka.

Primjena simulacije diskretnih događaja u proizvodnji se raspodijelila na sljedeća područja proizvodnje:

1. Projektiranje i evaluacija novih proizvodnih procesa
2. Poboljšanje izvedbe postojećih proizvodnih procesa
3. Uspostava optimalne operativne politike
4. Algoritam za potporu planiranja i terminiranja proizvodnje[10]

3.2. Tecnomatix Plant Simulation

U ovom radu korišten je softver Tecnomatix Plant Simulation koji je moćan alat za modeliranje, simulaciju i optimizaciju proizvodnih procesa. Razvijen je od strane tvrtke Siemens PLM Software. Ovaj alat postaje sve popularniji u industriji jer pomaže kompanijama da bolje razumiju svoje proizvodne operacije, identificiraju slabosti i poboljšaju efikasnost proizvodnje.[13]

Glavne karakteristike softvera Tecnomatix Plant Simulation:

1. Modeliranje proizvodnih procesa: Tecnomatix Plant Simulation omogućuje korisnicima da modeliraju složene proizvodne procese koristeći intuitivno grafičko sučelje. Korisnici mogu kreirati detaljne modele svojih postrojenja, transportnih sustava, obradnih ćelija i proizvodnih linija.
2. Dinamička simulacija: jedna od glavnih prednosti ovog softvera je sposobnost dinamičke simulacije. Korisnici mogu pratiti kretanje resursa, radnika i proizvoda u stvarnom vremenu kako bi bolje razumjeli kako se procesi odvijaju. Ovo pomaže pri identifikaciji problema i optimizaciji protoka rada.
3. Optimizacija proizvodnih operacija: Tecnomatix Plant Simulation omogućuje korisnicima eksperimentiranje s različitim scenarijima kako bi optimizirali svoje proizvodne operacije. Također mogu se testirati različite konfiguracije postrojenja, vremenske rasporede i kapacitete kako bi se pronašlo najbolje rješenje za referente potrebe i probleme.
4. Analiza performansi: Softver pruža napredne analitičke alate koji omogućuju korisnicima da analiziraju performanse svojih proizvodnih procesa. Mogu se pratiti stvarni podatci, analizirati vrijeme čekanja, stvarati izvještaje i identificirati moguća potencijalna poboljšanja.[13]

Prednosti ovog softvera:

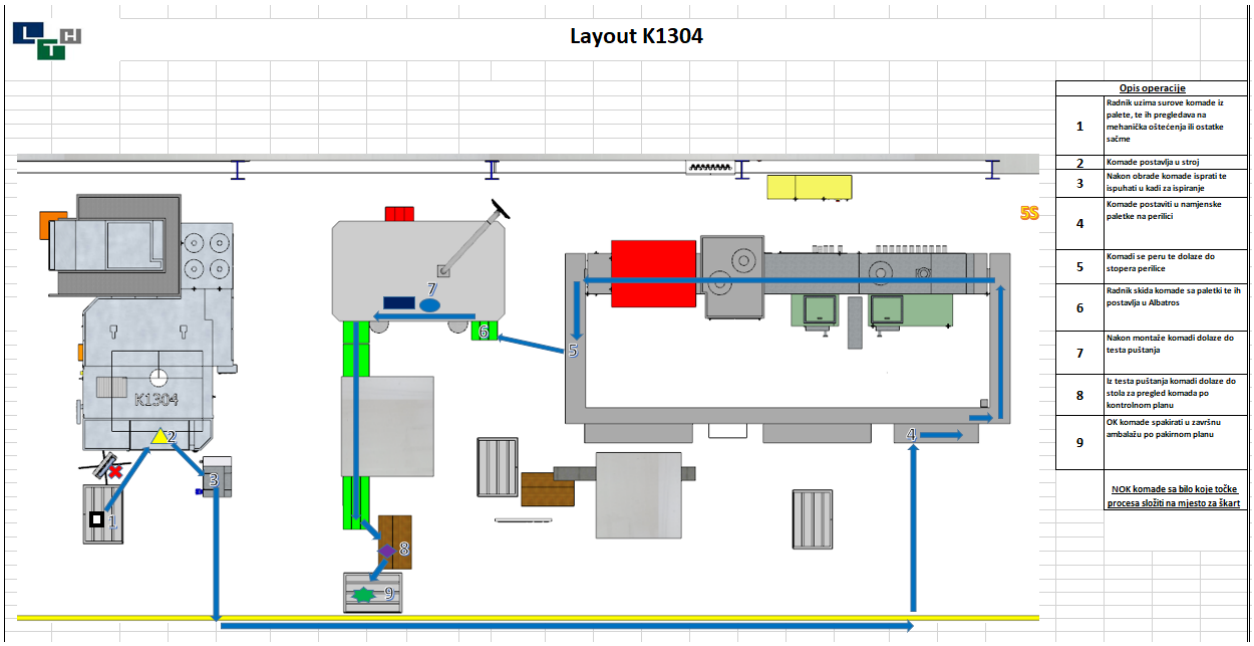
1. Smanjenje troškova: Korištenje ovog softvera pomaže smanjiti troškove proizvodnje identificiranjem nepotrebne potrošnje resursa, optimizacijom radnih stanica i smanjenjem zastoja.
2. Povećanje efikasnosti: Optimizacija proizvodnih operacija rezultira povećanjem efikasnosti, smanjenjem vremena ciklusa i boljom iskorištenošću resursa.
3. Bolja planiranja: Tecnomatix Plant Simulation omogućuje bolje planiranje resursa i vremenskih rasporeda, što pomaže u smanjenju zastoja i poboljšava upravljanje proizvodnim operacijama.[13]

4. SIMULACIJSKI MODEL DIJELA PROIZVODNOG POGONA

U ovom dijelu je opisana izrada simulacijskog modela proizvodnog pogona. Dio pogona je uzet iz tvrtke LTH Metal Castings u Benkovcu. Ovakav dio pogona koristi se za izradu poklopca mjenjača koji ima tri izvedbe (u daljnjem dijelu teksta: pozicije) koje se dizajnom ne razlikuju previše a nazivi su 840.39, 840.40 i 840.41. Trenutne količine su nekoliko tisuća komada jer je pozicija još u procesu uhodavanja. Povećanje količina se očekuje sljedeću godinu na nekoliko desetaka tisuća komada a godinu iza na nekoliko stotina tisuća komada. Iz tog razloga javlja se potreba za automatizacijom ovog proizvodnog sustava. Da bi uopće mogli razmatrati buduću automatizaciju proizvodnog sustava potrebno je izraditi simulacijski model sadašnjeg proizvodnog sustava. U daljnjem dijelu teksta opisani su proizvodni sustav i proces izrade simulacijskog modela sadašnjeg proizvodnog sustava.

4.1. Proizvodni sustav za izradu poklopca mjenjača

Ovaj proizvodni sustav se sastoji od tri obradna centra, protočne perilice, pokretne trake te dva namjenska uređaja. Tri obradna centra, namjenska protočna perilica i dva namjenska uređaja. Opisat će se strojevi prema hodogramu prikazanom na slici 4.1.



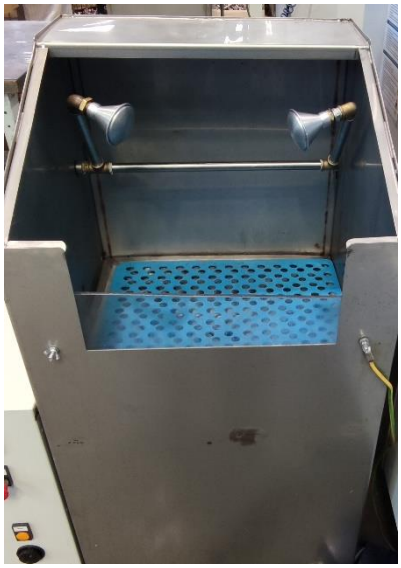
Slika 4.1 Hodogram promatranog dijela proizvodnog pogona

Na broj 1 prema hodogramu dovoze se obradci koji su prethodno prošli operacije lijevanja i pjeskarenja. Transport komada do obradnog centra se obavlja uz pomoć viličara. Kad su komadi dovezeni na za to predviđeno mjesto radnik može započeti s obradom. Strojna obrada se vrši na obradnim centrima S1, S2 i S3 (slike 4.2 i 4.3). Njihova cijena je bila oko 800 000 eura. Riječ je peto-osnim dvovretenim strojevima. Imaju dva okretna stola koji svaki na sebi imaju, za ovu poziciju, stezni uređaj s po 4 stezna mjesta. Također, posjeduju i magazin alata u koji stane 8 alata, odnosno kako je riječ o dvovretenom stroju 16. Operacije koje se vrše su: obrada glodanjem, bušenje, valjanje navoja te skidanje srha. Vrijeme obrade iznosi 56 sekundi. U daljnjem dijelu rada smo postavili ukupno vrijeme na 60 sekundi zbog okreta stola koji iznosi četiri sekunde. Sva tri stroja su ista te je slikama 4.2. i 4.3 prikazan izgled samog stroja.

Slika 4.2 Obradni centar – prednja strana

Slika 4.3 Obradni centar – bočna strana

Nakon strojne obrade komadi se ispiru u kadi, slika 4.4 Nakon ispiranja radnik suši komade s komprimiranim zrakom te ih stavlja na kolica kojima ih prevozi do pokretne trake. Prije postavljanja na traku, komadi se postavljaju po dva na namjensku paletu. Komadi se s paletama postavljaju na pokretnu traku.

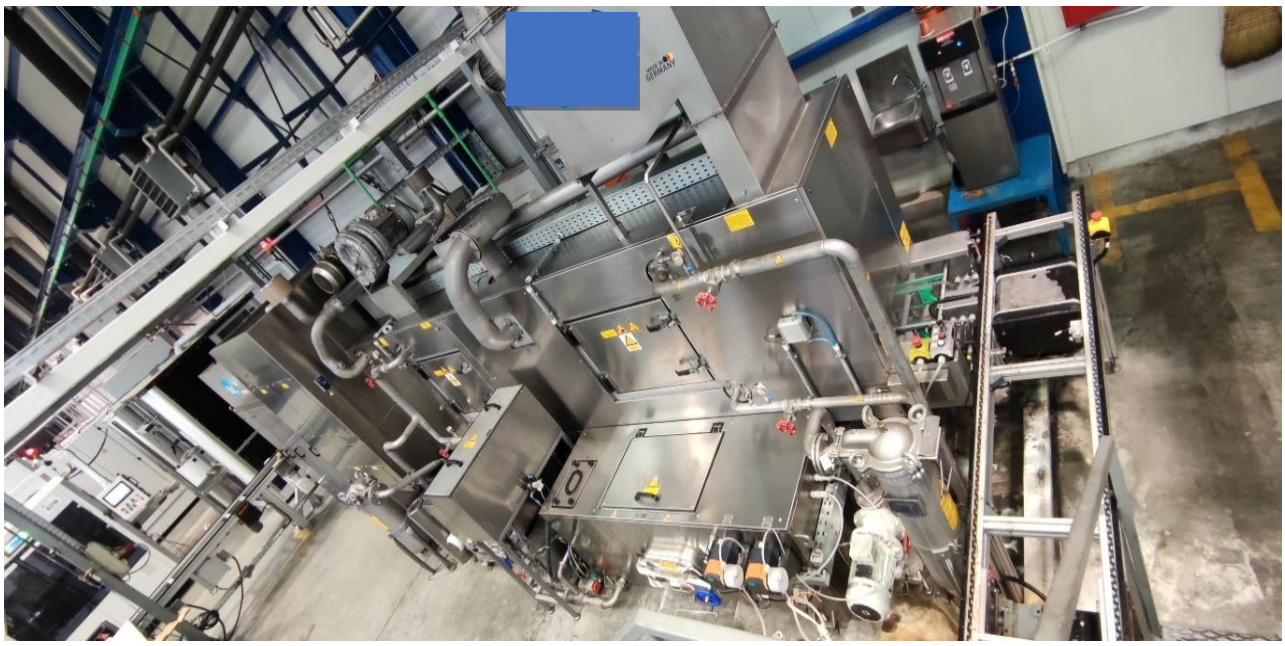


Slika 4.4 Kada za ispiranje

Pokretna traka se vrti brzinom od 1 m/s, a dimenzije su navedene u daljnjem dijelu teksta. Transportom preko pokretne trake komadi stižu u perilicu. Namjenska perilica je protočna perilica u koju stane odjednom 12 namjenskih paleta odnosno 24 komada. Prikaz pokretne trake i perilice se nalazi na slikama 4.5 - 4.7.

Slika 4.5 Namjenska protočna perilica – prednja strana

Slika 4.6 Namjenska protočna perilica, pokretna traka i prijelaz za servisiranje

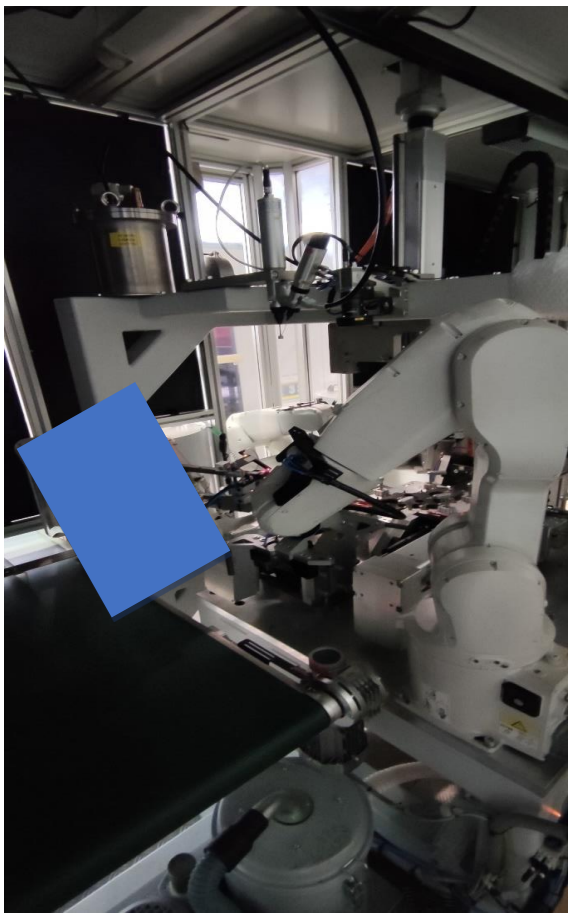


Slika 4.7 Namjenska perilica – ulaz namjenskih paleta

Kada komadi izađu iz protočne perilice ručno ih se podiže s pokretne trake (slika 4.8) te ih se postavlja na ulaznu traku u namjenski stroj. Kada komadi dođu do ruba ulazne trake namjenskog uređaja, zaustavljaju se te na senzor te stroj uzima sliku s kamerom kako bi odredio položaj komada na pokretnoj traci. Izgled vanjski i unutarnji namjenskog stroja prikazan je na slikama 4.9 i 4.10.

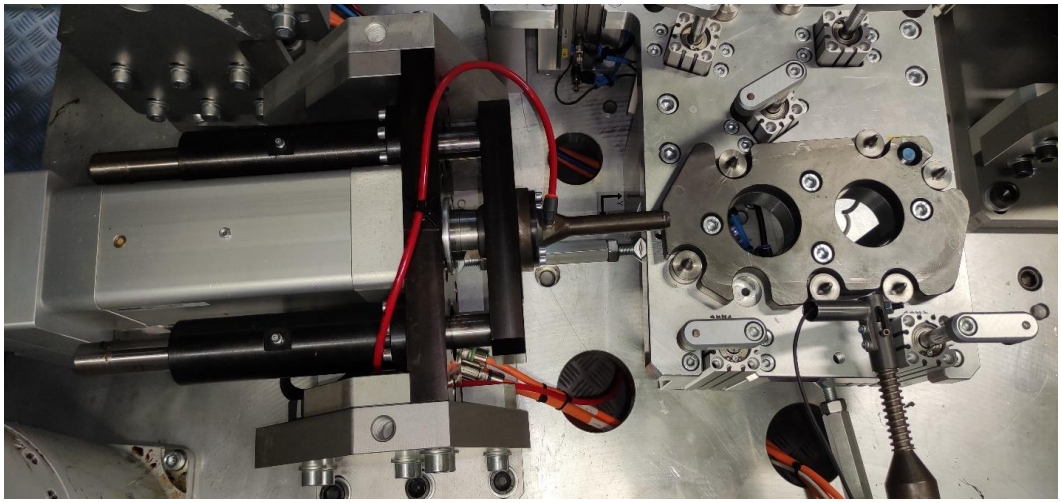
Slika 4.8 Pokretna traka

Slika 4.9 Namjenski uređaj 1

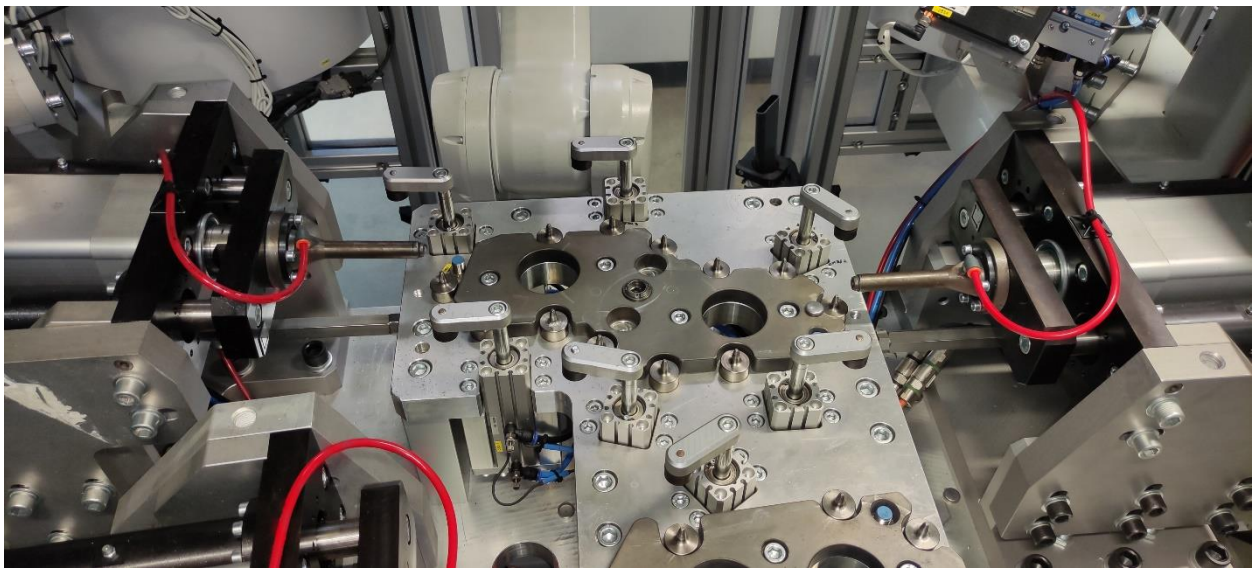


Slika 4.10 Ulazna traka

Robotska ruka uzima komad i nanosi ljepilo. Nakon nanošenja ljepila slijedi kontrola nanošenja ljepila i ako je pravilno naneseno komad se postavlja na stezni pribor, slike 4.11 i 4.12, gdje se utiskuju konektor i čep. U stroju postoje tri stezna mjesta, po jedno za svaku poziciju.

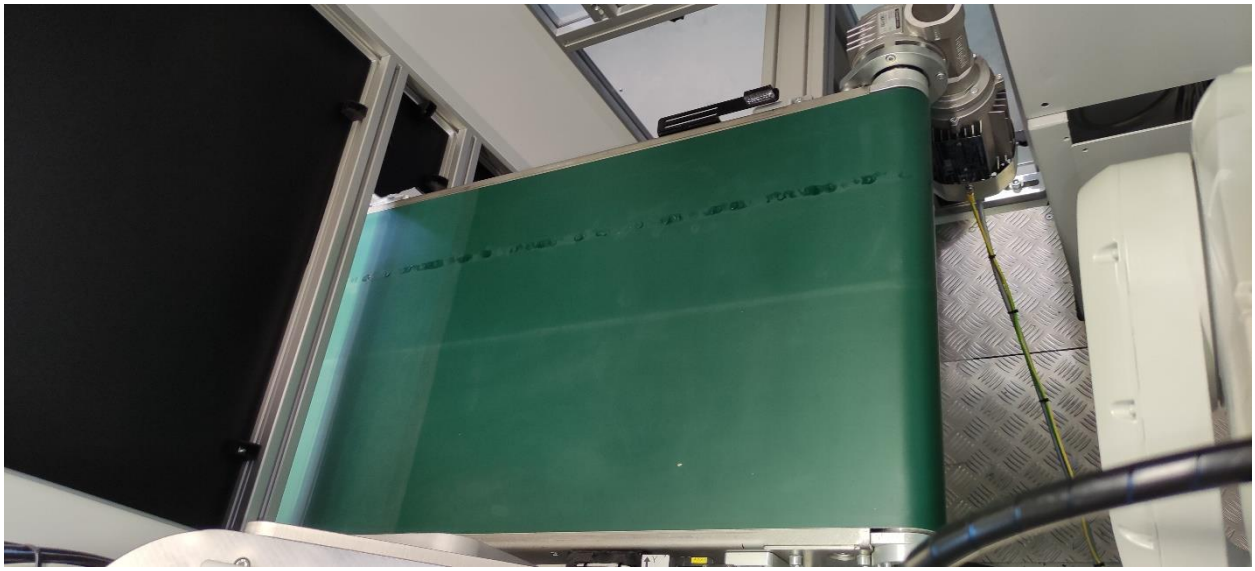


Slika 4.11 Stezno mjesto pozicije 840.41

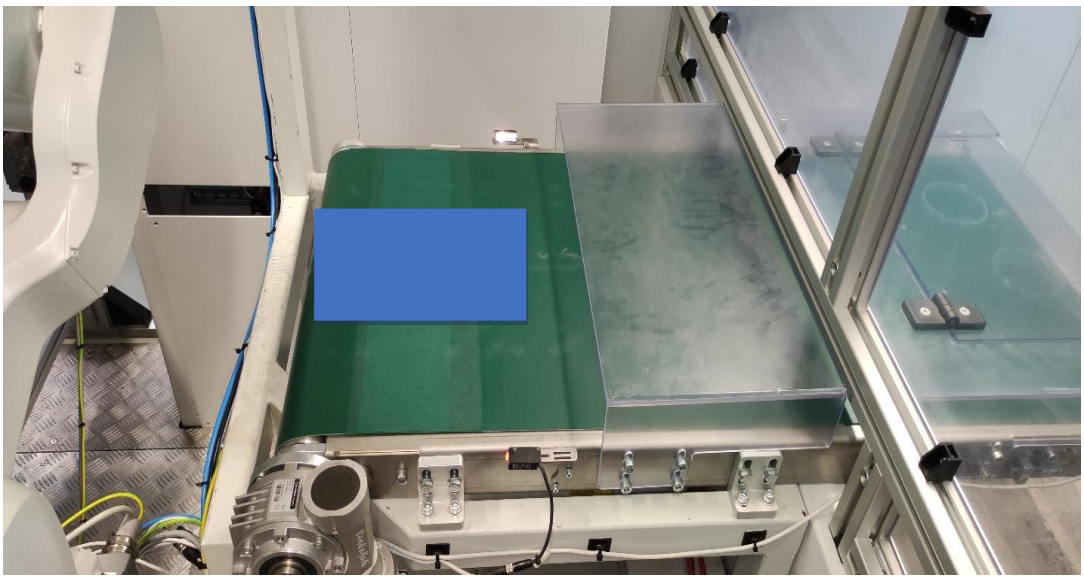


Slika 4.12 Stezna mjesta pozicija 840.39 i 840.40

Nakon utiskivanja druga robotska ruka uzima komad i izvršava kontrolu utiska konektora i čepa te ako je zadovoljavajuća komad se postavlja na izlaznu traku, slika 4.13, prema namjenskom stroju za propusnost. U slučaju da komadi nisu zadovoljavajući prilikom kontrole, oni se prenose na drugu izlaznu traku na koju se odlažu škartni komadi (slika 4.14).



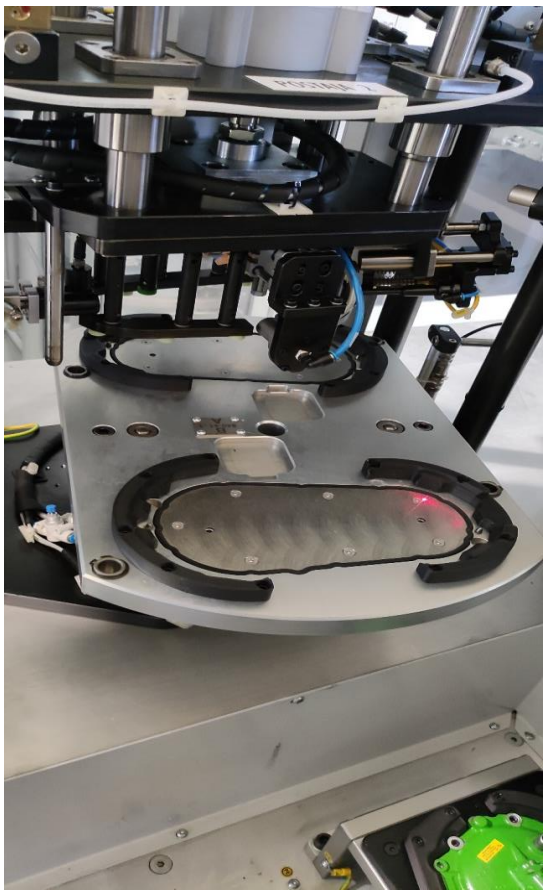
Slika 4.13 Izlazna traka – OK komadi



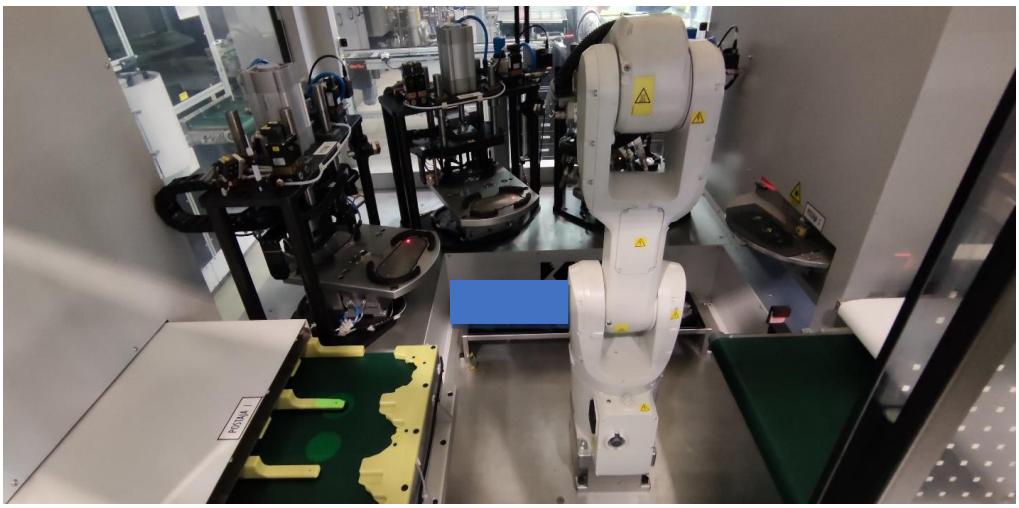
Slika 4.14 Izlazna traka – NOK komadi

Kada komad uđe u namjenski stroj za test puštanja te dostigne krajnju poziciju, robotska ruka ga kupi te manipulacijom ga prenosi na stezno mjesto gdje se komad brtvi i ispituje pod zadanim tlakom. Unutar stroja postoje po dva stezna mjesta za testiranje propusnosti za svaku poziciju. Vanjski izgled kao i unutarnji prikazani su na slikama 4.15 - 4.17.

Slika 4.15 Namjenski uređaj za test propusnosti

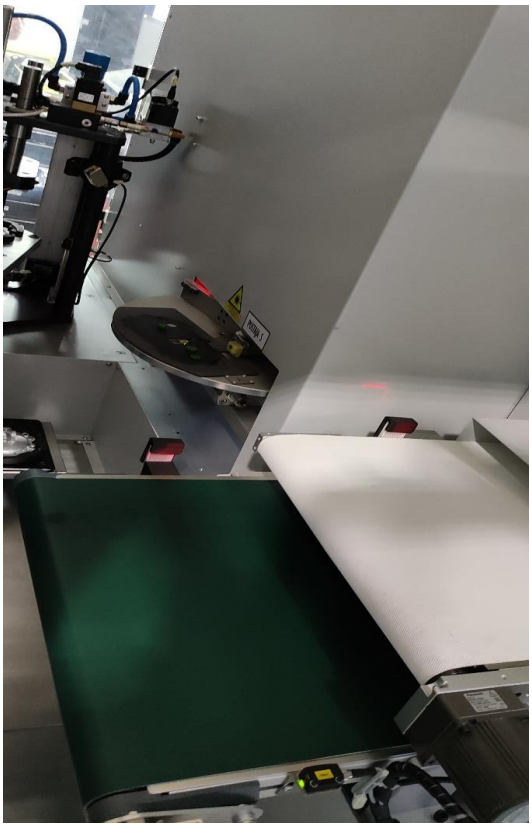


Slika 4.16 Drugi namjenski stroj i stezno mjesto



Slika 4.17 Unutrašnjost drugog namjenskog uređaja

Nakon što ispitani komad zadovoljava kontrolu propusnosti robotska ruka ga uzima i na njegovo mjesto stavlja drugi komad. Ispitani komad prenosi se do postaje za DMC (eng. *Data matrix code*) graviranje (slika 4.18). Postaja ima dva stezna mjesta. Nakon graviranja robotska ruka mijenja gravirani komad za testiranog te graviranog prenosi na izlaznu traku.



Slika 4.18 Gravirka i izlazna traka

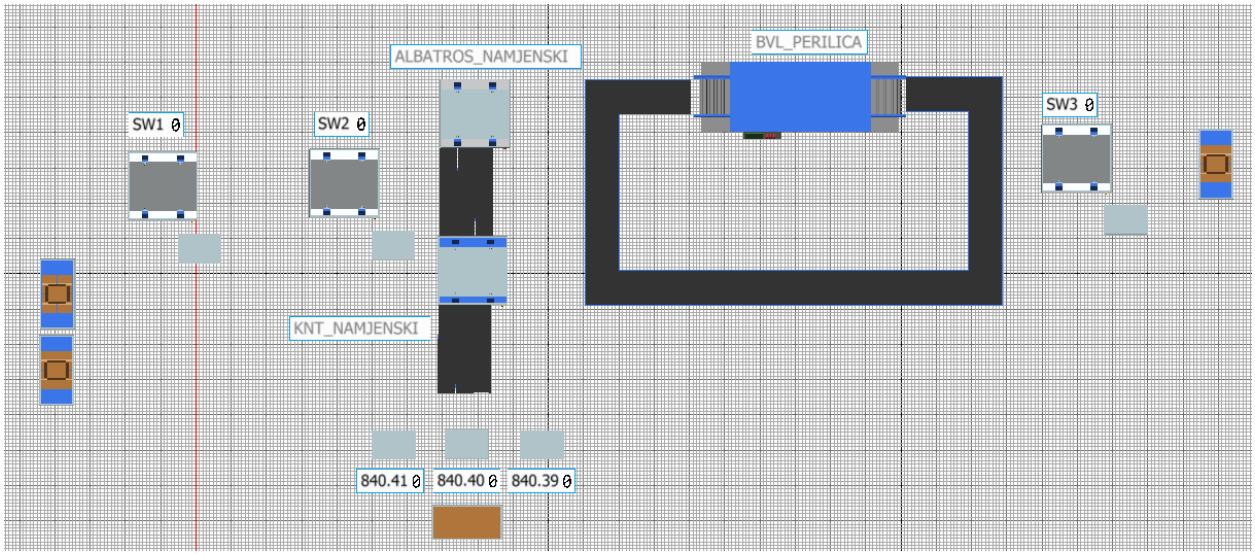
Slikom 4.19. dane su sve potrebne informacije za izradu simulacijskog modela postojećeg proizvodnog pogona za izradu poklopca mjenjača.

Br.	Aktivnost	Oprema	Vrijeme (sec)		
		Oznaka			
1.	Ulazni Bufferi	Euro paleta	-	-	-
2.	Obradni centar	SW	60	60	60
3.	Namjenska perilica	BWL	300	300	300
4.	Montažni uređaj	Alba	15	15	15
5.	Namjenski uređaj	KT	20	20	20
6.	Velika pokretna traka	-	Brzina vrtnje (m/s)		
			1		
7.	Male pokretne trake	-	0,5		
8.	Izlazni Buffer-i	Euro paleta	-	-	-

Slika 4.19 Ulazni podaci za izradu simulacijskog modela postojećeg proizvodnog pogona

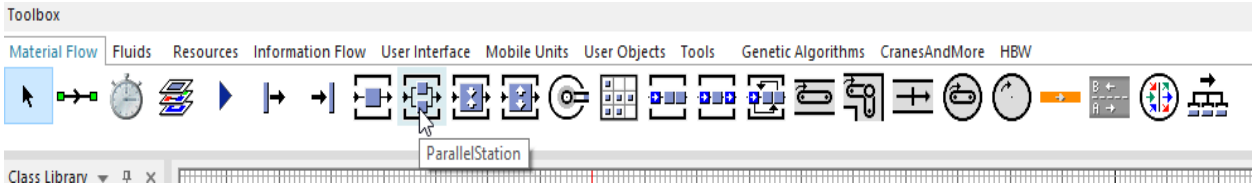
4.2. Izrada simulacijskog modela dijela proizvodnog pogona za izradu poklopca mjenjača

U ovom poglavlju opisana je izrada simulacijskog modela koji oponaša rad sadašnjeg proizvodnog sustava. Tek nakon što je simulacija izrađena može se pristupiti koncipiranju i izradi automatizacije istog. Također će biti potrebno ispitati i odrediti optimalan broj namjenskih paleta na kojima se dijelovi (u daljnjem tekstu komadi) kreću na pokretnoj traci te automatizaciju tog dijela pogona. Tlocrt i prostorni raspored izrađenog simulacijskog modela promatranog proizvodnog pogona prikazan je na slici 4.20.



Slika 4.20 Simulacijski model postojećeg proizvodnog pogona

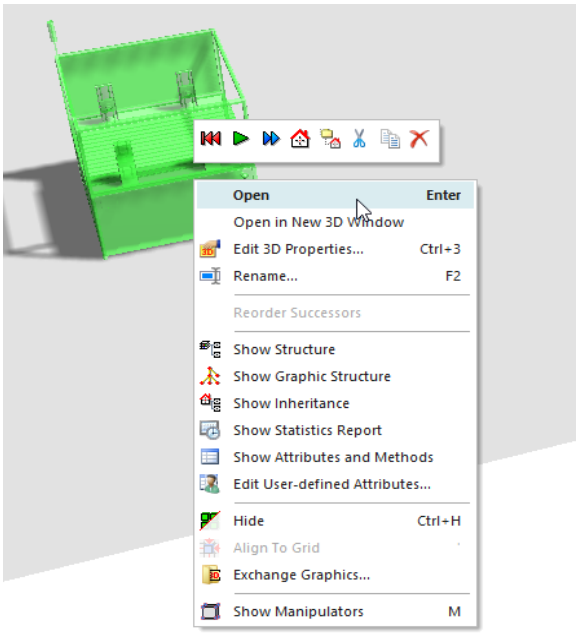
Prvi korak pri izradi simulacijskog modela je ubaciti sve potrebne objekte koji predstavljaju strojeve iz alatne trake u radni prostor (slika 4.21). To se radi po principu *drag and drop*.



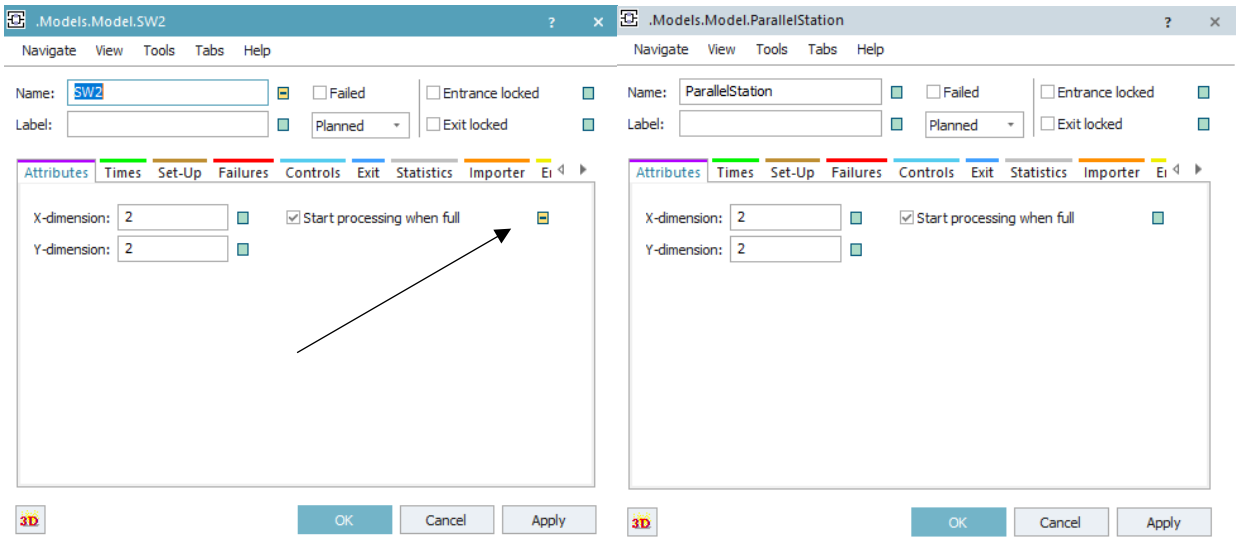
Slika 4.21 Alatna traka

Za gore navedene strojeve koristili su se entiteti „*Parallel Station*“ te smo ih prema navedenom principu ubacili u radni prostor. Osim gore navedenih strojeva, za funkcionalnu

simulaciju potrebna su nam još tri objekta *Source* i tri objekta *Buffer*. Nakon što su strojevi ubačeni potrebno ih je razmjestiti prema prostornom rasporedu u pogonu kako bi se, uz postavljene uvjete, dobili što točniji rezultati. Nakon razmještaja strojeva, potrebno je definirati postavke strojeva. Vrijeme obrade na obradnim centrima iznosi 60 sekundi odnosno jednu minutu. Vrijeme postavljamo tako da otvorimo postavke stroja dvostrukim klikom na njega ili desnim klikom potom lijevim na *Open* (otvori) padajućem izborniku (slika 4.22). Postavke stroja izgledaju kako je prikazano na slici 4.23.

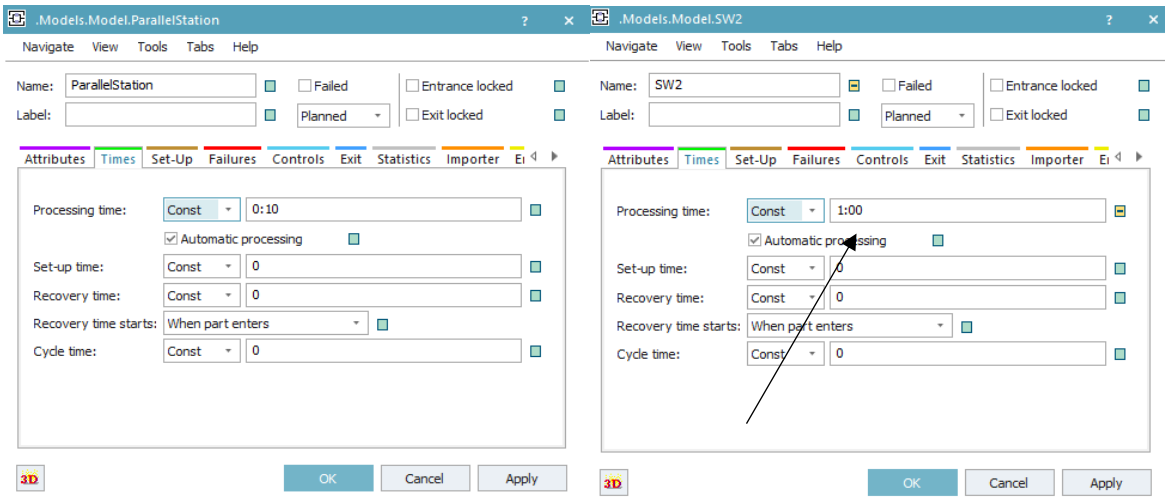


Slika 4.22 Otvaranje postavki stroja



Slika 4.23 Postavke obradnog centra – prije i poslije

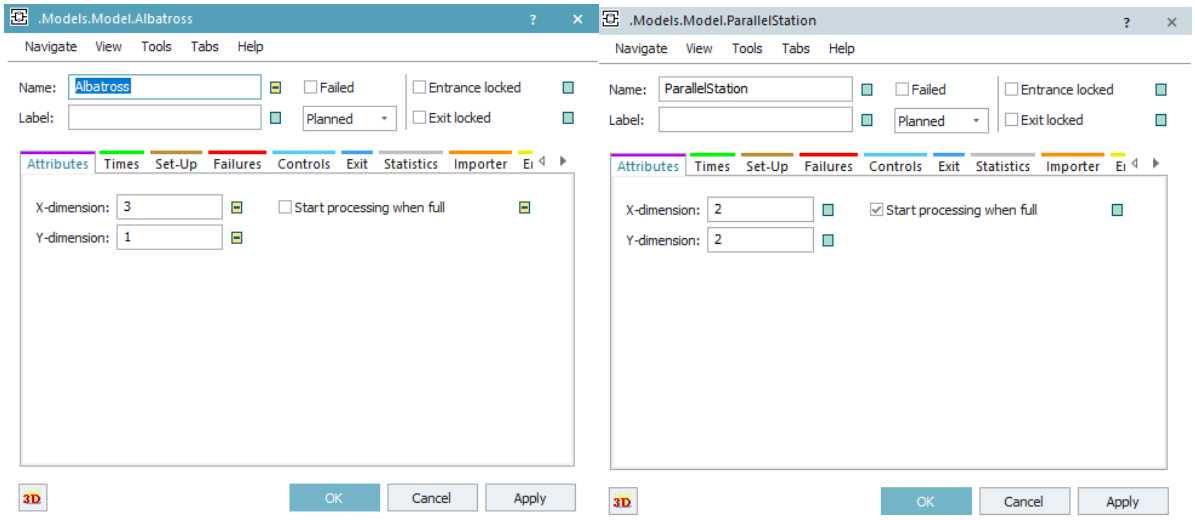
U stvarnom procesu svaki stroj posjeduje dva okretna stola na kojima se nalaze stezne naprave. Na svaku napravu se mogu staviti 4 komada, što znači da se pri jednom stezanju i operaciji obrađuju 4 komada. Iz tog razloga smo odabrali objekt *Parallel Station* koji omogućuje istovremenu obradu više od jednog komada. Pod karticom atributi (*Attributes*) postavljaju se dimenzije radnog prostora stroja. Dimenzije komada u simulaciji iznose x = 1, y = 1. Kako smo već napomenuli da se obrađuju 4 komada odjednom, dimenzije stroja trebaju omogućiti isto. Dimenzije obradnog centra su već zadane na x = 2, y = 2 odnosno tako da stanu sva 4 te ih nećemo mijenjati. Također smo odabrali postavku „*Start processing when full*“ (Započeti obradu kad se kapacitet stroja napuni) kako bi osigurali da se komadi počnu i završe obrađivati u isto vrijeme. Potom je potrebno postaviti vrijeme obrade prelaskom na karticu „*Times*“ (Vremena). Za postavke vremena obrade potrebno je pod stavkom „*Processing Time*“ (Vrijeme obrade) odabrati *Const* (slika 4.24). odnosno konstantno vrijeme obrade te u prozoru pokraj postaviti koliko iznosi to vrijeme, u našem slučaju je to 1 minuta. Ovaj korak potrebno je ponoviti za sva tri obradna centra. Za namjenske strojeve postavke se mijenjaju.



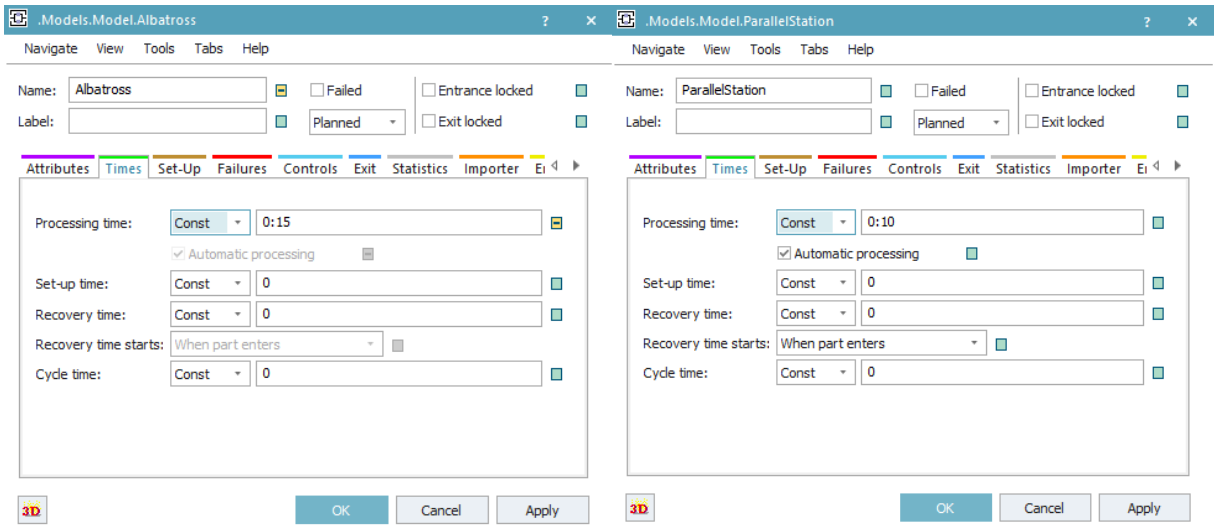
Slika 4.24 Vrijeme obrade obradnih centara – prije i poslije

Za montažni namjenski stroj također se koristi objekt *Parallel Station*, no on se razlikuje dimenzijama i vremenom obrade od obradnih centara. Postavke stroja otvara se na isti način kao i za obradne centre. U njega stanu tri komada, po jedan od svake pozicije, a to znači da ćemo dimenzije stroja postaviti sa zadanih x = 2 i y = 2 na x = 3, y = 1 (slika 4.25). U ovom slučaju je potrebno isključiti stavku „*Start processing when full*“ jer uređaj obrađuje svaki komad zasebno. U njemu se odvijaju operacije nanošenja ljepila, nadzor nanošenja istog s pomoću kamere, utiskivanje konektora i čepa te kontrola utiska istih s pomoću kamere. Ukupno vrijeme obrade u

ovom slučaju iznosi 15 sekundi. Za postavke vremena obrade potrebno je pod stavkom „*Processing Time*“ (Vrijeme obrade) odabrati *Const* (slika 4.26) odnosno konstantno te u prozoru pokraj postaviti koje vrijeme, u slučaju namjenskog uređaja za montažu vrijeme iznosi 15 sekundi.



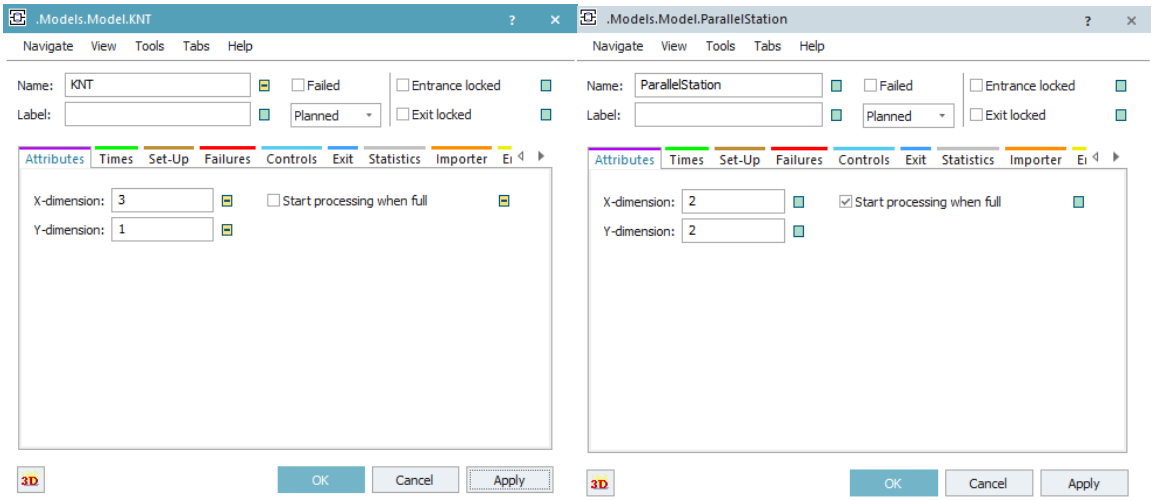
Slika 4.25 Dimenzije montažnog uređaja – prije i poslije



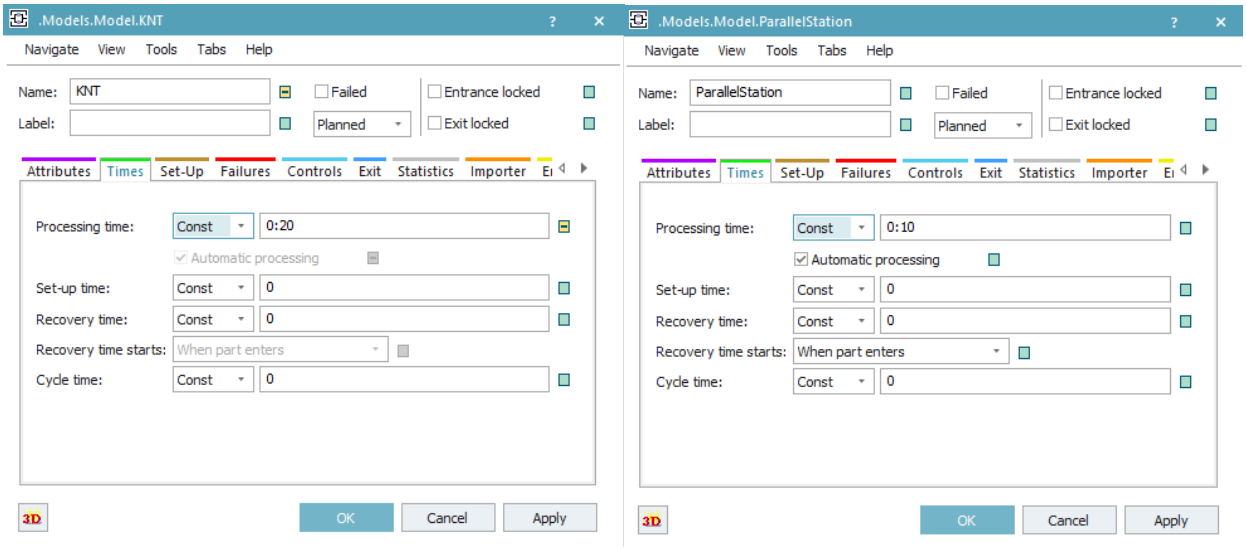
Slika 4.26 Vrijeme obrade montažnog uređaja – prije i poslije

Isto kao i u prethodnom slučaju za namjenski stroj za test propusnosti korišten je objekt *Parallel Station*. Dimenzijski je isti kao i namjenski uređaj za montažu ali se razlikuje u vremenu obrade (slike 4.27 i 4.28). Tu se odvijaju procesi kontrole puštanja brtvljenjem pod zadanim tlakom te nakon toga graviranje DMC koda koji sadrži popis svih operacija koje je komad prošao kao i vrijeme i mjesto izvršenih operacija. Postavke stroja se otvaraju po prethodno opisanom postupku.

I u ovom slučaju nije potrebno uključiti stavku „*Start processing when full*“ radi pojedinačne obrade. Ukupno vrijeme operacija iznosi 20 sekundi. Nakon toga se s pomoću pokretne trake izbacuje iz stroja na finalni pregled i pakiranje.



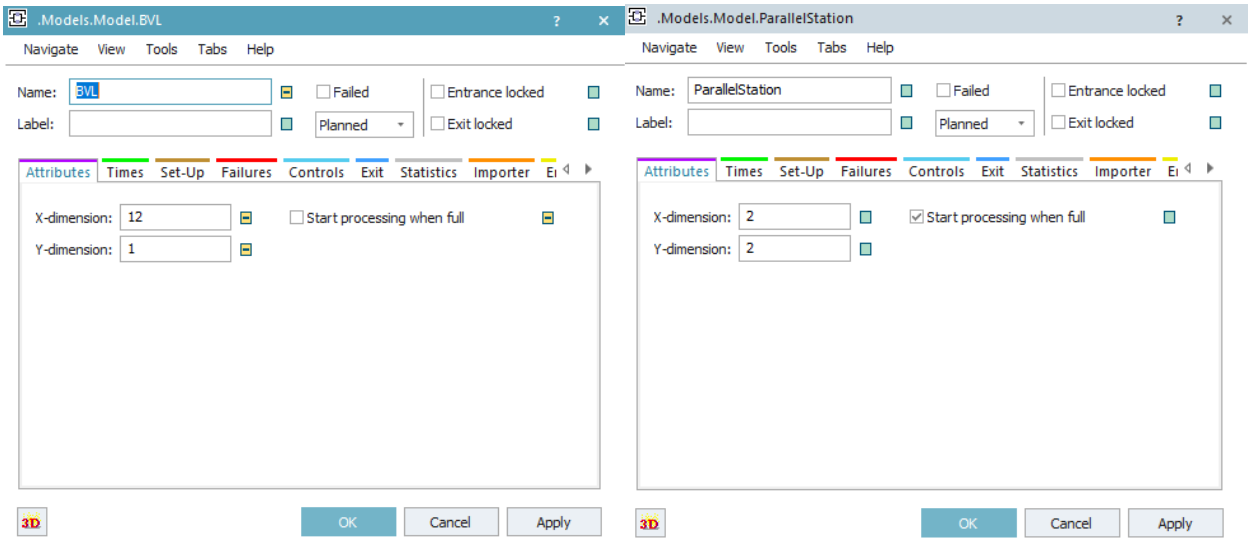
Slika 4.27 Dimenzije namjenskog uređaja za test propusnosti – prije i poslije



Slika 4.28 Vrijeme obrade test propusnosti – prije i poslije

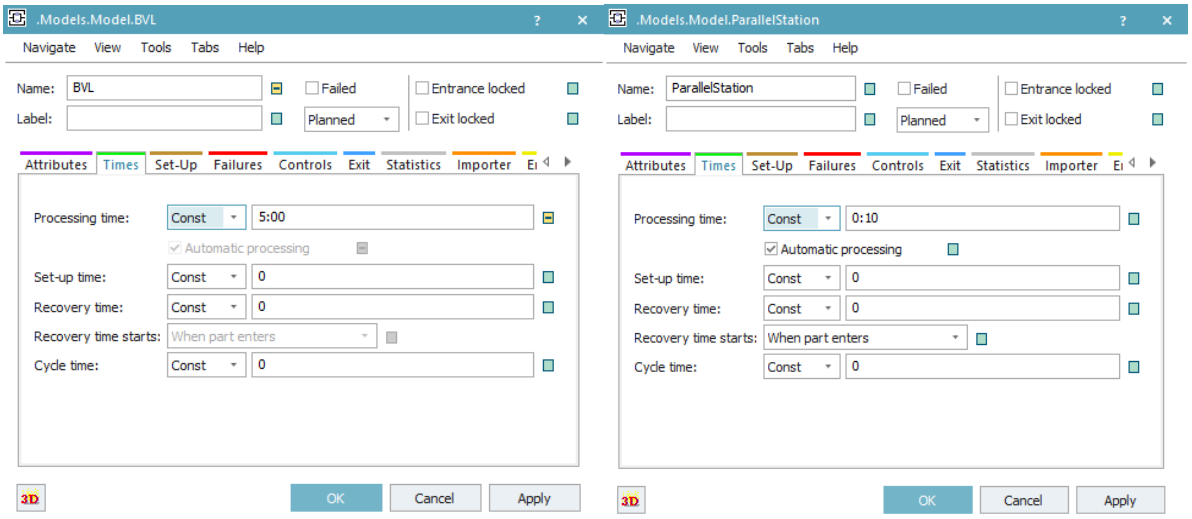
Posljednji od šest objekata *Parallel Station* je protočna perilica. Za nju se mijenjaju postavke dimenzija i vremena obrade za razliku od prethodnih strojeva. Postavke stroja otvoriti prema prethodno prikazanom primjeru. Nakon što komadi prođu kroz strojnu obradu stavljaju se po dva na namjensku paletu te se paleta postavlja na pokretnu traku koja prolazi kroz protočnu

perilicu. Ukupan broj namjenskih paleta koje mogu stati odjednom u perilicu je 12 stoga dimenzije istog izgledaju kako je prikazano na slici 4.29.



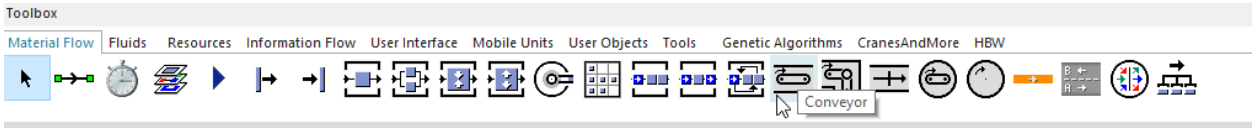
Slika 4.29 Dimenzije protočne perilice

Za x dimenziju je postavljena vrijednost 12 a za y je postavljeno 1. Kako je riječ o protočnom uređaju koji radi konstantno nismo uključili stavku „*Start processing when full*“. Vrijeme obrade iznosi pet minuta, drugim riječima pet minuta je potrebno paleti da prođe kroz perilicu. Stoga postavke vremena postavljene prema slici 4.30.



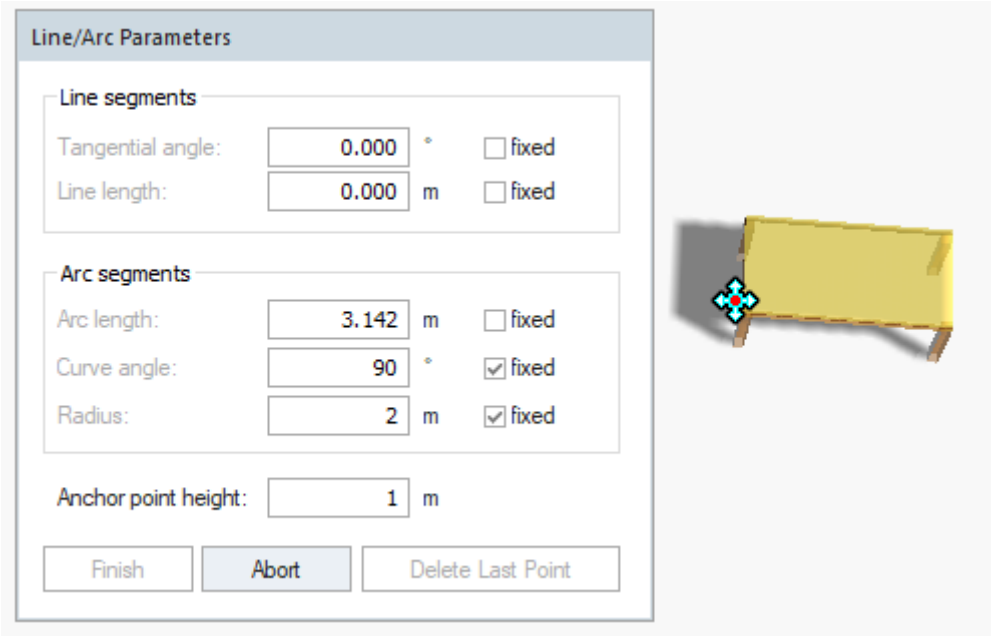
Slika 4.30 Vrijeme pranja u protočnoj perilici

Na posljetku potrebno je postaviti pokretnu traku u radni prostor te definirati njene postavke. Pokretna traka u pogonu ima oblik kvadra kako je prikazano na slici 4.31 s tim da nema jednu stranicu jer se umjesto nje nalazi protočna perilica.



Slika 4.31 Položaj pokretne trake u alatnoj traci

Pokretna traka se unosi *drag and drop* metodom iz alatne trake pod nazivom *Conveyor*. Zatim se otvara prozor s raznim postavkama i mogućnostima same trake (slika 4.32).

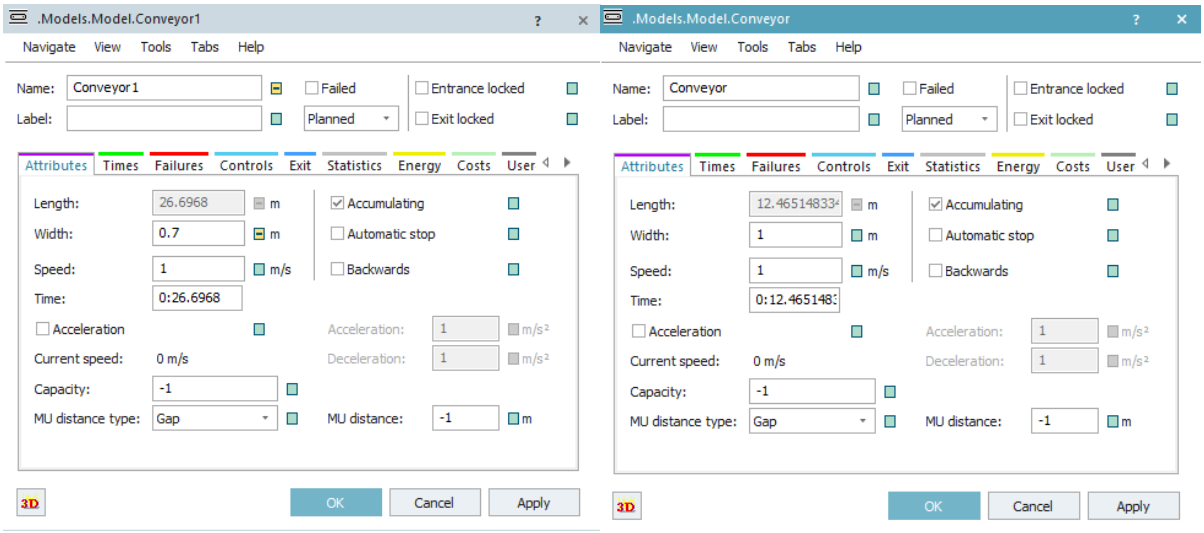


Slika 4.32 Postavke pokretne trake

U ovom slučaju promijenit će se jedino postavka duljine pokretne trake. Prvi korak je pritisnuti na željeno mjesto u radnom prostoru, tako se određuje prvo uporište pokretne trake. Nakon pritiska „vuče se“ pokazivač u željenom smjeru. Kao i stranice kod kvadra tako su i dijelovi pokretne linije pod pravim kutom. Da bi to osigurali potrebno je pritisnuti i držati stisnuti tipku „*Shift*“ na tipkovnici dok mišem pomičemo u željenom smjeru. Duljine dijelova pokretne trake iznose redom, počevši od gornjeg lijevog dijela koji stoji na izlazu iz perilice (slika 4.33) do gornjeg desnog dijela koji ulazi u perilicu, 1 m, 4,5 m, 10 m, 4.5 m i 1 m.

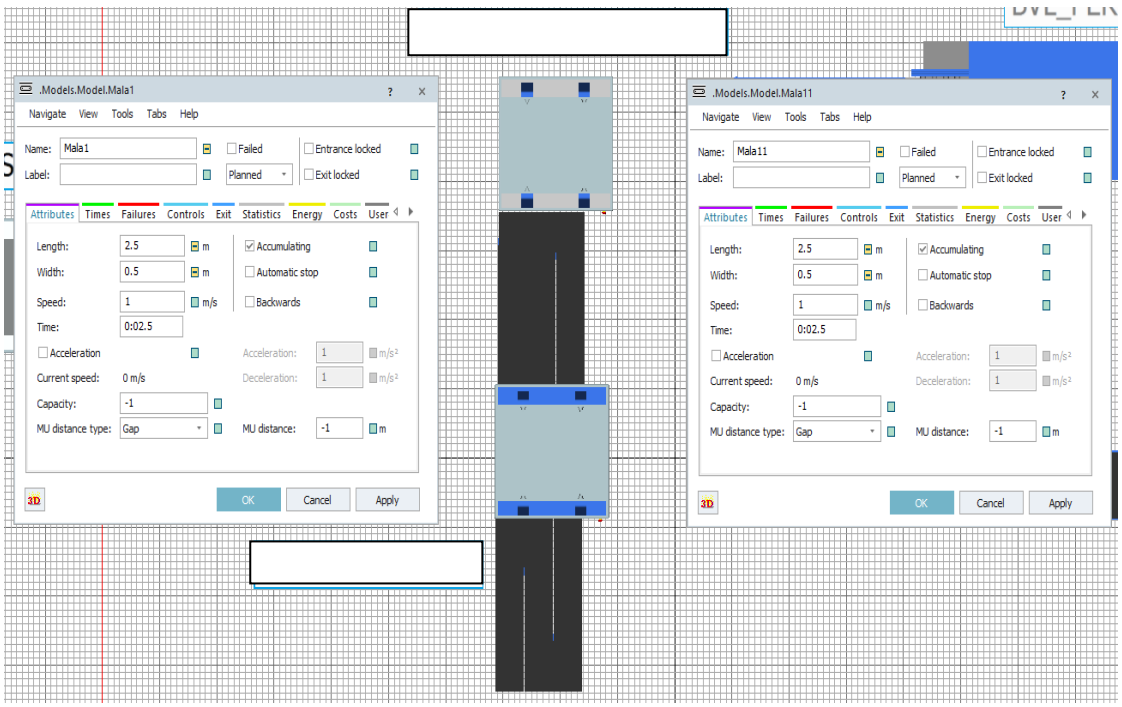
Slika 4.33 Postavke i izgled definiranog objekta Conveyor

Kako bi lakše odredili duljinu softver nudi mogućnost da pod prozorom „*Line length*“ se unese željena duljina. Ta mogućnost funkcionira tako da se nakon pritiska na željeno mjesto u radnom prostoru upiše tražena duljina čime softver automatski produljuje ili skraćuje prethodnu duljinu u traženu. Nakon toga se stisne na radni prostor u željenom smjeru koje će biti sljedeća uporišna točka. Sljedeći korak je unijeti postavke linije. Dvostrukim klikom na pokretnu traku otvaraju se postavke iste. Brzina pokretne trake je zadana s 1 m/s te se unosi pod „*Speed*“. Širina je također podešena na 0,7 m pod prozorom „*Width*“. Kapacitet je ostavljen na zadanih -1 unutar prozora „*Capacity*“ zato što će se u daljnjem dijelu rada provoditi ispitivanja optimalnog broja namjenskih paleta pa nije u cilju ograničavati se s kapacitetom. Zadnja stavka koju treba podesiti je udaljenost između entiteta koja je također ostavljena na -1 jer se palete mogu dodirivati tijekom transporta. Sve postavke su prikazane na slici 4.34.



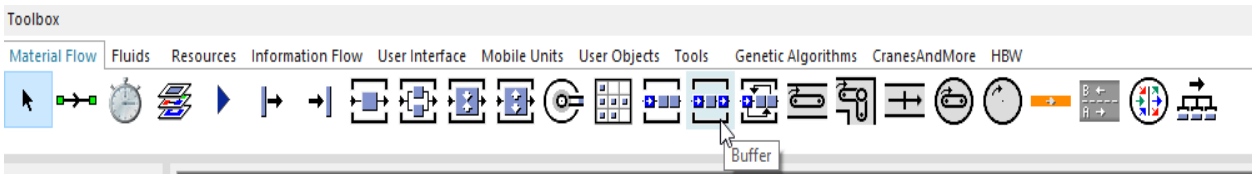
Slika 4.34 Postavke pokretne trake

Uz ovu veliku pokretnu traku potrebno je postaviti još šest malih, tri preko kojih će komadi putovati iz namjenskog stroja za montažu u namjenski stroj za test propusnosti i tri preko kojih će komadi izlaziti iz namjenskog stroja za test propusnosti na pregled i pakiranje. Pokretne trake koje vode iz montažnog uređaja u test propusnosti nazvane su redom: Mala1, Mala 2 i Mala 3, a pokretne trake koje vode iz uređaja za test propusnosti na pakiranje: Mala11, Mala12 i Mala13. Sve su one iste duljine koja iznosi 2,5 m i širine koja iznosi 0,5 m. sve ostale postavke pokretnih traka ostaju ne promijenjene izuzev brzine kretanja koja iznosi 0,5 *m/s* te izgledaju kako je prikazano na slici 4.35.



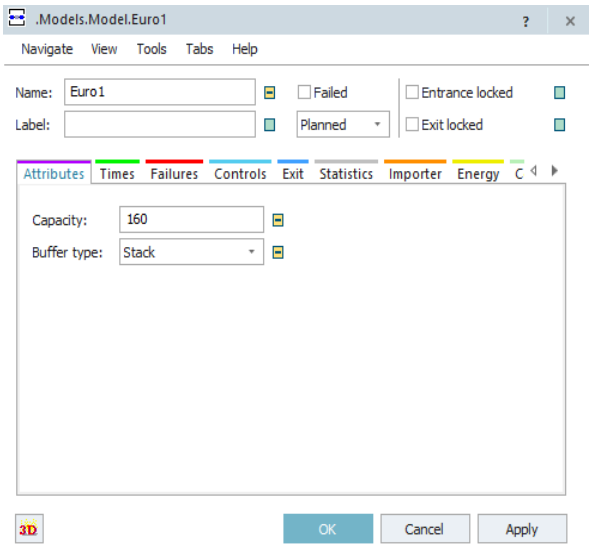
Slika 4.35 Postavke malih pokretnih traka

Tri objekta *Buffer* koji su ubačeni u radni prostor predstavljaju palete za završno pakiranje u koje se komadi, nakon što su prošli sve operacije, slažu i u kojima se šalju kupcu. Dakle, *Buffer* je objekt koji služi kao međuskladište Njih se također ubacuje iz alatne trake u radni prostor *drag and drop* metodom. Nakon što su ubačeni, razmještaju se prema prikazu na slici 4.36.



Slika 4.36 Položaj objetka Buffer u alatnoj traci

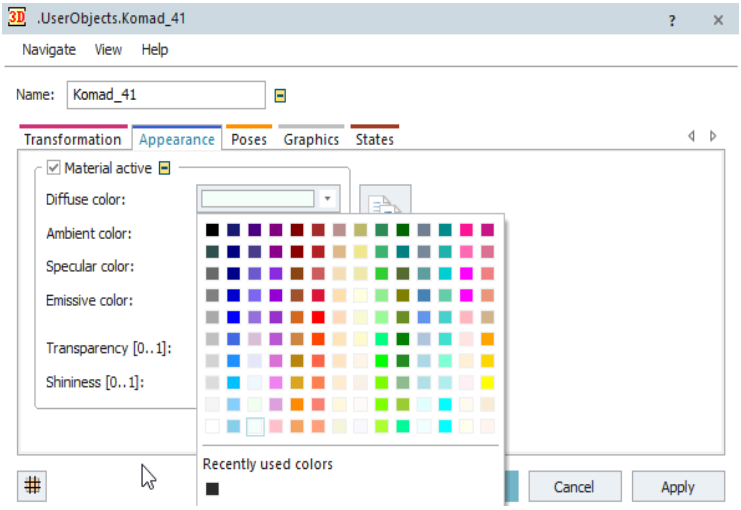
Potom se dvostrukim klikom otvaraju postavke. Pod karticom „*Attributes*“ potrebno je promijeniti „*Capacity*“ (Kapacitet) u 160 i pod „*Buffer type*“ u padajućem izborniku odabrati „*Stack*“. Kapacitet je podešen prema stvarnom kapacitetu konačne ambalaže. Postupak je potrebno ponoviti za sva tri objekta *Buffer*, a postavke trebaju izgledati kako je prikazano na slici 4.37.



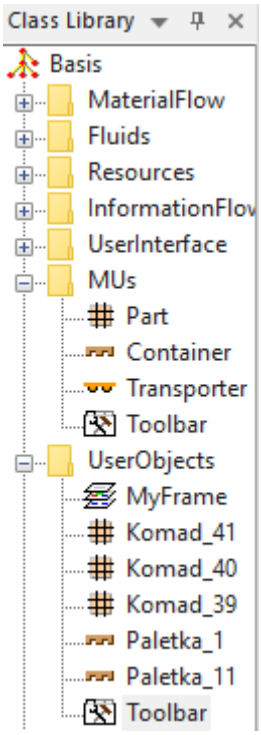
Slika 4.37 Postavke Buffer-a

U *Class Libraryu* su se duplicirala tri *User Objecta* od stavke *Part*. Taj *Part* se naziva MU, a predstavlja mobilnu jedinicu. Tri *Parta* su preimenovana u: „Komad_39“, „Komad_40“, „Komad_41“.

Kako bi se mogli vizualno razlikovati unutar simulacijskog modela promijenjene su im boje, prema gore navedenom redoslijedu, redom u: plavu, crvenu i bijelu. To se radi na način da se stisne desni klik na željeni entitet u „*Class Libraryu*“ i u padajućem izborniku pritisne na „*Edit 3D properties*“. Kada se to pritisne otvara se novi prozor u kojem je potrebno otvoriti karticu „*Appearance*“ i aktivirati kvačicom „*Material active*“. Tada se omogućuje promjena boje pod stavkom „*Diffuse color*“. Desno od nje pritiskom na strelicu otvara se padajući izbornik s cijelom paletom boja, gdje se odabire jedna po izboru i pritiskom na „*Apply*“ i „*OK*“ boja se mijenja (slika 4.38). Također, u „*Class Librar-u*“ su duplicirana dva „*User Objecta*“ od stavke „*Container*“ (slika 4.39). Preimenovani su u: Paletka_1 i Paletka_11. Radi lakšeg modeliranja duplicirani su dva puta a detaljniji opis nalazi se u poglavlju 6.



Slika 4.38 Izmjena boje entiteta



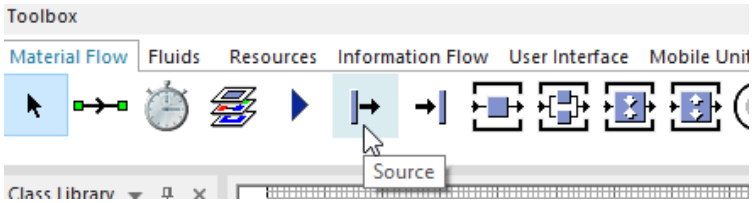
Slika 4.39 Dupliciranje Containera

Na paletku 1 stavljat će se komadi 39 i 40, a na paletku 11 komadi 41. U simulacijski model je potrebno ubaciti još četiri objekta *Source* (u daljnjem tekstu: izvor). Izvor je objekt koji služi, kao što mu samo ime kaže, kao objekt koji stvara entitete (slika 4.40). Prvi izvor nazvan je „Crveni“, drugi je nazvan „Paletke“, treći „Plavi“ i četvrti „Bijeli“. Postavke izvora Crveni, Plavi i Bijeli su praktički iste, razlikuju se samo u proizvodu koji će stvarati (slika 4.41). Količinu je potrebno namjestiti na -1, što definira da se proizvodi stvaraju neograničeno, a pod stavkom „*MU*

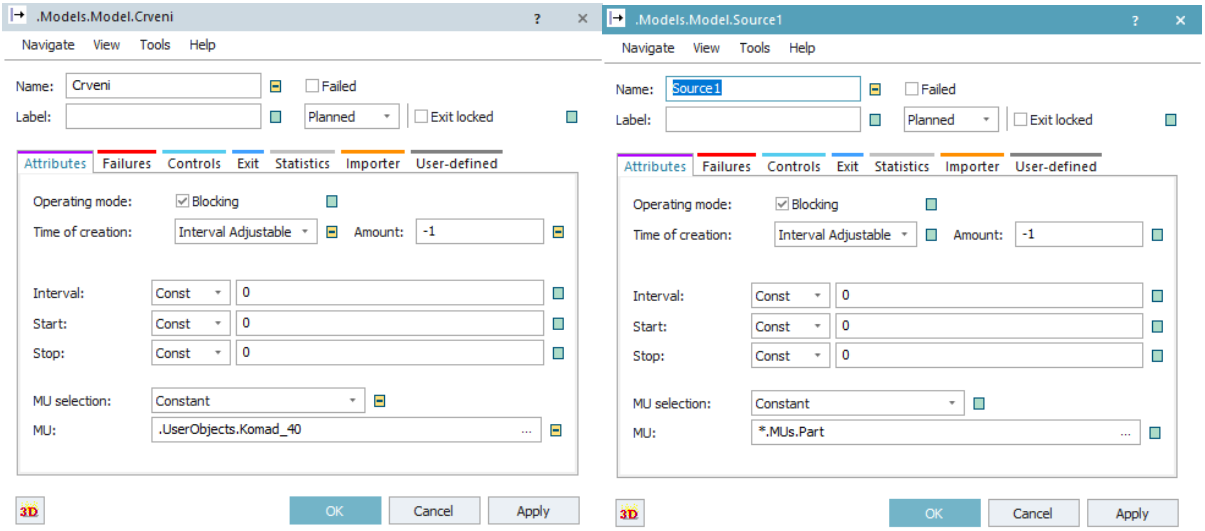
selection“ odabrati iz padajućeg izbornika *Constant* i pod stavkom MU unijeti *drag and drop* metodom sljedeće:

- Za Crveni izvor: Komad_40
- Za Plavi izvor: Komad_41
- Za Bijeli izvor: Komad_39

Prema tome postavke za objekte *Source* koji stvaraju proizvode trebaju izgledati isto izuzev vrste komada koje proizvode.

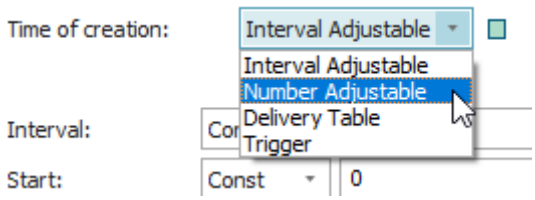


Slika 4.40 Položaj objekta izvor u alatnoj traci



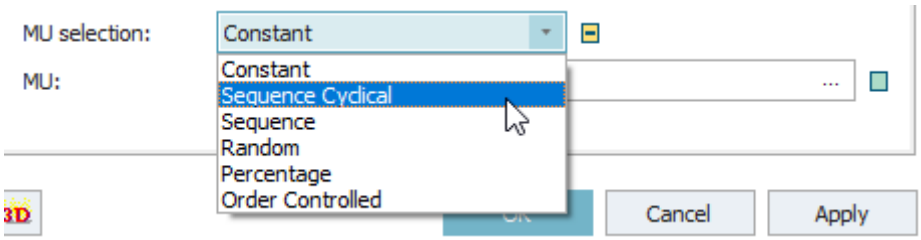
Slika 4.41 Postavke izvora – prije i poslije

Izvor za namjenske palete će se postaviti tako da će generirati proizvode iz tablice (Paletke_tablica). Prvi korak je otvoriti postavke izvora dvostrukim klikom ili desni klik pa iz padajućeg izbornika *Open*. Prvu stavku koju je potrebno promijeniti nalazi se pod *Time of creation*. Klikom na strelicu u prozoru pokraj otvara se padajući izbornik u kojem je potrebno odabrati *Number Adjustable* (slika 4.42).



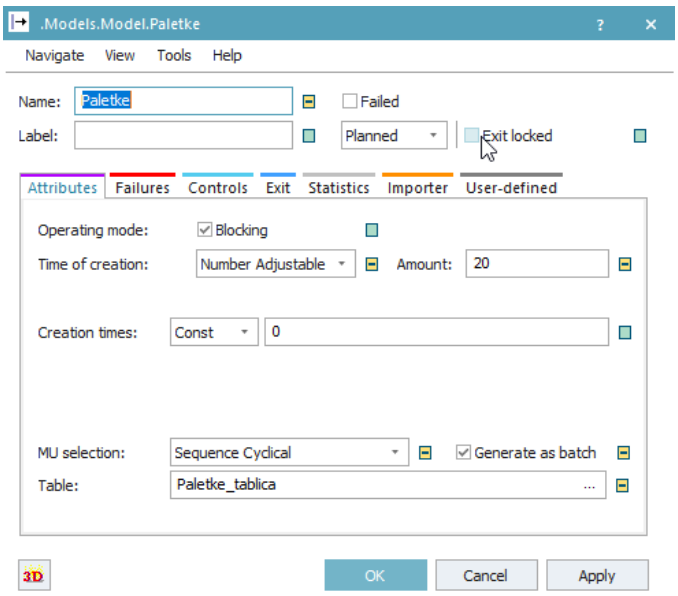
Slika 4.42 Number Adjustable

Ta stavka je odabrana zbog podrobnijeg ispitivanja optimalnog broja namjenskih paleta. Desno od ove stavke stoji stavka *Amount*. Nju nije potrebno odmah promijeniti već će se ona mijenjati tijekom ispitivanja u daljnjem dijelu rada. Da bi se moglo ubaciti tablicu pod stavku MU potrebno je promijeniti zadanu vrijednost stavke iznad, *MU selection*. Kada se klikne na strelicu u prozoru pokraj otvara se padajući izbornik (slika 4.43).

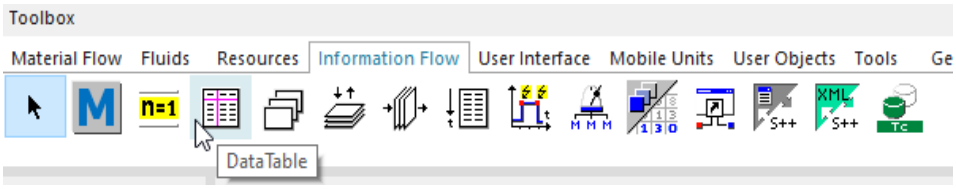


Slika 4.43 Opcija MU selection

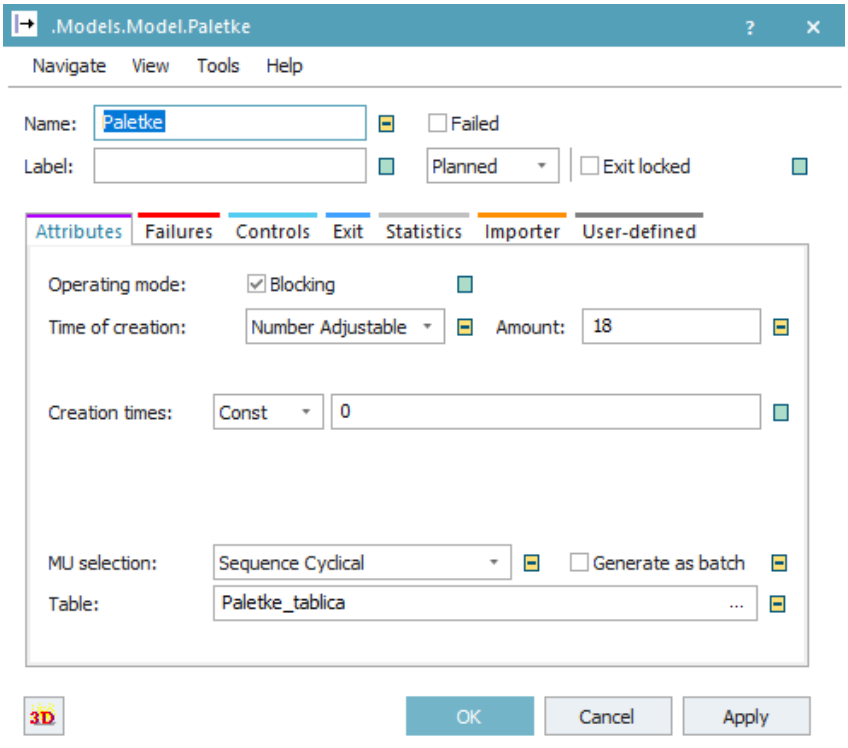
U padajućem izborniku odabire se *Sequence Cyclical* (slijedno ciklički) i pritisne tipka *Apply*. Nakon toga stavka postavke izvora mijenjaju izgled i stavka MU se mijenja u *Table* (tablicu). Izvor nam dopušta tu mogućnost korištenja tablice samo u slučaju odabira: *Sequence Cyclical, Sequence, Random i Percentage*. U ostala dva slučaja potrebno je ubaciti MU, objekt *drag and drop* metodom. Sljedeći korak je odabir tablice koja se želi koristiti. Da bi se to moglo ispuniti potrebno je unijeti tablicu iz izbornika *Information Flow*. Imenuje se tablica po želji, u ovom slučaju „Paletke_tablica“, i isto ime upisuje se u opcijama za izvor pod *Table* (slike 4.44 - 4.46). Osim tog načina unošenja moguće je tablicu unijeti u prozor *drag and drop* metodom. Kada se pritisne tipka *Apply* tablica će se automatski formatirati u potreban format za unos broja jedinica za svaku seriju proizvoda. Tablica bi trebala izgledati kao što je prikazano na slici 4.47.



Slika 4.44 Odabir mobilnih jedinica



Slika 4.45 Položaj objekta tablica u alatnoj traci

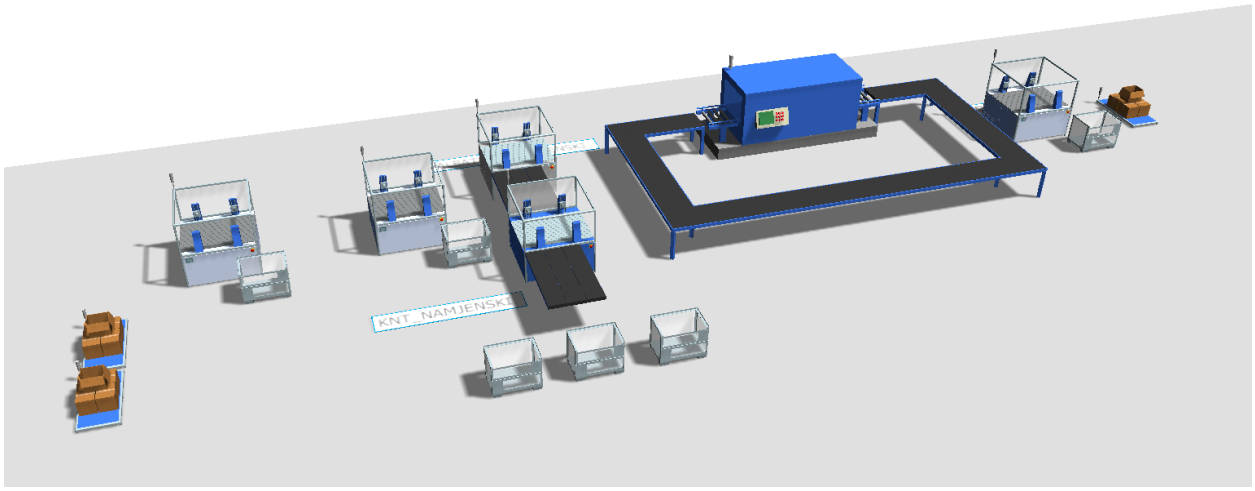


Slika 4.46 Postavke izvora za namjenske palete

	object 1	integer 2	string 3	table 4
string	MU	Number	Name	Attrib...
1	.UserObjects.Paletka_11	1		
2	.UserObjects.Paletka_1	2		
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				

Slika 4.47 Izgled Paletke_tablica

Sa svim postavljenim dobili smo trenutni vizualni prikaz dijela proizvodnog pogona koji izgleda kao na slici 4.46.



Slika 4.48 Vizualni prikaz izrađenog simulacijskog modela dijela proizvodnog pogona

Sljedeći korak je plan i izrada potpuno automatiziranog dijela proizvodnog pogona te ispitivanje optimalnog broja namjenskih paleta, što će biti razrađeno u poglavljima koji slijede.

5. IZRADA AUTOMATIZIRANOG DIJELA PROIZVODNOG POGONA

Potrebno je osmisliti automatizaciju trenutnog pogona, to uključuje automatizaciju manipulacije komadima na strojnoj obradi, automatizaciju transporta od strojne obrade do pokretne trake koja prolazi kroz protočnu perilicu za slučaj S1 i S2. Za S3 potrebno je osmisliti automatizaciju manipulacije komada na strojnoj obradi, automatizaciju manipulacije sa strojne obrade na gore navedenu pokretnu traku. Sljedeće je potrebno osmisliti manipulaciju komada s pokretne trake na namjenski uređaj za montažu.

Ideja je u proizvodni sustav ubaciti umjesto transporta komada kolicima, od strojne obrade do pokretne trake, pokretnu traku te umjesto ručne manipulacije komada s radnicima ubaciti manipulaciju robotskom rukom, a dostava komada do mjesta za obradu i odvoz izradaka ostaje isti odnosno vrši se s pomoću viljuškara.

Idejno rješenje za manipulaciju komadima na strojnoj obradi je korištenje robotske ruke. Kako je riječ o dva stroja koja treba posluživati predložena su tri koncepta. Prvi koncept je korištenje dvije robotske ruke, po jedna za svaki stroj, sljedeći je korištenje jedne pokretne robotske ruke te korištenje portalne robotske ruke. Za transport od strojne obrade do pokretne trake koristila bi se također pokretna traka. Za manipulaciju komadima s jedne pokretne trake na drugu koristila bi se robotska ruka. U slučaju trećeg obradnog centra S3, manipulaciju komadima, pri obradi te postavljanje komada na namjenske palete, također je ideja koristiti robotsku ruku. Kako bi kontrolirati namjenske palete na velikoj pokretnoj traci ideja je koristiti senzore za zaustavljanje i propuštanje istih. Potrebna će biti tri senzora na pokretnoj traci, od kojih će dva morati razlikovati namjenske palete. Prvi koji će morati razlikovati se nalazi na mjestu prijenosa komada s pokretne trake koja dovodi komade sa strojne obrade do velike pokretne trake, a drugi kod trećeg obradnog centra. Treći senzor se nalazi na izlazu iz protočne perilice.

Prilikom razrade koncepta idejnog modela krenulo se od dva obradna centra, odnosno od manipulacije komada na strojnoj obradi. Korištenje portalne robotske ruke je isključeno nakon kratkog razmatranja zato što je otežan pristup okretnom stolu (steznim napravama) zbog kuta ulaska. Sljedeća ideja korištenja pokretne robotske ruke je također odbačena jer se korištenjem iste stvara usko grlo.

Transport komada od strojne obrade s dva obradna centra do velike pokretne trake je osmišljen korištenjem manje pokretne trake. Ta traka će se nalaziti na određenoj visini kako bi omogućila prilaz obradnim strojevima, robotima i ostalim strojevima u slučaju kvara ili održavanja. Na kraju pokretne trake se nalazi senzor koji zaustavlja traku i javlja robotskoj ruci

koja će preuzimati komade s viseće pokretne trake te koja će ih prenositi na veliku pokretnu traku. Ta robotska ruka će imati dvostruku hvataljku (gripper) kako bi se povećao protok na toj pokretnoj traci. Manipulaciju komadima na trećem obradnom centru također će odrađivati robotska ruka. Radi sprječavanja stvaranja uskog grla na senzoru, koji se nalazi na velikoj pokretnoj traci kod trećeg obradnog centra, koji zaustavlja namjenske palete, robotska ruka će također imati dvostruku hvataljku. Još jedan razlog dvostruke hvataljke je brojno stanje komada na izlazu iz proizvodnog sustava po poziciji nakon sat vremena. U slučaju robotske ruke s jednom hvataljkom broj komada 840.41 je za trećinu manji u odnosu na slučaj s duplom hvataljkom. Na izlazu iz perilice na pokretnoj traci nalazi se treći senzor na kojem se zaustavljaju obje vrste namjenskih paleta. Na tom dijelu je potrebno osmisliti transport komada s pokretne trake u namjenski uređaj za montažu, što je objašnjeno u daljnjem dijelu teksta.

5.1. Problematika

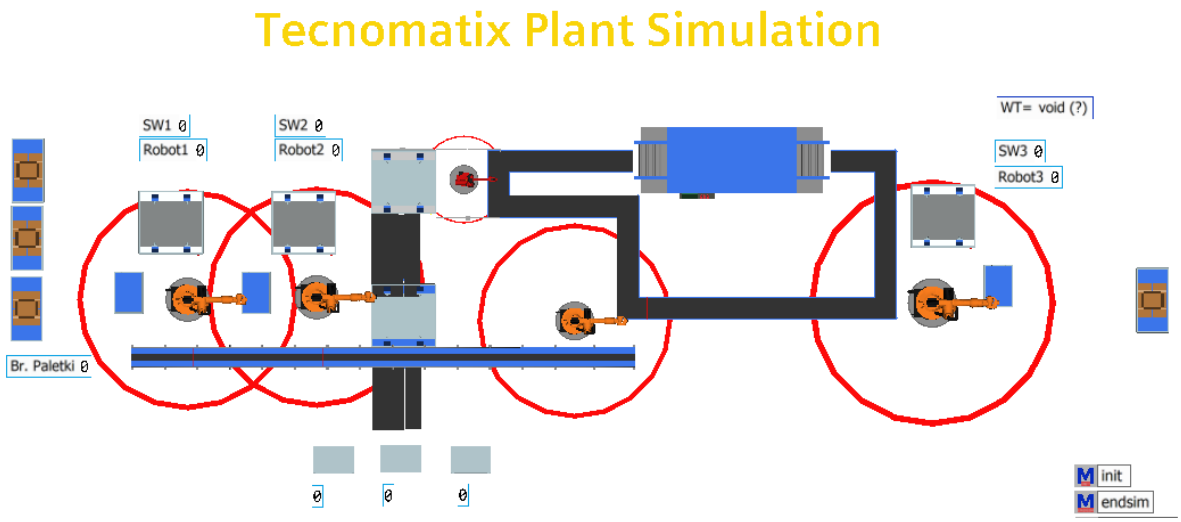
Kako namjenski uređaj za montažu ima samo tri stezna mjesta on predstavlja usko grlo gdje će se nakupljati komadi, što zapravo predstavlja glavni problem za koji je potrebno osmisliti rješenje. Preinake u vidu dodavanja dodatnih steznih mjesta u stroju nisu moguće. Osmišljena su dva rješenja ovog problema:

1. Prvo rješenje je produljenje pokretne trake na izlazu iz protočne perilice. Montažni namjenski uređaj bi se također produljio te bi pokretna traka ulazila u njega. Na pokretnoj traci bi se postavio senzor koji bi zaustavljao namjenske palete. Robotska ruka koja već postoji unutar montažnog uređaja bi, uz korekciju pozicije unutar uređaja, obavljala manipulaciju komadima.
2. Drugo rješenje koje se nameće je korištenje kolaborativnog robota - kobota koji bi bio postavljen na pomoćni stol. Konstrukcija bi se nalazila točno ispred perilice uz pokretnu traku, a između pokretne trake i montažnog uređaja. Sklop bi funkcionirao tako da se namjenska paleta, prilikom kretanja po pokretnoj traci, zaustavi u položaju gdje je najbliža kobotu. U tom trenutku kobot uzima komade te ih manipulacijom postavlja na pomoćni stol, svaku poziciju na svoje mjesto na stolu. Strana montažnog uređaja koja gleda prema protočnoj perilici bi bila napola otvorena kako bi robotska ruka koja se nalazi unutra mogla uzimati komade s pomoćnog stola.

U daljnjem dijelu teksta objašnjena je izrada simulacijskog modela za automatizirani proizvodni sustav. Izrađeni su simulacijski modeli za oba rješenja kao i detaljnije objašnjenje istih.

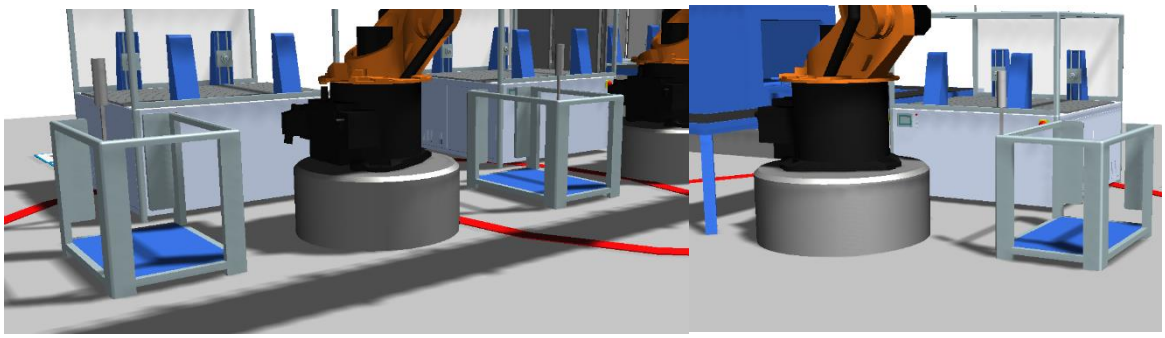
5.2. Izrada simulacijskog modela za automatizirani proizvodni sustav – verzija s produljenom pokretnom trakom

Verzija proizvodnog modela s produljenom trakom je jedan od promatrana dva slučaja (slika 5.1). Ova verzija je u odnosu na drugu verziju jeftiniji izbor te je jednostavnija sama izvedba.



Slika 5.1 Simulacijski model automatiziranog proizvodnog sustava – verzija s produljenom pokretnom trakom

Kako bi se prikazao automatizirani proizvodni sustav potrebno je nadograditi trenutni simulacijski model.Kreće se od postavljanja objekata *Buffer*. Oni će predstavljati palete u kojima će se nalaziti. Te palete će dovoziti radnici pomoću s viljuškara na za to predviđeno mjesto (slika 5.2). Oni se nalaze na alatnoj traci pod nazivom „*Buffer*“ koji se ubacuju jednostavnom *drag and drop* metodom. Bit će potrebna tri objekta *Buffer*, po jedan za svaki obradni centar.

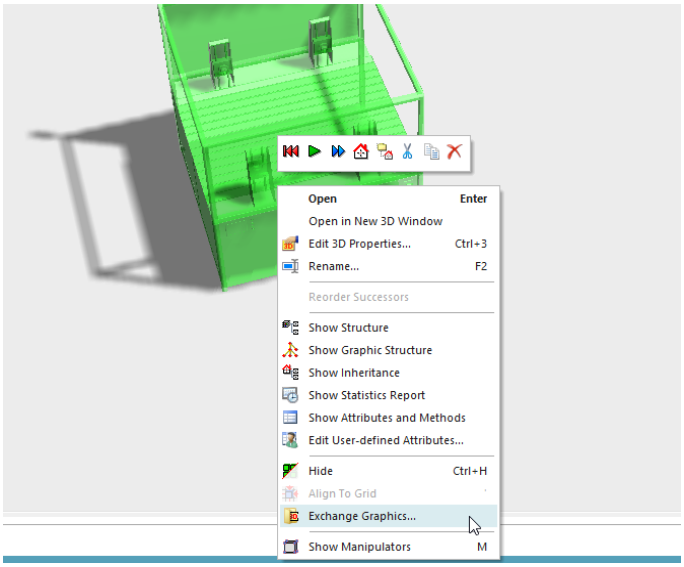


Slika 5.2 Raspored objekata B*uffer*

Nakon što ih se ubaci u simulacijski model, potrebno ih je postaviti na za to predviđena mjesta. Postavke *Buffera* nije potrebno mijenjati. Inače je potrebno promijeniti odnosno namjestiti željene postavke prema zadanim parametrima. U ovom slučaju nije potrebno zato što oni neće utjecati na testiranje te će nam oni poslužiti samo kao usputna stanica gdje će izvor - *Source* slati komade a robot preuzimati za daljnju obradu.

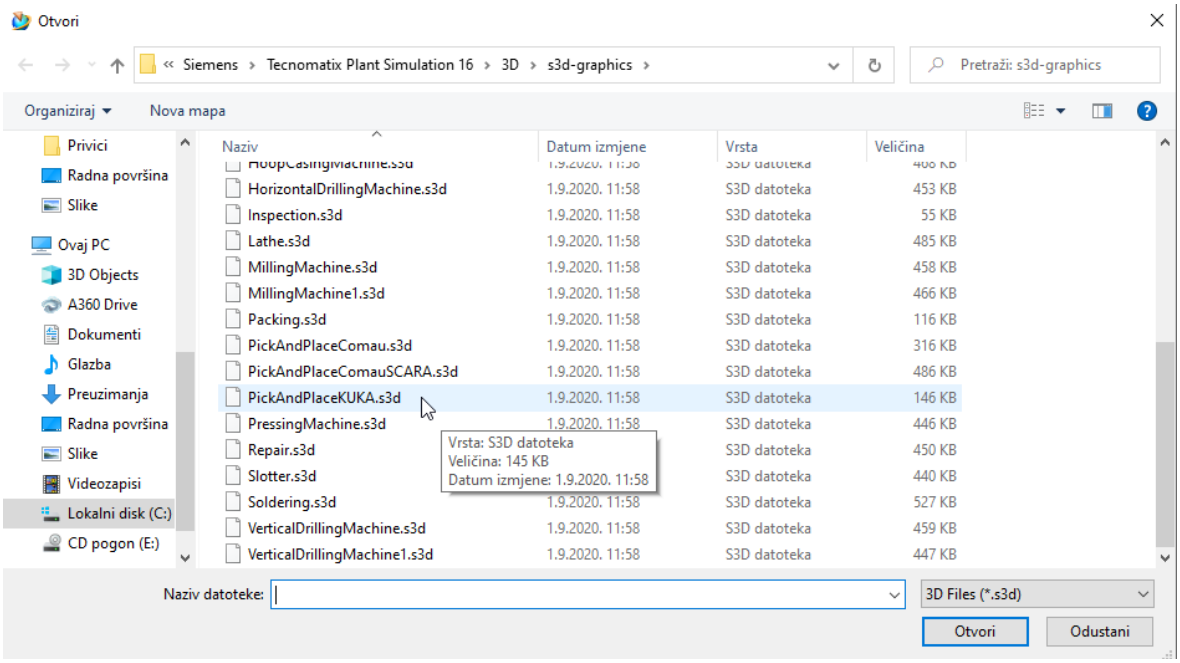
Nakon *Buffera* potrebno je ubaciti i robotske ruke. Softver Tecnomatix Plant Simulation nudi opciju robotske ruke koja se nalazi u alatnoj traci pod nazivom „*PickAndPlace*“. Nakon ubacivanja u radni prostor i povezivanja uvidjelo se da oni ne ispunjavaju zadane zadatke. Naime, potrebno je da robotska ruka uzima komade iz *Buffera*, postavlja ih u obradni centar za obradu te nakon obrade ih uzima i postavlja na pokretnu traku. Problem se javljao u trenutku kada robotska ruka uzima komade iz obradnog centra nakon obrade. Nakon običnog povezivanja pokušaj je bio prijeći na opciju, „*Target*“. Kod te opcije je potrebno unijeti željena mjesta na koja će se robotska ruka kretati te obavljati popratne radnje. Nažalost ta opcija nije dala očekivane rezultate. Robotska ruka ne može postavljati komade na namjensku paletu nakon što ih uzme nakon obrade. Nakon brojnih pokušaja nije se uspjelo postaviti da robotska ruka izvrši sve željene zadatke.

Poslije toga se, nakon istraživanja, krenulo s drugačijim pristupom. Ideja je da se obradni centar odnosno objekt u alatnoj traci „*ParalellStation*“ iskoristi kao robotska ruka. Uzet je objekt „*ParalellStation*“ zbog kasnije moguće potrebe korištenja dvostruke hvataljke jer za razliku od objekta „*Station*“ omogućuje obrade veće količine komada. Prvi korak ove ideje je promjena izgleda obradnog centra. Bitno je napomenuti da se ovom naredbom ne mijenja samo izgled već se mijenjaju i animacije samog objekta. To se postiže pritiskom desnog klika na obradni centar i lijevim klikom na stavku „*Exchange graphic*“ (slika 5.3).

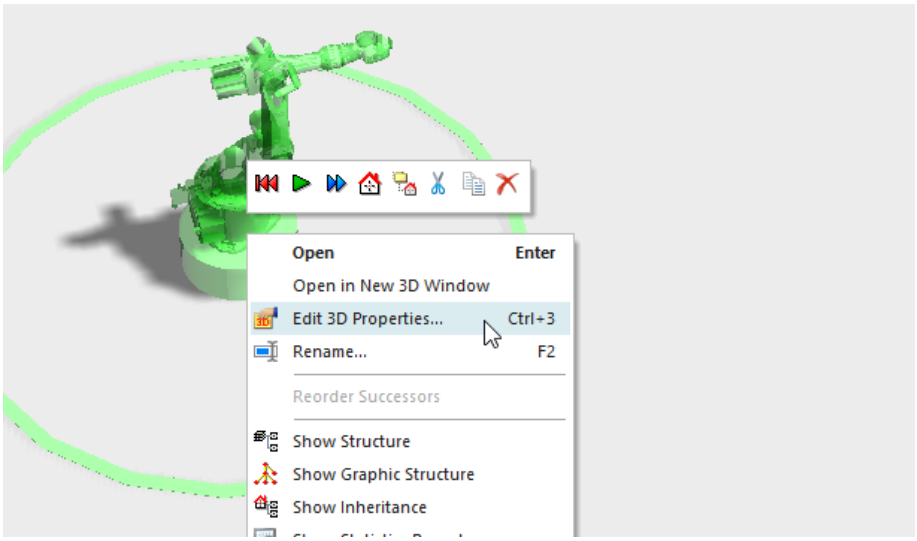


Slika 5.3 Promjena izgleda objekta ParalellStation

Otvara se prozor u kojem je potrebno otići u mjesto datoteke, odnosno u mjesto gdje je datoteka instalirana. Unutar mape softvera nalazi se mapa pod nazivom „*3D*“ u kojoj se nalaze dvije mape. Te dvije mape sadrže grafičke izglede raznih strojeva i uređaja, a za potrebe ovog rada će biti potrebne one koje u svom nazivu sadrže riječi „*Pick And Place*“. Može se odabrati bilo koja od ponuđenih jer se razlikuju samo u izgledu dok su animacije i funkcije iste. U ovom slučaju odabran je izgled „*PickAndPlaceKuka.s3d*“,(slika 5.4). Sljedeći korak je namjestiti 3D postavke robotskih ruku. One se otvaraju tako da se desnim klikom miša stisne na istu i u padajućem izborniku pritisne se na opciju „*Edit 3D Properties*“. Otvara se prozor kao na slici 5.5.

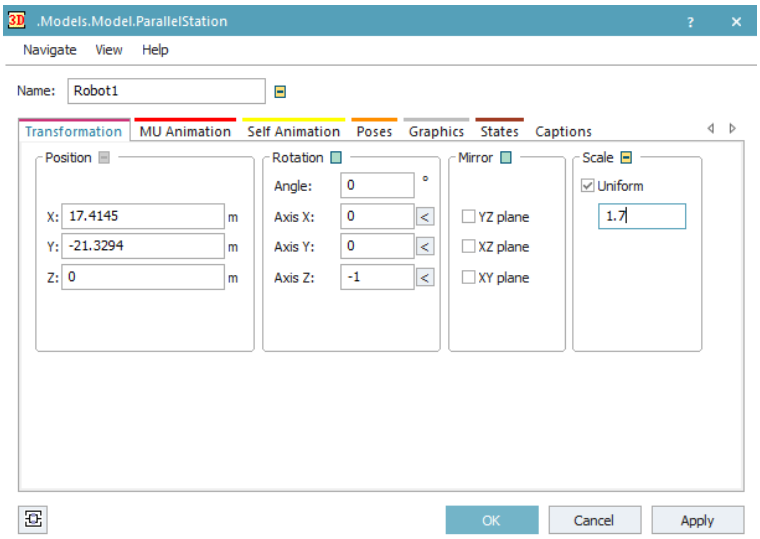


Slika 5.4 Lokacija na računalu za promjenu izgleda objekata



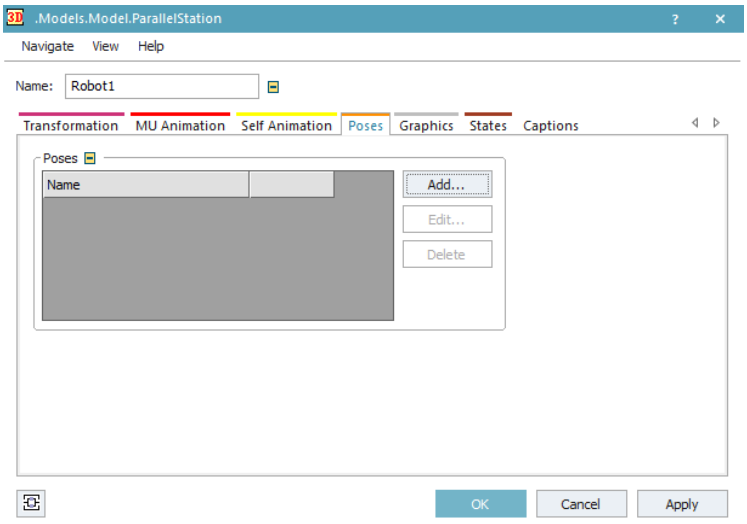
Slika 5.5 3D postavke novostvorene robotske ruke

Prvo što se mijenja je naziv stroja u „Robot1“, pod opcijom „*Name*“. Sljedeće se mijenja stavka „*Scale*“. S tom stavkom mijenja se veličina robota u odnosu na trenutnu. Potrebno je odrediti zadovoljavajuću visinu pošto će ti roboti postavljati komade na viseću pokretnu traku. U ovom slučaju traka će se nalaziti na visini od 2,5 m, no u simulaciji je namještena na visini od 1,3 m kako traka ne bi bila previsoka u odnosu na visinu strojeva u simulaciji. Zbog toga se metodom pokušaja i pogrešaka došli do vrijednosti 1,7 m, za prethodnu stavku (slika 5.6).



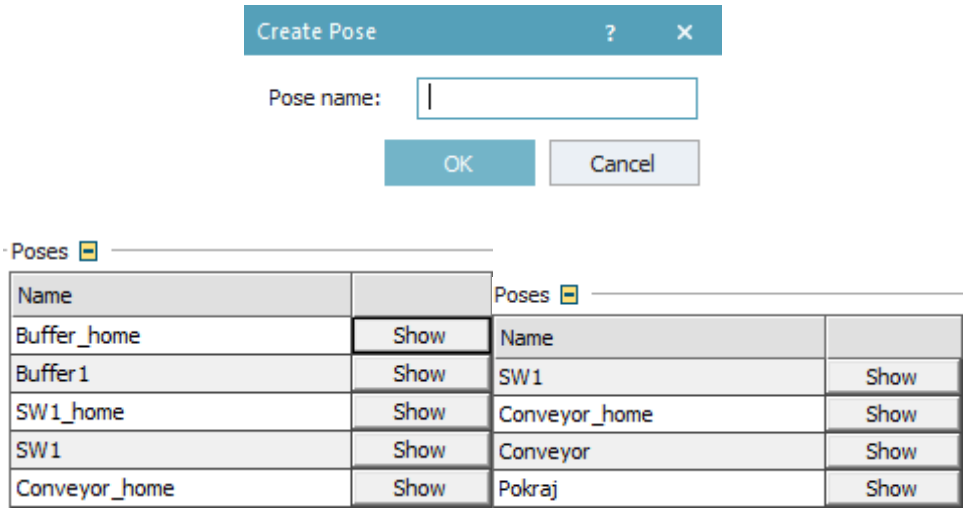
Slika 5.6 Naziv i skaliranje robotske ruke

Zatim se mijenjaju postavke pod opcijom „*Poses*“ (pozicije). Kao što i sam naziv govori postavljat će se robotsku ruku u željene pozicije. Kada se klikne na tab „*Poses*“ otvara se sljedeći prozor (slika 5.7).



Slika 5.7 Pozicije robotske ruke

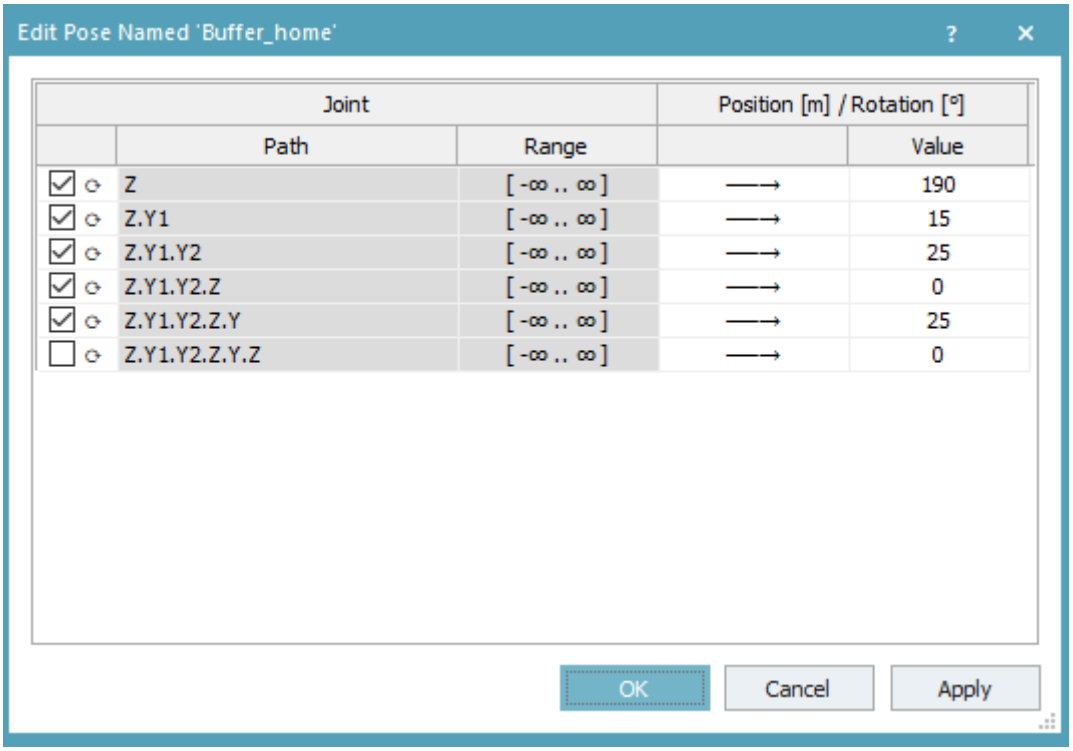
Da bi dodali poziciju potrebno je pritisnuti na stavku „*Add*“. Pritiskom na nju otvara se novi prozor u kojem je potrebno unijeti ime pozicije koju se postavlja. Prva pozicija koju će se postavljati je „*Buffer_home*“. Nazivi svih pozicija koje će se postavljati u daljnjem dijelu teksta su povezane s radnjama koje će robotska ruka izvršavati (slika 5.8)



Slika 5.8 Dodavanje i popis pozicija robotske ruke

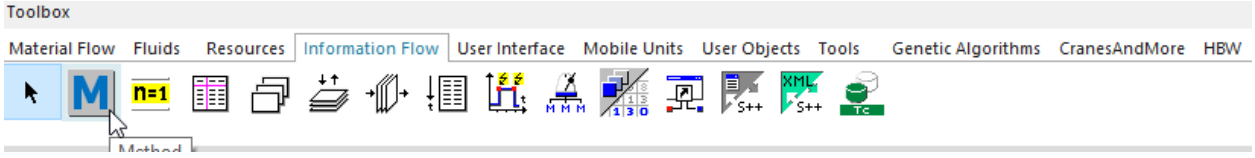
Nakon što smo odredili naziv pozicije i stisnuli „*OK*“ otvara nam se novi prozor u kojem su prikazani svi dijelovi robotske ruke i njihove moguće rotacije po osima. U poziciji „*Buffer_home*“ potrebno je namjestiti robotsku ruku u poziciju u kojoj će se ona nalaziti iznad *Buffera* spremna za uzimanje komada. Bitno je napomenuti da je pri odabiru dijela koji će se kretati potrebno s lijeve strane označiti s kvačicom, a s desne strane unijeti veličinu koja je potrebna. Ne postoji način na koji se može odmah znati koju veličinu treba odabrati stoga se koristila metoda pokušaja i pogreški. Drugim riječima, unese se veličina za koju se smatra da bi robotska ruka došla u željeni položaj i stisne se na tipku „*Apply*“ čime robotska ruka automatski prelazi u željenu poziciju. Ako robotska ruka nije u poziciji koju tražimo izbriše se unesena veličina i unese nova koja je veća ili manja od prethodno unesene, zavisno o potrebi. Za ovaj slučaj, za prvu poziciju, veličine su dobivene eksperimentalno i nalaze se kao na slici 5.9.

Svaka pozicija koja u svom nazivu ima riječ „*home*“ je pozicija u kojoj robotska ruka stoji ispred određenog stroja kod kojeg izvršava radnju. Kod obradnog centra će se ta pozicija koristi kao pozicija u kojoj robotska ruka čeka završetak strojne obrade dok su ostale pozicije s tim nazivom napravljene radi realnije i fluentnije animacije robotske ruke. Pozicija naziva „Pokraj“ je napravljena kako robotska ruka ne bi ulazila u koliziju s pokretnom trakom nakon ostavljanja komada na istu.



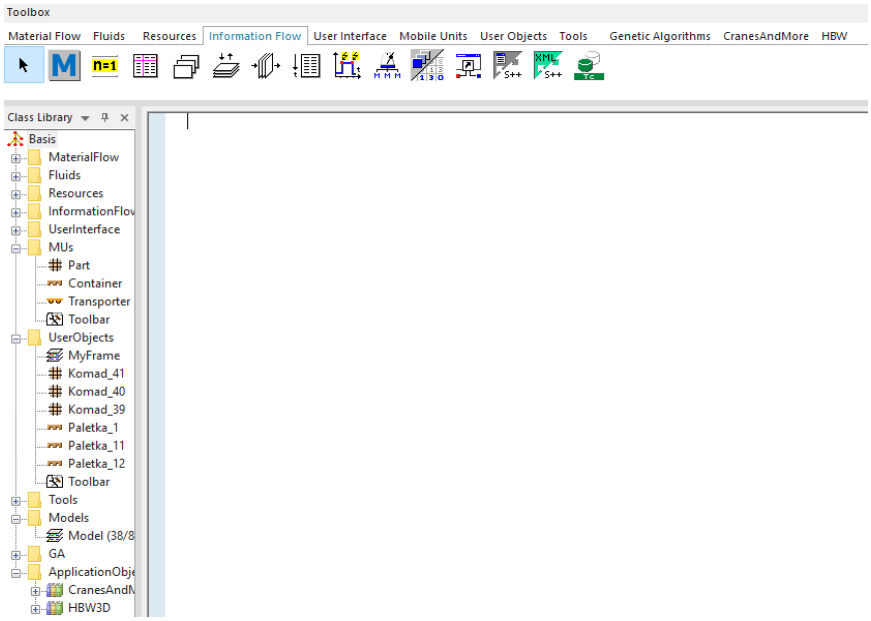
Slika 5.9 Koordinate poza robotske ruke

Kad su sve pozicije napravljene potrebno je napisati programski kod s pomoću kojeg će se robotska ruka kretati u simulaciji. Postoje dva načina korištenja programa za programiranje objekata. Prvi način je da se unese objekt *Method* (slika 5.10) u radni prostor i u njega se unese željeni programski kod. Drugi je da se otvori objekt, za koji se piše programski kod, dvostrukim lijevim klikom miša i pređe na karticu „*Controls*“. Pod tom karticom se nalazi više prozora: *Entrance, Exit, Pull i Shift Calendar*. Ovisno o tome što će objekt raditi kroz izvršavanje programskog koda, odabire se jedan od prozora. U desnom kraju prozora nalaze se tri točkice te pritiskom lijevog klika mišem na njih otvara se novi prozor koji nudi da se ili odabere objekt *Method* koji se nalazi u radnom prostoru (*Select object*) ili da se stvori jedan takav objekt unutar objekta za koji se piše programski kod. Za pisanje programskog koda u ovom slučaju korišten je prvi način te je potrebno u radni prostor ubaciti objekt „*Method*“ koji se nalazi u alatnoj traci pod karticom „*Information Flow*“.



Slika 5.10 Alatna traka – Method

Promijenjeno je ime objekta u „M_Robot1“. Zatim se otvara objekt lijevim dvostrukim klikom miša na njega. Kada ga se otvori, otvoreni novi prozor izgleda kao prazan papir, kako je prikazano na slici 5.11.



Slika 5.11Objekt Method - prazan prozor

Osnovna naredba koja će se koristiti prilikom programiranja kretanja robotske ruke je „*robot1._3D.poses.moveTo*“ ("Naziv pozicije", vrijeme izvođenja animacije), prema tome primjer kako treba izgledati jedna naredba slijedi: „*robot1._3D.poses.moveTo("Buffer1_home",2)*“. Prije no što se krene s ispisivanjem navedene naredbe potrebno je uvesti varijablu. Postoje dvije vrste varijabli, lokalne i globalne, a razlikuju se prema djelovanju i mjestu unosa:

- Globalna varijabla se unosi u sam radni prostor i sve metode (naredbe) mogu povući varijablu napisanu u radnom prostoru.
- Lokalna varijabla se unosi unutar metode, na početku, i nju može povlačiti samo ta naredba u kojoj se ona nalazi.

U ovom radu korištena je lokalna varijabla te su njen unos i uporaba opisani u nastavku teksta. Ona se unosi tako da se na početku napiše „*var*“ i u nastavku se napiše oznaka varijable te uz dodavanje dvotočke vrstu varijable koja će se koristiti. Vrste varijabli su sljedeće: *integer, real, length, speed, acceleration, weight, time, date, datetime, boolean, string, object, table, list, stack, queue* i *any*. Ovisno o veličini koja se koristi i onom što ona predstavlja koristit će se jedna od gore navedenih vrsta varijabli. Vrsta „*integer*“ predstavlja cijele brojeve, a u ovom slučaju to će se odnositi na količinu i imenovana je kao „*i*“. Na slici ispod (slika 5.12) je prikazana cijela naredba/programski kod, a opis se nalazi ispod iste.

```
var i:integer

waituntil Buffer1.occupied
wait 2
robot1._3D.poses.moveTo("Buffer1",2)
wait 3
for i:= 4 downto 1
    wait 3
    Buffer1.cont.move(robot1)
    wait 3
next
wait 2
robot1._3D.poses.moveTo("Buffer_home",2)
wait 3
robot1._3D.poses.moveTo("SW1_home",2)
wait 4
waituntil SW1.occupied=false or SW1.resworking=false
robot1._3D.poses.moveTo("SW1",2)
wait 3
for i:=4 downto 1
wait 2
robot1.cont.move(SW1)
wait 2
next
wait 2
wait robot1._3D.poses.moveTo("SW1_home",2)
SW1.startProcessing
waituntil SW1.resworking=false
wait 2
wait Robot1. 3D.poses.moveTo("SW1",2)

for i:=4 downto 1
wait 2
SW1.cont.move(Robot1)
wait 2
next
wait 2
wait Robot1._3D.poses.moveTo("SW1_home",2)
wait 2
wait Robot1._3D.poses.moveTo("Conveyor_home",2)
wait 3
wait Robot1._3D.poses.moveTo("Conveyor",2)
for i:=4 downto 1
wait 2
Robot1.cont.move(Conveyor4,1.9)
wait 2
next
waituntil Robot1.empty
wait 2
wait Robot1._3D.poses.moveTo("Conveyor_home",2)
wait 4
wait Robot1._3D.poses.moveTo("Pokraj",2)

self.execute
```

Slika 5.12 Programski kod prve robotske ruke - M_Robot1

Nakon što je unesena lokalna varijabla kreće se u ispisivanje programskog koda. Logika same metode je da se redom, od početne do krajnje pozicije, napišu naredbe kretanja robotske ruke te popratne naredbe koje sama robotska ruka mora izvršavati između svakog pokreta koji radi. Prva naredba je „*waituntil*“ kojom se robotskoj ruci govori da čeka u određenoj poziciji dok se ne izvrši navedena radnja. Puna naredba glasi: „*waituntil Buffer1.occupied*“, a čiji opis je: „Čekaj dok Buffer1 ne bude okupiran, odnosno pun.“ Zatim se uvodi naredba „*wait*“, a ona ima značenje kao i doslovan prijevod te riječi s engleskog na hrvatski a to je „Čekaj“. Sastoji se od naredbe i desno u nastavku vrijeme koje robotska ruka treba provesti čekajući, a zadaje se u sekundama. Zatim naredba za zauzimanje željene pozicije robotske ruke koja je navedena ranije u tekstu. Sljedeća je naredba za čekanje u ovom slučaju je radnja povećana s 2 sekunde na 3. Trenutni položaj robotske ruke je iznad *Buffera* i spremna je na preuzimanje komada. Kako nije riječ o klasičnoj robotskoj ruci kojoj se povezivanjem može narediti preuzimanje komada već o preoblikovanom obradnom centru potrebno je za to napisati naredbu u kojoj će se komadi slati na robotsku ruku i obratno s robotske ruke na predviđena mjesta. Naredba koja je korištena je „*for next*“, a u nju su dodane još tri naredbe: „*wait“, „cont.move“* te *„downto“* naredba. Također, u ovu naredbu prvi put unosimo lokalnu varijablu „i“. *Downto* naredba govori objektu, u ovom slučaju *Bufferu,* da broji unazad. *Cont.move* naredba govori da se sadržaj objekta pošalje na referentni objekt. Opis pune naredbe glasi: za varijablu „i“ koja iznosi 4 da broji unazad do 1, odnosno da uzme 4 komada, da se napravi pauza od 3 sekunde, da *Buffer* pošalje ta 4 komada na robotsku ruku, da opet sačeka 3 sekunde i naredba *next* označava da se izvršavaju sljedeće naredbe. Unutar ove naredbe robotska ruka primi 4 komada od strane *Buffera*. U daljnjem dijelu teksta naredba *wait* se više neće opisivati jer označava isto kroz cijeli program. Sa sljedeće dvije naredbe govori se robotu da zauzme pozicije, redom, iznad *Buffera* i ispred obradnog centra. Zatim opet naredba koja govori da čeka dok (*waituntil*) obradni centar nije okupiran, odnosno pun, ili ako ne radi (*resworking*) da uđe unutar obradnog centra. Nakon toga se ponavlja naredba *for-next* za prebacivanje komada s robotske ruke u obradni centar na isti princip kao i u prethodnom slučaju. Sljedeće, programom se naređuje robotskoj ruci da izađe iz obradnog centra i zauzme poziciju „*SW1_home*“, ispred obradnog centra. Slijedi naredba za pokretanja rada obradnog centra. Bitno je za napomenuti da je se naredba nalazi unutar ovog programa te isto tako da se nalazi nakon što je robotska ruka zauzela položaj izvan obradnog centra. Zatim naredba *waituntil* kojom se robotskoj ruci govori da čeka kraj obrade obradnog centra kako bi mogao neometano uzeti komade. Potom slijedi naredba zauzimanja pozicije unutar obradnog centra kao priprema za preuzimanje komada. Zatim slijedi opet *for-next* naredba kojom se govori obradnom centru prijenos komada na robotsku ruku. Sljedeće tri naredbe u programskom kodu služe za zauzimanje pozicija. Prva govori robotskoj ruci da izađe iz obradnog centra i zauzme poziciju ispred istog, druga da zauzme poziciju ispred pokretne trake i treća da se

zauzme pozicija za prijenos komada s robotske ruke na pokretnu traku. Postoji mogućnost da se prethodni korak odradi i sa samo dvije naredbe zauzimanja pozicija no tako se vrijeme i izgled animacije, odnosno kretanje robota, odstupaju od stvarne pa se zbog toga nije pristupilo tom načinu. Zatim naredba *for-next* kojom se govori robotskoj ruci da prebaci komade na pokretnu traku. U ovom slučaju naredba *cont.move* se sastoji od objekta s točnom pozicijom na koju je potrebno odložiti komade. Ovo je moguće samo kod objekata koji su orijentirani prema dužini. Točan položaj se odredio eksperimentalno metodom pokušaja i pogrešaka tako da su se unosile vrijednosti dok se nije dobila vrijednost 1.9 m, od početka pokretne trake, u kojoj se robotska ruka nalazi okomito na pokretnu traku. Slijedi naredba *waituntil* kojom se govori robotskoj ruci da sačeka s daljnjim izvršenjem programskog koda dok ne bude prazan, potom dvije naredbe zauzimanja pozicija iznad pokretne trake i dodatna pozicija „pokraj". Nakon pokretanja simulacije uvidjelo se da robotska ruka nakon otpuštanja komada na pokretnu traku kreće prema *Bufferu* prilikom čega prolazi kroz samu pokretnu traku krateći sebi put. Kako bi se to izbjeglo dodana je još jedna dodatna pozicija, pokraj, kako bi robotska ruka izbjegla koliziju s pokretnom trakom. I posljednja naredba u ovom programskom kodu je *self.execute* kojom se osigurava da će se programski kod, nakon što se izvrši do kraja, iznova pokrenuti.

Druga robotska ruka se također izrađuje od objekta obradnog centra prema navedenom postupku, a postavke su iste. Programski kod za drugu robotsku ruku, Robot2, je isti kao i u prethodnom slučaju, jedina razlika je u poziciji otpuštanja komada na pokretnu traku koja u ovom slučaju iznosi 5.9 *m*.

Treća robotska ruka se izrađuje kao i u prethodna dva slučaja, no postavke se razlikuju samo u veličini, odnosno u stavki „*Scale*". U ovom slučaju ona je podešena na 1,9. Programski kod za treću robotsku ruku je u prvom pokušaju bio isti kao i za prethodne dvije robotske ruke. Pokretanjem simulacije te provođenjem ispitivanja utvrđeno je da broj komada, na izlazu, sve tri pozicije se razlikuje. Količina komada prve dvije pozicije, s prva dva obradna centra, je puno veći u odnosu na treću poziciju čime rezultati simulacije ne zadovoljavaju. Problem je stvaralo usko grlo na trećoj robotskoj ruci zbog slabijeg protoka komada. Rješenje problema pronašlo se u dvostrukoj hvataljki treće robotske ruke. S njom robotska ruka uvijek ima spremna nova četiri komada koja će zamijeniti s četiri obrađena čime smo osigurali bolji protok i rješenje uskog grla. Stoga programski kod za treću robotsku ruku izgleda kako je prikazano na slici 5.13.

Na početku programskog koda potrebno je unijeti lokalnu varijablu kao i u prethodnim slučajevima. Zatim se uvodi uvjet, „*root.Robot3.isRunning := true*". Tim uvjetom uvodi se sigurnost pokretanja programskog koda, s obzirom na to da je riječ o kompleksnijem programskom kodu. Slijedi naredba zauzimanja pozicije ispred *Buffera.* Zatim slijedi *if-else-end* petlja, to je uvjetna petlja koja govori što treba izvršiti u slučaju da je uvjet ispunjen i što u slučaju da nije. U

tu petlju uvrštene su 3 *for-next* naredbe. Prvi uvjet u petlji je da treći obradni centar mora biti prazan, tek kada je ispunjen uvjet da *Buffer* prenosi 8 komada na robotsku ruku *for-next* petljom.

```
var i:integer

root.Robot3.isRunning := true

waituntil Buffer3.occupied=true
wait 2
robot3._3D.poses.moveTo("Buffer",2)
wait 2
if SW3.empty
    for i:= 1 to 8
        wait 2
        Buffer3.cont.move(robot3)
        wait 3
    next
        //Prilaz stroju
        wait 2
        robot3._3D.poses.moveTo("SW_home",2)
        wait 2
        waituntil SW3.resworking=false or SW3.empty
        robot3._3D.poses.moveTo("SW",2)
        wait 2
        //Puni SW3
        for i:= 1 to 4
            wait 2
            robot3.cont.move(SW3)
            wait 2
        next
        //Izlazi iz radnog područja stroja
        wait robot3._3D.poses.moveTo("SW_home",2)

else
    for i:= 1 to 4
        wait 2
        Buffer3.cont.move(robot3)
        wait 3
    next
    wait 2
    robot3._3D.poses.moveTo("SW_home",2)
    wait 2



end

waituntil SW3.resworking=false or SW3.empty
robot3._3D.poses.moveTo("SW",2)
wait 2
```

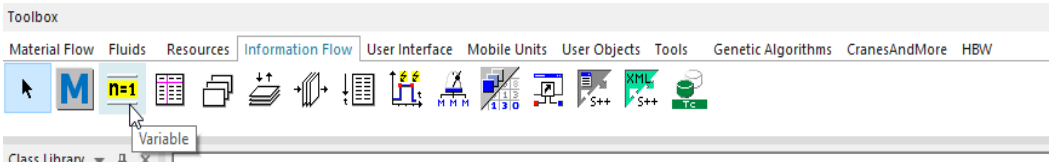
```
//Vađenje prva 4 komada
if SW3.occupied
    for i:= 1 to 4
        wait 2
        SW3.cont.move(Robot3)
        wait 2
    next
end
//Izlaz iz radnog područja stroja
wait robot3._3D.poses.moveTo("SW_home",2)
//Promjena pozicije
wait 5
//Ulaz u radno područje storja
wait Robot3._3D.poses.moveTo("SW",2)
//Punjenje stroja
for i:= 1 to 4
    wait 2
    robot3.cont.move(SW3)
    wait 2
next
//Izlaz iz radnog područja stroja
wait robot3._3D.poses.moveTo("SW_home",2)

SW3.startProcessing

//Pokret prema conveyor-u
wait Robot3._3D.poses.moveTo("Conveyor_home",2)
//Robot čeka na paletku
repeat
waituntil WT /= void
wait Robot3._3D.poses.moveTo("Conveyor",2)
for i:=1 to 1
    wait 2
    Robot3.cont.move(WT)
next


//Puštanje paletke
waituntil WT.NumMu=2
WT.stopped:=false
WT:=void
until Robot3.empty

//Odmak od conveyor-a
wait Robot3._3D.poses.moveTo("Conveyor_home",2)
//Početak novog ciklusa

self.executeNewCallChain
```

Slika 5.13 Programski kod treće robotske ruke - M_Robot3

Slijedi naredba zauzimanja pozicije ispred obradnog centra. Zatim *waituntil* naredba koja govori da robotska ruka čeka završetak rada obradnog centra ili da čeka da obradni centar bude prazan kako bi kroz *for-next* petlju prenio 4 komada u obradni centar. Nakon toga naredbom za

zauzimanje pozicije robotska ruka izlazi iz obradnog centra. Drugi dio *if-else-end* petlje govori da u slučaju da obradni centar nije prazan da ode do *Buffera* te da *buffer for-next* petljom prebaci četiri komada na robotsku ruku. S ovom petljom osiguran je rad dvostruke hvataljke robotske ruke. Zatim *waituntil* naredbom uvjetuje se robotsku ruku da ako je obradni centar prazan ili ne radi da mu priđe. Slijedi još jedna *if-end* petlja u kojoj se nalazi *for-next* naredba. Uvjet *if-end* petlje je da je obradni centar zauzet odnosno da se u njemu nalaze obrađeni komadi, a *for-next* naredbom se govori obradnom centru da prebaci komade na robotsku ruku. Potom slijede dvije naredbe zauzimanja pozicije. Kroz ove dvije pozicije prikazuje se izlaz robotske ruke iz obradnog centra, okret hvataljke te ponovni ulaz robotske ruke u obradni centar. Sljedeća je *for-next* naredba kojom robotska ruka odlaže nova četiri komada za obradu, potom naredba za zauzimanje pozicije izvan obradnog centra te naredba za pokretanje rada obradnog centra nakon što robotska ruka izađe izvan radnog prostora istog.

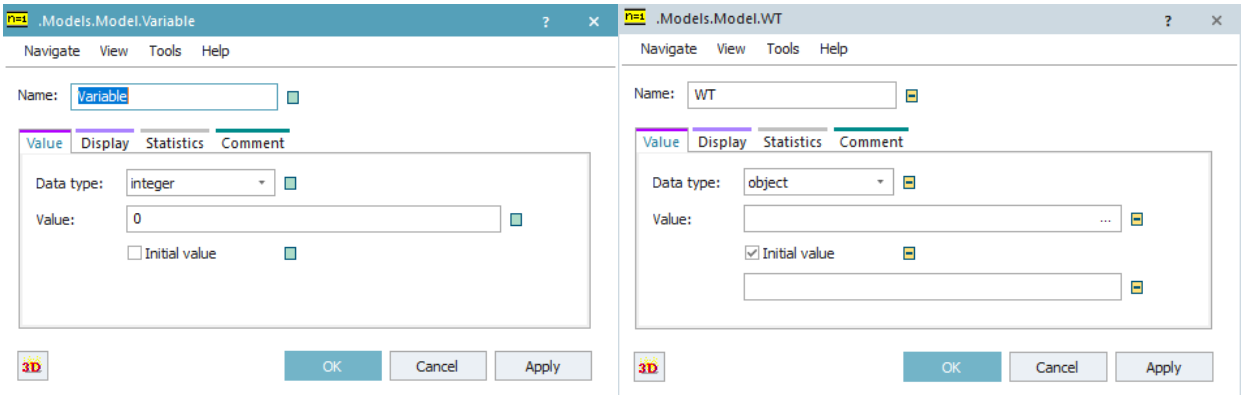
Slijedi dio programskog koda u kojem robotska ruka prenosi komade na namjensku paletu koja se kreće na velikoj pokretnoj traci. Kako bi to bilo moguće potrebno je uvesti globalnu varijablu „WT“ čija vrijednost će biti „*void*“ (slika 5.14). Simulacija dodjeljuje vrijednost *void* kada varijabla vrijednosti objekt ukazuje na objekt koji u tom trenutku ne postoji.



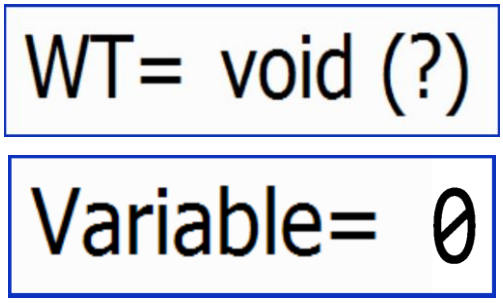
Slika 5.14 Objekt - Globalna varijabla

Globalna varijabla se dodaje tako da se iz alatne trake (slika 5.14) ubaci u radni prostor *drag and drop* metodom. Sljedeći korak je namještanje postavki istog prema prikazanom na slici 5.15. Potrebno je promijeniti ime, vrstu predmeta, vrijednost je potrebno ostaviti prazno i na kraju upaliti stavku „*Initial value*“ (početna vrijednost) te i nju ostaviti praznu.

Izgled globalne varijable u radnom prostoru bi se trebao promijeniti kao što je prikazano na slici 5.16.



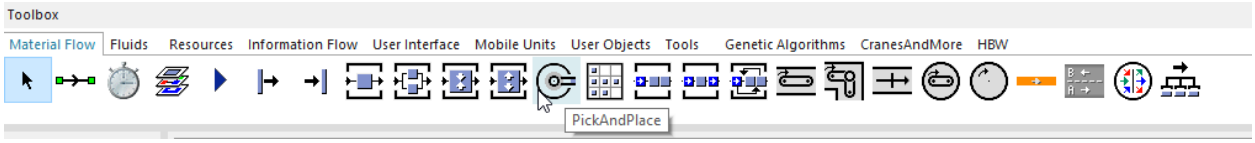
Slika 5.15 Postavke globalne varijable



Slika 5.16 Izgled globalne varijable – prije i poslije

U nastavku programskog koda nalazi se nova petlja *repeat-until* s pomoću koje se govori robotskoj ruci da izvršava radnju iznova dok se nekakav uvjet ne izvrši. Unutar ove petlje prva naredba je *waituntil* kojom se govori robotskoj ruci da čeka dok globalna varijabla bude različita od vrijednosti *void*. To će se ostvariti kada se namjenska paleta zaustavi na senzoru na pokretnoj traci pokraj robotske ruke. Kada se to ostvari, robotska ruka s pomoću naredbe za zauzimanje pozicije dolazi iznad namjenske palete u položaj spreman za prijenos komada na iste. Zatim *for-next* naredbom se govori robotskoj ruci da prenese dva komada na namjensku paletu. Potom slijedi *waituntil* naredba kojom se govori namjenskoj paleti da čeka dok se na nju ne prenesu dva komada, nakon čega je ona puna i može se nastaviti kretati po pokretnoj traci. Nastavak kretanja namjenske palete izvršava se naredbom „*WT.stopped:=false*“ te s naredbom „*WT:=void*“ kojima se namjenskoj paleti dodjeljuje vrijednost *void*. Petlja *repeat-until* završava s uvjetom „*until Robot3.empty*“ kojom se govori robotskoj ruci da izvršava prethodne radnje sve dok se ne isprazni. Programski kod završava s dvije naredbe. Prva je zauzimanje pozicije iznad pokretne trake a druga je naredba za ponovno izvršenje cijelog programskog koda.

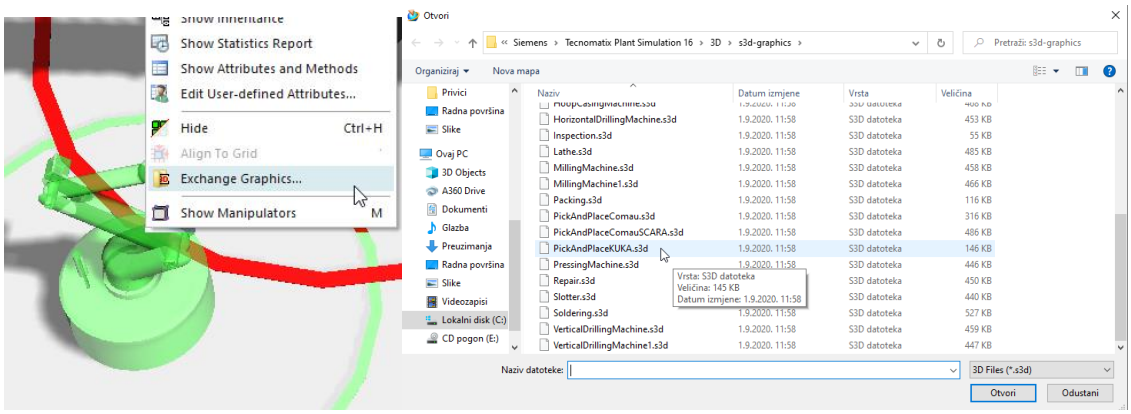
U simulacijski model je potrebno dodati još dvije robotske ruke. Njih će se ubaciti kao robotske ruke koje nudi softver, kao objekte *PickAndPlace*. Kako je već ranije napomenuto, nalazi se u alatnoj traci pod karticom „*Material Flow*“ - „*PickAndPlace*“ (robotska ruka, slika 5.17).



Slika 5.17 Objekt – PickAndPlace / Robotska ruka

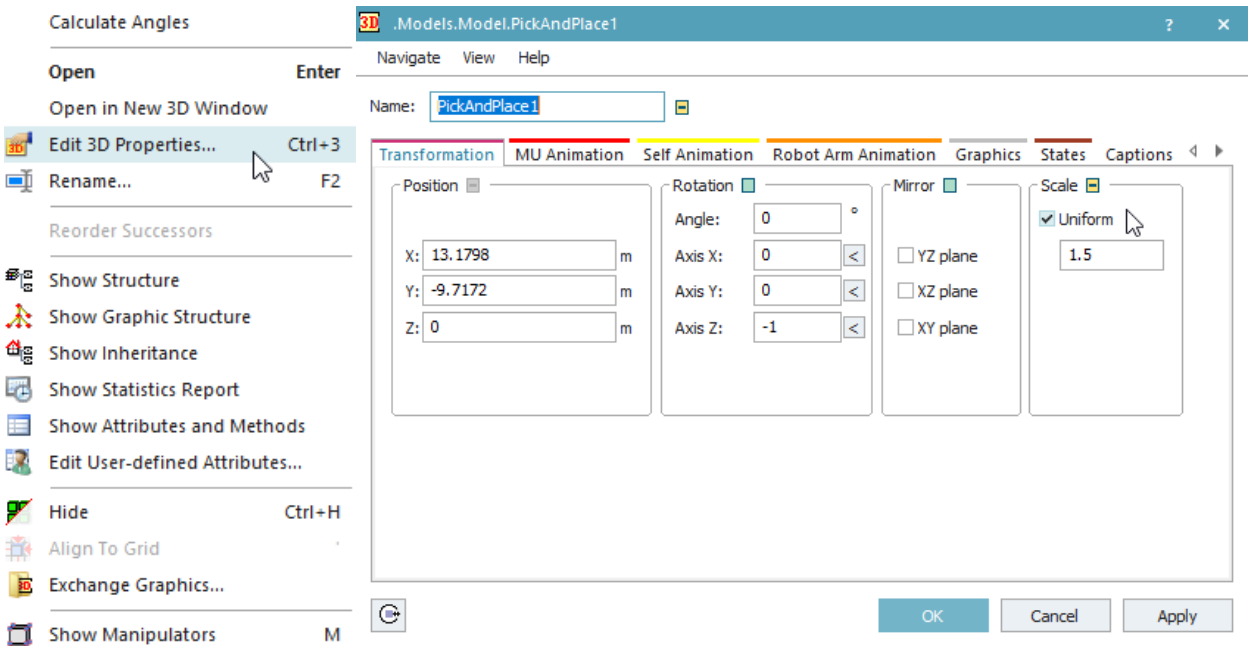
Jedna robotska ruka nalazit će se na drugom kraju viseće pokretne trake, a druga će se nalaziti unutar montažnog uređaja. Bitno je za napomenuti da se ta druga robotska ruka inače nalazi unutar spomenutog uređaja no u osnovnoj verziji simulacijskog modela nije ubačen već je vrijeme njegove manipulacije uračunato u vrijeme operacije montaže.

Prvu robotsku ruku postavlja se prema rasporedu prikazanom na početku potpoglavlja 5.2. Mijenja mu se izgled kao i prethodnim robotskim rukama. Također se odabire robot-kuka izgled robota (slika 5.18).



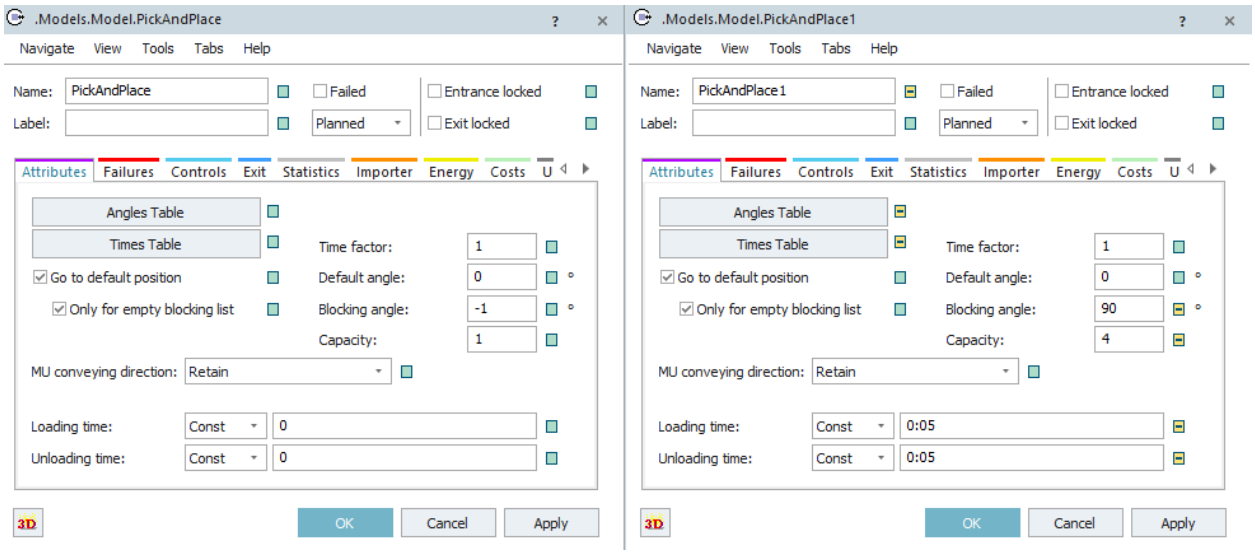
Slika 5.18 Izmjena izgleda robotske ruke_Robot4 - Robot Kuka

Što se tiče izgleda, mijenjaju se još i dimenzije kako bi se omogućilo robotskoj ruci da neometano preuzima komade s viseće pokretne trake. Ta postavka se mijenja u 3D postavkama uređaja. Potrebno je desnim klikom miša pritisnuti na robotsku ruku i u padajućem izborniku odabrati „*Edit 3D Properties*“. Mijenja se samo stavka „*Scale*“ i postavljaju se na vrijednost 1,5. Važno je označiti opciju „*Uniform*“ kako bi se robotska ruka ravnomjerno povećavala (slika 5.19).



Slika 5.19 3D postavke robotske ruke_Robot4 - Scale

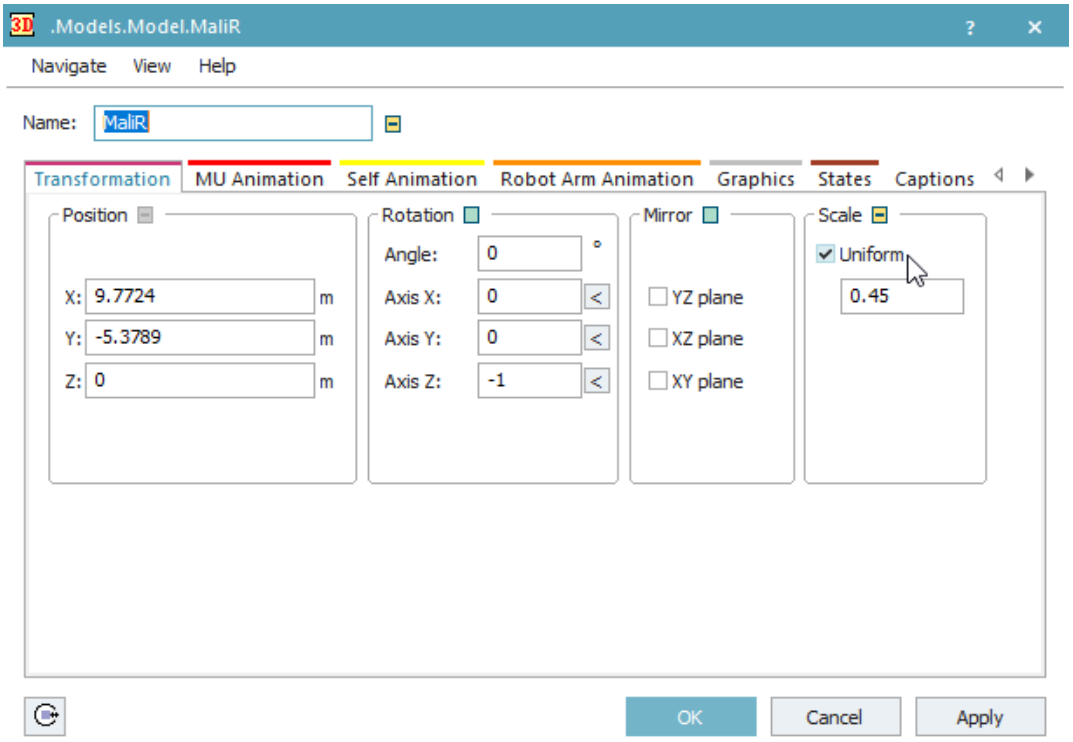
Dodatno potrebno je namjestiti postavke robotske ruke, one se otvaraju dvostrukim lijevim klikom miša na istu. Nakon otvaranja prozora potrebno je promijeniti kut okreta (*Blocking angle*), kapacitet (*Capacity*) te vremena uzimanja i otpuštanja komada. Vrijednosti su prikazane kao što je dano na slici 5.20.



Slika 5.20 Postavke robotske ruke_Robot4

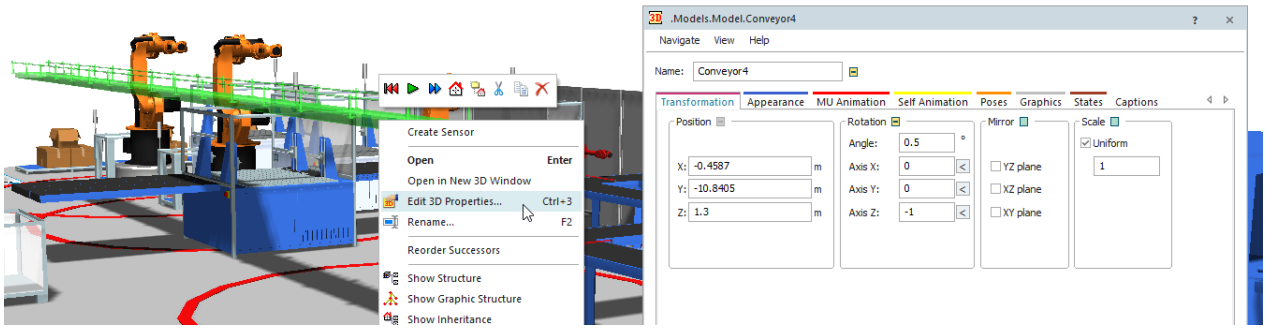
Povezivanje robotske ruke s ostalim objektima bit će opisano u daljnjem dijelu teksta pod dodavanjem velike pokretne trake.

Drugu robotsku ruku (crveni) je također potrebno postaviti prema rasporedu s početka potpoglavlja. Ona se nalazi, kako je već rečeno, unutar stroja. No, kako to simulacijski model ne omogućuje robotska ruka se postavlja izvan ali je izrađena ograda koja predstavlja vanjski dio stroja. Također mijenja se izgled robotske ruke kao i u prethodnom slučaju, na isti način, osim što se odabire druga ponuda „*PickAndPlace Comau*“. Ova robotska ruka se smanjuje. To se radi na isti način kao i povećavanje preko „*Scale*“ opcije, ali u ovom slučaju se veličina mora postaviti na manje od 1 kako bi se robotska ruka smanjivala. Također je važno uključiti opciju „*Uniform*“, a veličina na koju je postavljena *Scale* opcija iznosi 0,45 (slika 5.21). Ostale postavke robotske ruke se ne mijenjaju. Također kao i u prethodnom slučaju, povezivanje ove robotske ruke objasnit će se u daljnjem dijelu teksta pod dodavanjem velike pokretne trake.



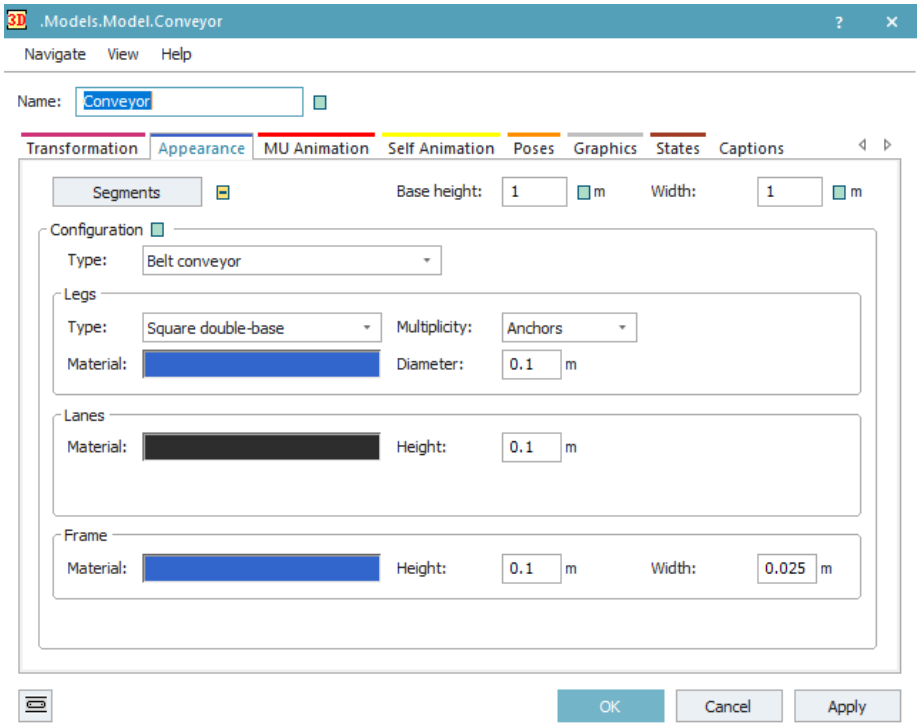
Slika 5.21 3D postavke crvene robotske ruke - Scale

Sljedeća se dodaje viseća pokretna traka. Ona se nalazi na povišenom položaju iz dva razloga. Prvi je pristup transportnog vozila do *Buffera*, drugi je olakšan pristup strojevima radi održavanja i popravaka istih. Kao objekt unutar simulacijskog modela je u potpunosti ista kao i ostale pokretne trake te se na isti način dodaje iz alatne trake *drag and drop* metodom. Duljina trake je u ovom slučaju podešena na 15,5 m, širina na 0,6 m i brzina na 1 m/s. Za visinu trake i ostale postavke iste potrebno je otvoriti 3D postavke. One se otvaraju desnim klikom miša na objekt i u padajućem izborniku pritisne se na „*Edit 3D properties*“ (slika 5.22).

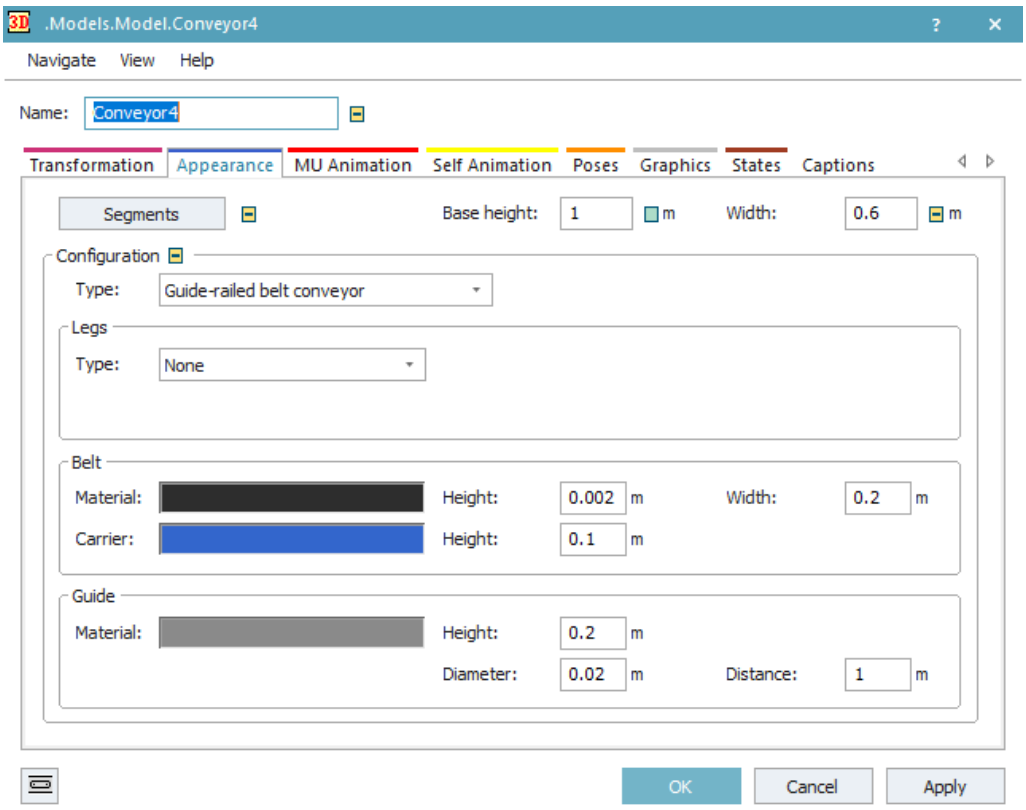


Slika 5.22 3D postavke viseće pokretne trake

Na prvoj kartici namješta se visina pokretne trake. Podešavanje visine se nalazi pod stavkama „*Position*“ te pod „*z*“ prozorom. U ovom slučaju visina je podešena na 1,3 m, no ona u stvarnosti je veća od navedene. Razlog zbog kojeg je visina manja je visina obradnih centara koja je zadana kao standardna. Sljedeće se mijenja izgled, pod karticom „*Appearance*“. Pritiskom na karticu otvara se sljedeći prozor (slike 5.23 i 5.24). Iz prikazanog je vidljivo na koji način treba podesiti postavke viseće pokretne trake. Mijenjaju se sveukupna širina na 0,6 m i vrsta trake na „*Guide-railed blet conveyor*“, dok ostale postavke se automatski mijenjaju pri izboru tipa trake te ih se kao takve i ostavlja.

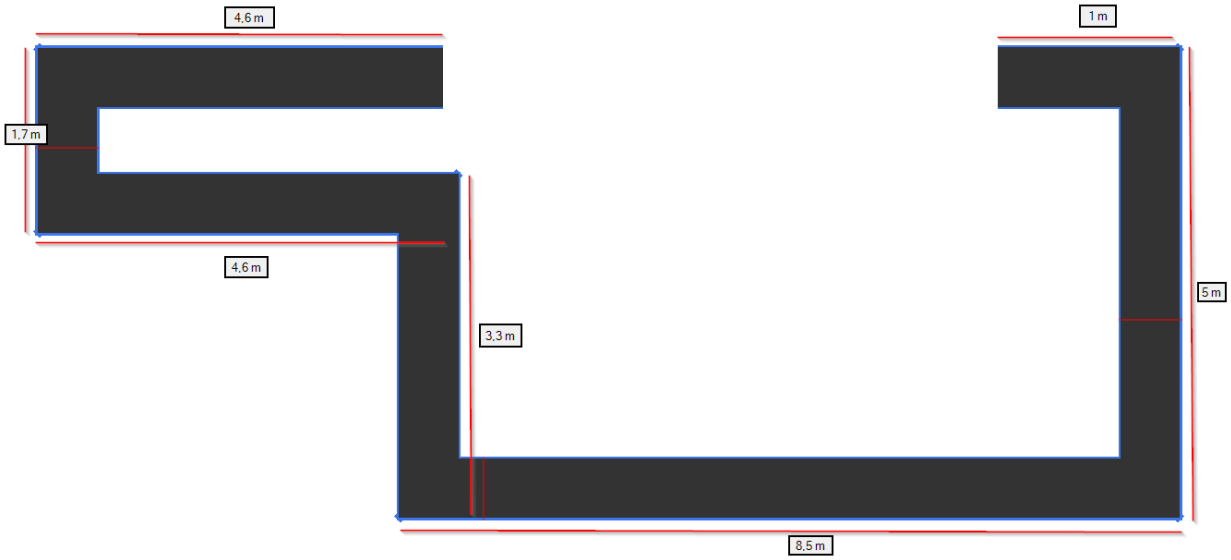


Slika 5.23 Postavke viseće pokretne trake – prije promjena

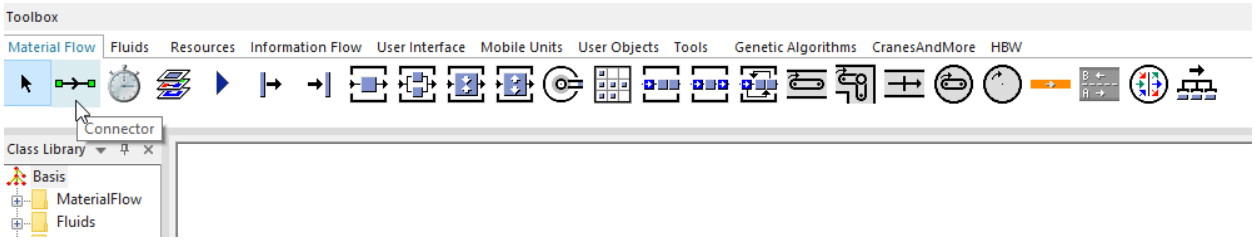


Slika 5.24 Postavke viseće pokretne trake - poslije

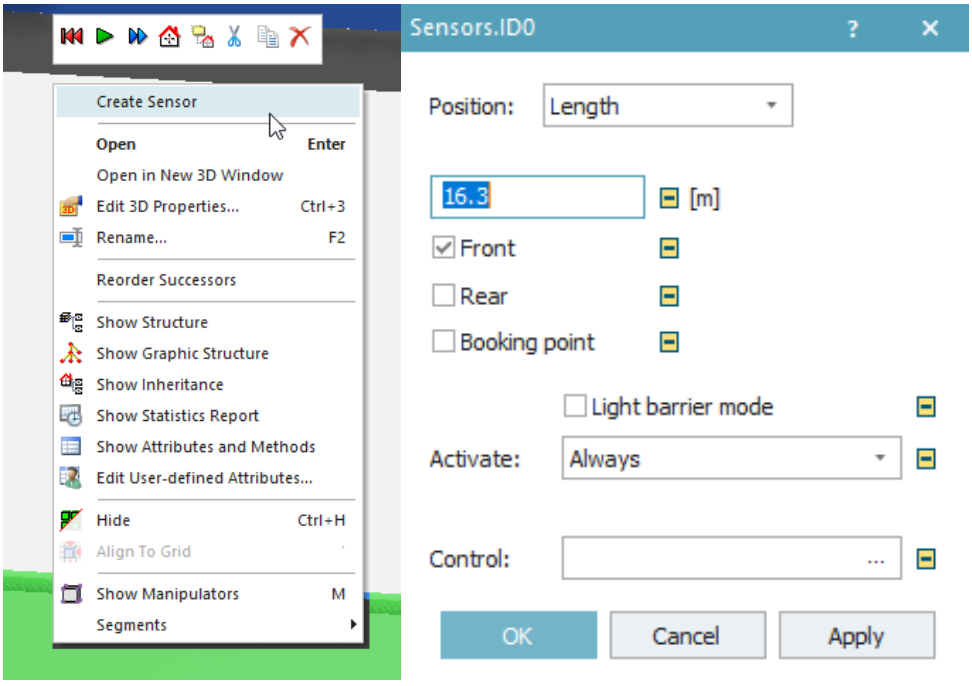
Ne postoji način za promjenu dijelova velike pokretne trake, stoga ju je potrebno izbrisati i ubaciti novu. Postupak ubacivanja je isti, *drag and drop* tehnikom iz alatne trake objekt "*Conveyor*". Prvo uporište postavlja se na izlazu iz perilice te se prva dimenzija mijenja s 1 m na 4,6 m. Sljedeća uporišna točka se nalazi okomito dolje. Okomitost se postiže pritiskom na tipku *Shift*, a traku se spušta prema dolje za vrijednost od 1,7 m. Zatim se pokretnu traku povlači prema perilici, također pod pravim kutom, za duljinu 4,6 m. Potom se opet spušta prema dolje za 3,3 m, te se povlači udesno za 8,5 m. Ostale dvije dimenzije ostaju kao i u osnovnoj verziji simulacijskog modela. Odnosno prva prema gore za 5m i zadnja je lijevo za 1m. Prema tome, nova pokretna traka bi trebala izgleda kao na slici 5.25. Postavke nove produljene trake su iste kao i u prethodnoj verziji. Potrebno je provjeriti je li pokretna traka povezana s perilicom, te ako nije potrebno ju je povezati s objektom *Connector* iz alatne trake (slika 5.26). Iz slike 5.25 vidljivo je da postoje crvene crtice koje idu okomito preko pokretne trake. One predstavljaju senzore pokretne trake, a u ovom simulacijskom modelu ih ima pet. Senzori su, u ovom slučaju, svjetlosna barijera na kojoj se zaustavljaju namjenske palete. One se mogu unositi na dva načina te su oba korištena u ovom radu. Prvi način je da se na željeno mjesto na traci pritisne desnim klikom miša te u padajućem izborniku odabere stavka „*Create Sensor*“ (slika 5.27).



Slika 5.25 Produljena pokretna traka



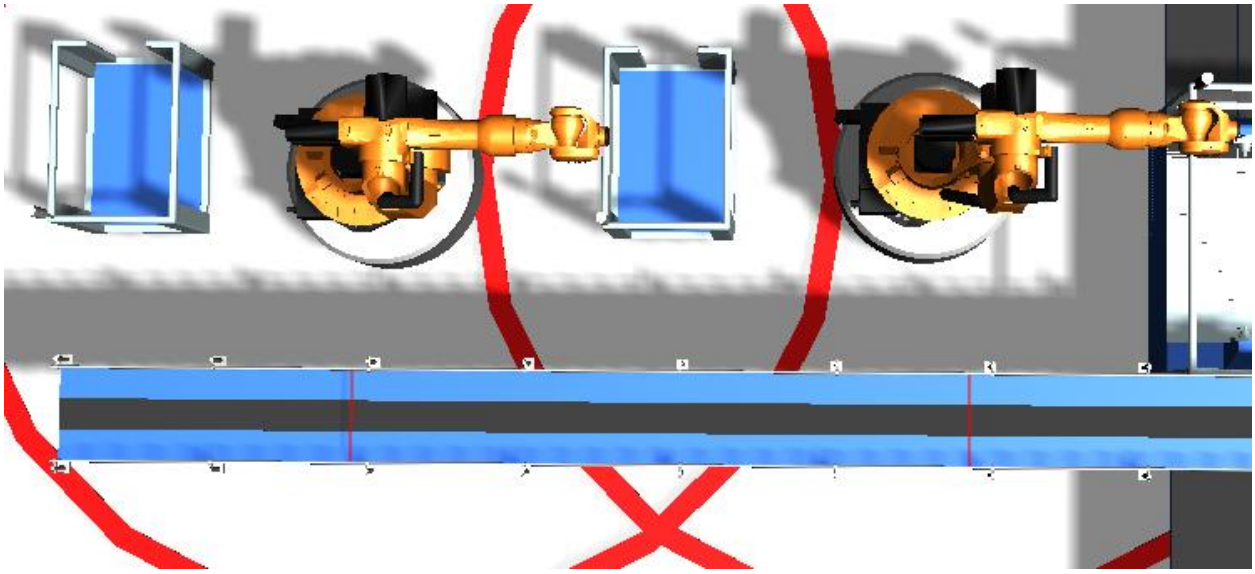
Slika 5.26 Objekt -Connector



Slika 5.27 Izrada senzora – prvi način

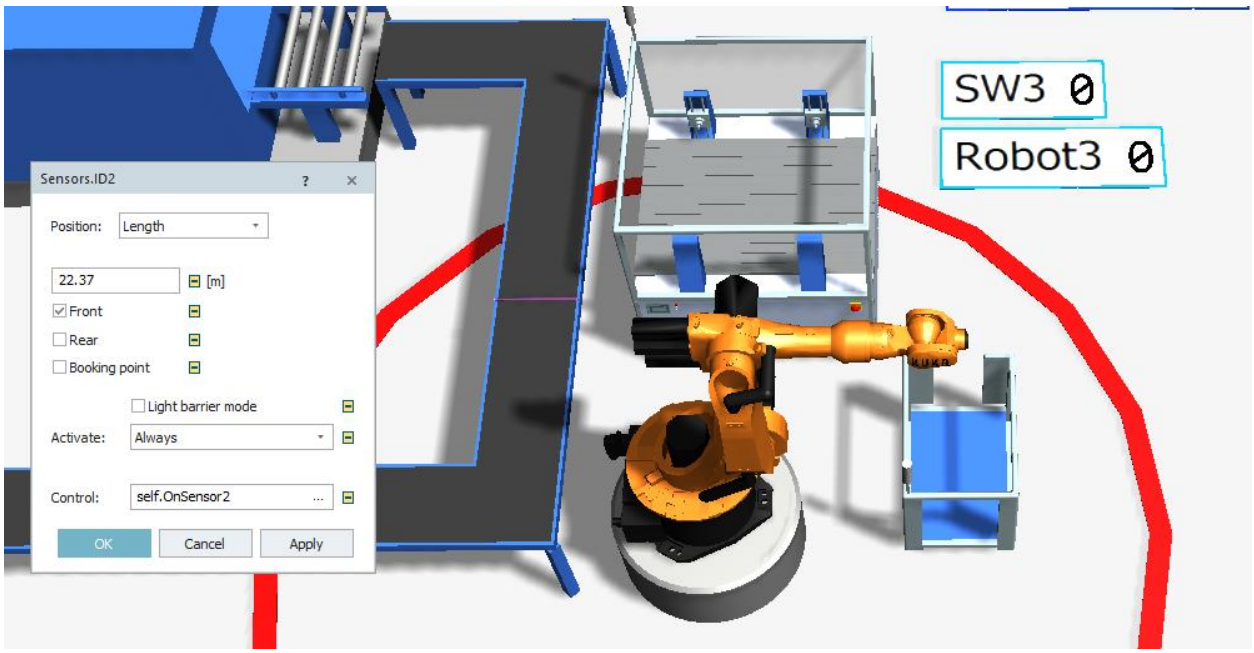
Nakon što se klikne na *Create Sensor* otvara se prozor prikazan na slici 5.27. Za poziciju (*Position*) ostavlja se ponuđena vrijednost „*Length*“. U prozoru ispod se nalazi vrijednost udaljenosti senzora od početka pokretne trake. Tu vrijednost može se mijenjati po želji te tako premjestiti senzor na točno željenu poziciju. Ispod njega nalaze se tri stavke kojima se odabire način na koji će se namjenske palete zaustaviti. Ako se odabere opcija „*Front*“, namjenska paleta će se zaustaviti kada prednji dio namjenske palete aktivira senzor. Ako se odabere „*Rear*“, namjenska paleta će se zaustaviti kada zadnji dio namjenske palete prođe kroz senzor, a ako se odabere „*Booking point*“ isključuje se svjetlosna barijera jer se uzima zadana duljina kao referentna za zaustavljanje namjenske palete. Odabir stavke ovisi o potrebi i poziciji koju se želi da namjenska paleta zauzme, a u ovom slučaju odabrana je opcija *Front*. Za aktivaciju potrebno je ostaviti zadanu vrijednost „*Always*“. Zadnju stavku „*Control*“ će se unositi u tri od ukupno pet senzora, te će se postupak objasniti u daljnjem dijelu teksta.

Prva dva senzora nanesena su na viseću pokretnu traku (slika 5.28). Prvog se unosi na udaljenost od 1,9 m, a drugog na 5,9 m. Oni u ovom slučaju služe kao pozicije na koje će prva i druga robotska ruka postavljati komade, te njihove postavke nije potrebno dirati.



Slika 5.28 Senzori na visećoj pokretnoj traci

Sljedeća tri senzora se razlikuju, a sva tri se nalaze na velikoj pokretnoj traci. Jedan od ta tri senzora je unesen na isti način kao i prethodna dva dok su ostala dva unesena na drugi način. Prvi senzor koji je unesen na već prethodno objašnjen način nalazi se kod treće robotske ruke (Robot3) koja se radila od obradnog centra. Nalazi se na udaljenosti 22.37 m u odnosu na početak pokretne trake, a postavke su iste kao i položaj (slika 5.29).



Slika 5.29 Prvi senzor na velikoj pokretnoj traci

U ovom slučaju potrebno je dodati programski kod, koji se dodaje pritiskom na tri točkice u desnom kutu prozora pokraj stavke „*Control*“. Pritiskom se otvara padajući izbornik s dvije stavke: „*Create control*“ i „*Select object*“. Ako se odabere *Create control* senzor stvara se programski kod unutar samog objekta, dok u slučaju *Select object* programski kod se unosi izvana (postojeći programski kod koji se nalazi unutar radnog prostora). U ovom slučaju odabrana je stavka *Create control*, nakon koje se otvara prozor, slika 5.30, u koji se upisuje programski kod preko kojeg će raditi senzor.

```
param SensorID: integer, Front: boolean, BookPos: boolean

if @.name = "Paletka_11"
    @.stopped:=true
    WT:=@

end
```

Slika 5.30 Programski kod unutar prvog senzora

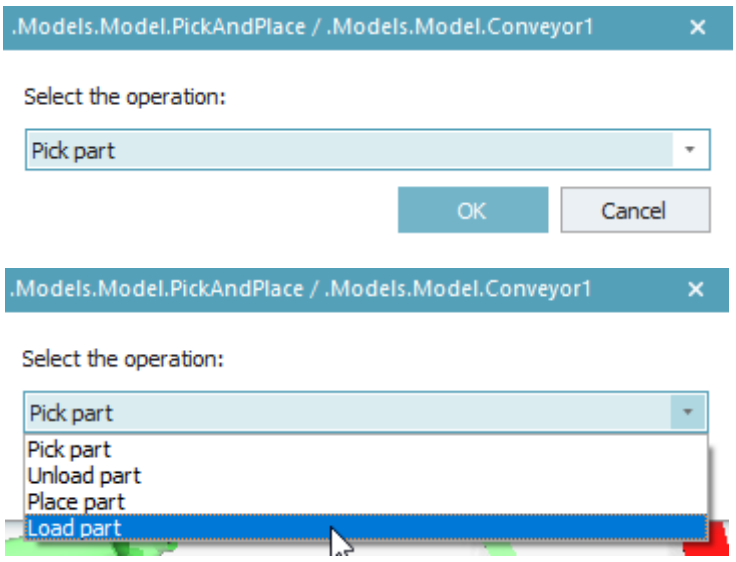
Ovaj programski kod je kratak i jednostavan. Prvi red se automatski pojavljuje pri samom otvaranju. Slijedi *if-end* petlja. Postavlja se uvjet da ako se radi o namjenskoj paleti Paletka_11 (*if @.name = „Paletka_11“*) da ju zaustavlja (*@.stopped:=true*). Simbol „@“ predstavlja entitet koji se kreće unutar simulacijskog modela. U predzadnjem redu nalazi se globalna varijabla, ona je poveznica između programskog koda treće robotske ruke i programskog koda prvog senzora, koja

govori da je WT (globalna varijabla) jednaka entitetu @. Na slici 5.31 nalazi se dio programskog koda robotske ruke (Robot3) u kojem se nalazi globalna varijabla koja je poveznica.

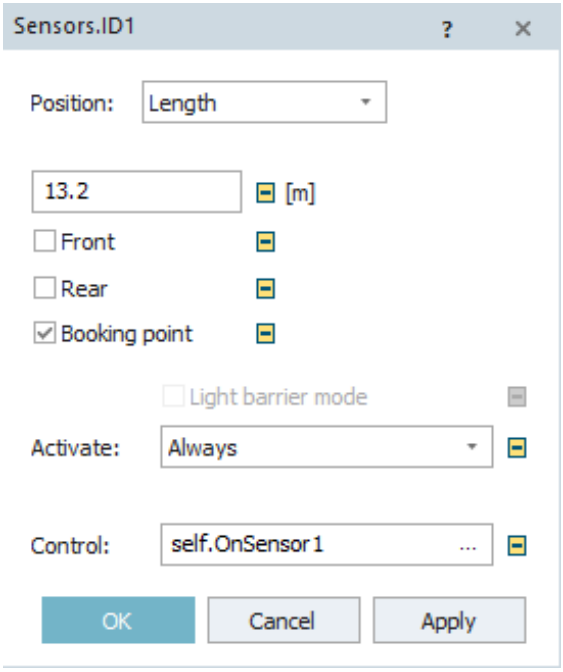
```
repeat
waituntil WT /= void
wait Robot3._3D.poses.moveTo("Conveyor",2)
for i:=1 to 1
    wait 2
    Robot3.cont.move(WT)
next


//Puštanje paletke
waituntil WT.NumMu=2
WT.stopped:=false
WT:=void
until Robot3.empty
```

Slika 5.31 Globalna varijabla u programu robotske ruke_Robot3

Drugi senzor se nalazi kod robotske ruke koja prenosi komade s viseće pokretne trake na veliku pokretnu traku. On se dodaje na drugi način, a taj je da se lijevim klikom miša sa zadržanim pritiskom na robotsku ruku, robotska ruka povuče na pokretnu traku na mjesto gdje se želi stvoriti senzor i pusti tipka miša. Nakon toga otvara se novi prozor (slika 5.32) u kojem se nalazi padajući izbornik s nekoliko mogućnosti. Pri tome odabire se operacija koju će sama robotska ruka obavljati na tom senzoru. U ovom slučaju odabrana je posljednja stavka „*Load part*“ (Postaviti komad, na namjensku paletu). Pritiskom na „*OK*“ stvoren je novi senzor. Otvara se isti kako bi se definirale njegove postavke, kao na slici 5.33. Postavke se otvaraju dvostrukim lijevim klikom miša.



Slika 5.32 Odabir operacije stvorenog senzora



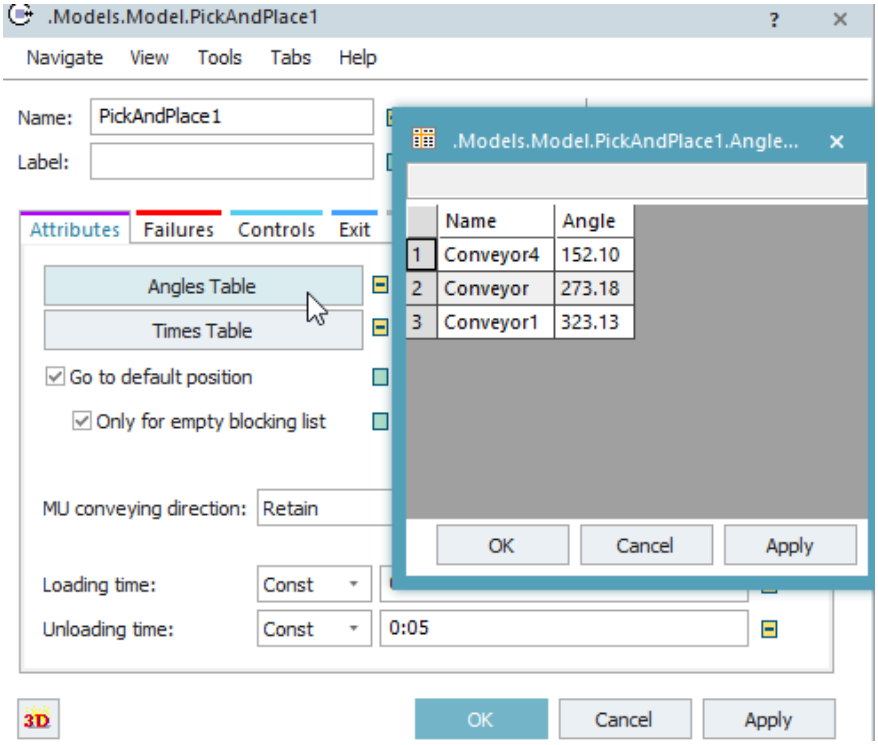
Slika 5.33 Postavke drugog senzora

Također, dodaje se i programski kod koji se nalazi unutar senzora, a izgleda kako je prikazano na slici 5.34. Prvi korak je dodavanje lokalne varijable „i“ tipa *integer*. Nakon toga slijedi naredba „*switch-case-end*“ s umetnutom „*if-else-end“* petljom u koju je umetnuta „*for-next“* naredba. *Switch* naredba funkcionira tako da vrti zadane uvjete, a zadano je da prepoznaje nazive objekata (*@.name*). *Case* (slučaj) govori da u slučaju da je nadolazeći objekt pod navedenim imenom da je potrebno izvršiti određenu radnju. Prvi slučaj je Paletka_1 gdje joj se govori da se zaustavi (*@.stopped:=true*). Zatim *if-else-end* petlja koja govori robotskoj ruci da u slučaju da mu dolaze komadi „Komad_41“ i „Komad_40“ izvrši *for-next* naredbu koja se nalazi unutar iste. S *for-next* naredbom govori se robotskoj ruci da dva komada prenese na Paletku_1. Zatim *else* (drugi uvjet *if-else-end* petlje) ostavlja se praznim i petlju se završava s *end*. Slijedi *waituntil* naredba kojom se govori namjenskoj paleti, Paletka_1, da sačeka dok na njoj ne budu postavljena 2 komada, nakon čega joj se govori da se može nastaviti kretati (*@.stopped:= false*). Potom dolazi drugi slučaj „Paletka_11“, kojem se govori da ako naiđe na senzor, da se zaustavi. I u drugom slučaju nalaze se unutra petlja *if-else-end* i naredba *for-next*. Petljom *if-else-end* postavlja se uvjet imena komada „Komad_41“ i „Komad_40“ kojim robotskoj ruci se govori da ako oni naiđu na pokretnoj traci, da jednog prebaci na Paletku_11. Drugi uvjet petlje, *else*, ostavlja se praznim i petlju se završava s *end*. Slijedi *waituntil* naredba kojom se govori namjenskoj paleti da čeka dok se na njoj ne bude nalazio jedan komad nakon čega se može nastaviti kretati po pokretnoj traci te slijedi završetak *switch* naredbe s *end* naredbom. Poslije toga potrebno je povezati robotsku ruku (Robot4) s visećom pokretnom trakom s pomoću objekta *Connector*.

Važno je prvo kliknuti na pokretnu traku, a potom na robotsku ruku (Robot4). Time se ostvaruje smjer kretanja komada, odnosno redoslijed obavljanja radnji. Popis objekata s kojima je povezana robotska ruka kao i redoslijed obavljanja radnji nalaze se unutar postavki pod stavkom „*Angles table*“ (slika 5.35).

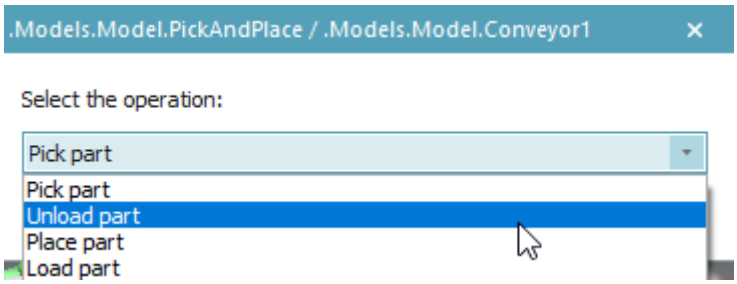
```
param SensorID: integer, Front: boolean, BookPos: boolean
var i:integer
    switch @.name
    case "Paletka_1"
        @.stopped:=true
        if @.name = "Komad_41" and "Komad_40"
                for i:=2 to 0
                    PickandPlace1.cont.move(Paletka_1)
                next
        else
        end
        waituntil @.NumMu = 2
        @.stopped:=false
    case "Paletka_11"
        @.stopped:=true
        if @.name = "Komad_41" and "Komad_40"
            for i:=1 to 0
                PickAndPlace1.cont.move(Paletka_11)
            next
        else
        end
        waituntil @.NumMu = 1
        @.stopped:=false
    end
```

Slika 5.34Programski kod drugog senzora

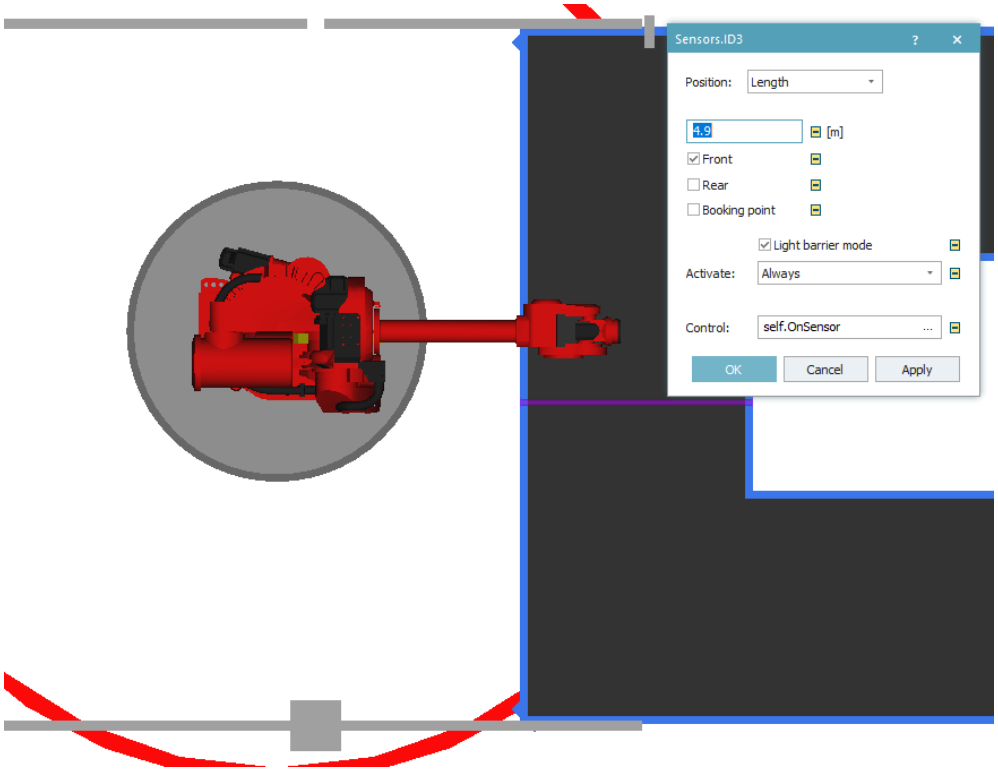


Slika 5.35 Postavke robotske ruke - Angles table

Treći senzor se nalazi pokraj male robotske ruke, unutar montažnog uređaja, na velikoj pokretnoj traci. Senzor se unosi na isti način kao u prethodnom slučaju, povlačenjem robotske ruke na željeno mjesto na pokretnoj traci. U ovom slučaju, kada se otvori prozor za odabir operacije potrebno je odabrati stavku „*Unload part*“ (slika 5.36). Položaj senzora i postavke istog prikazane su na slici 5.37. Ovaj senzor također sadrži u sebi programski kod, koji je unesen s pomoću opcije „*Create control*“. Programski kod ovog senzora dan je slikom 5.38.



Slika 5.36 Operacija trećeg senzora - Unload part



Slika 5.37 Postavke i položaj trećeg senzora

```
param SensorID: integer, Front: boolean

@.stopped := true

var Destination:object := MaliR

while not @.Empty
    var mu: object := @.cont
    if not mu.move(Destination)
        stopuntil mu.location /= @ and not Destination.isLoading
    end
end

@.stopped := false
```

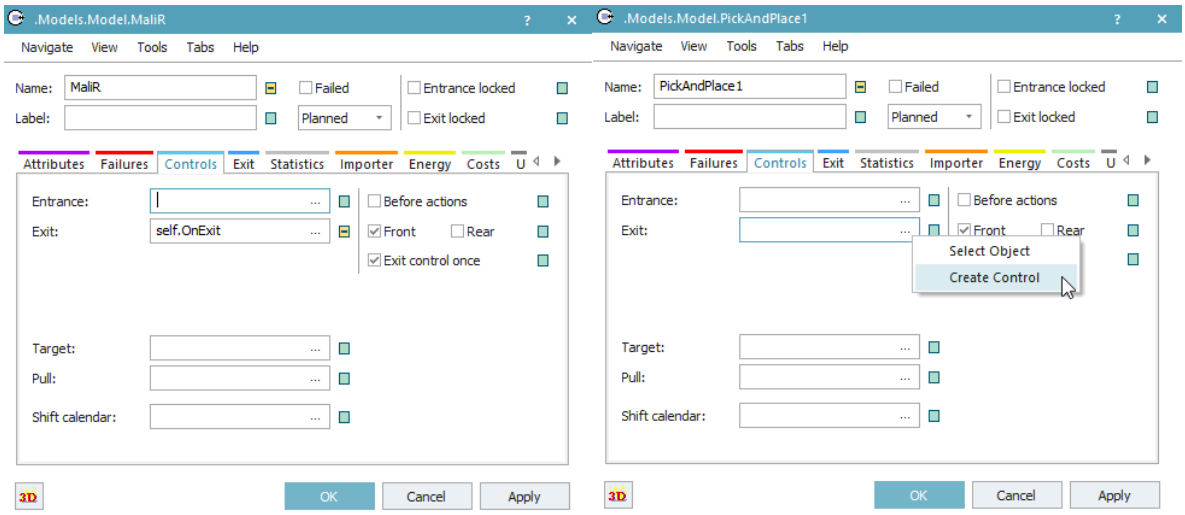
Slika 5.38 Programski kod trećeg senzora

Programski kod trećeg senzora kreće s naredbom za zaustavljanje (*@.stopped := true*) te ona zaustavlja sve namjenske palete koje nadolaze na senzor. Sljedeće se dodaje lokalna varijabla koja označava destinaciju te je zadano da je destinacija objekt MaliR, odnosno mala robotska ruka unutar montažnog uređaja. Zatim slijedi *while not-end* petlja u kojoj se nalazi nova lokalna varijabla i *if-end* petlja. Uvjet *while* petlje je da čeka dok namjenska paleta ne bude prazna (*@.empty*). Sljedeća je nova varijabla koja se unosi. Ona označava vrstu datoteke MU (*Mobile Unit* / mobilna jedinica) te se zadaje da je taj entitet zapravo komad s namjenske palete (*var mu: object := @.cont*). Potom slijedi petlja *if - not* koja govori da ako se komad ne prenosi na destinaciju (*if not mu.move(destination)*) da čeka dok komadi budu na namjenskoj paleti i da čeka ako destinacija komada nije slobodna. Slijede dvije *end* naredbe s kojima se zatvaraju *while* i *if not* petlje. Nakon njih slijedi naredba kojom se govori namjenskoj paleti da može nastaviti dalje.

Ovaj programski kod je važan zato što mala robotska ruka mora postaviti komade na točno predviđene stege unutar montažnog uređaja. Ako bi se dogodilo da robotska ruka postavi komad na krivu stegu, polomio bi se stezni alat, držači konektora i čepa čime bi nastala havarija unutar stroja. To bi zahtijevalo hitnu intervenciju i zaustavljanje ovog dijela pogona.

Kako bi robotska ruka (MaliR) postavljala komade na za to predviđena mjesta potrebno je napisati programski kod unutar robotske ruke s pomoću kojeg će ona raspoznavati komade i stezna mjesta unutar uređaja. Potrebno je dvostrukim lijevim klikom otvoriti postavke robotske ruke (MaliR) te prijeći na karticu „*Controls*“. Otvara se prozor kao na slici 5.39. Unosi se programski kod na isti način kao i kod senzora, unutar objekta. Pritiskom na tri točke u desnom kraju prozora otvara se padajući izbornik gdje je potrebno odabrati „*Create control*“. Otvara se novi prozor u koji se upisuje željeni kod. Na slici 5.40 dan je napisani programski kod. Sastoji se od jedne *if-elseif-end* petlje i od jedne naredbe. Petlja ima tri uvjeta, a to su imena komada koje robotska ruka

prenosi, te u slučaju svakog uvjeta definirana je pozicija unutar stroja (*Albatross[1,1]*). Pozicije stroja određene su prilikom postavljanja dimenzija stroja, x = 3 i y = 1. Na kraju potrebno je povezati robotsku ruku s montažnim uređajem s pomoću objekta *Connector* kako bi sve funkcioniralo.

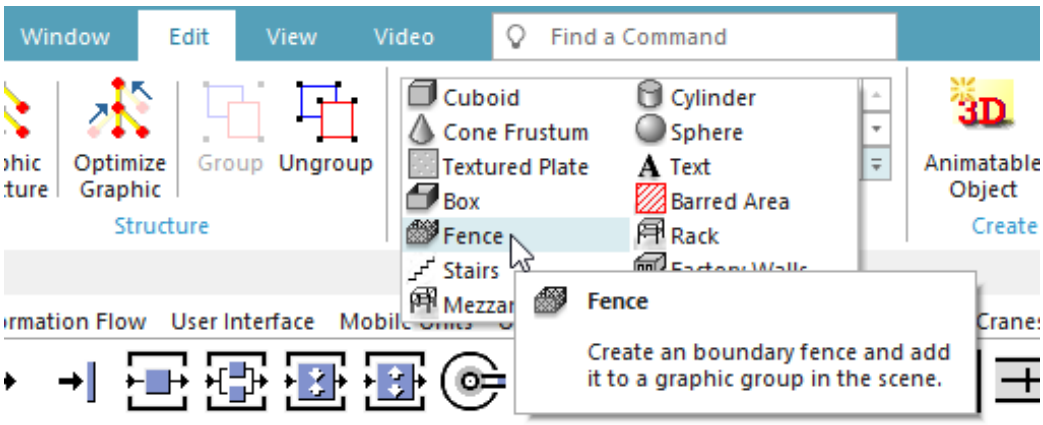


Slika 5.39 Postavke malog robota - prije i poslije

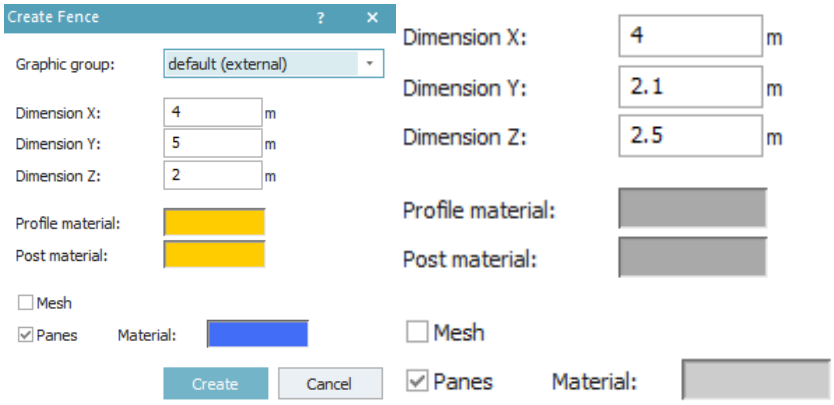
```
if @.name = "Komad_41"
    MaliR.cont.move(Albatross[1,1])
elseif @.name ="Komad_40"
    MaliR.cont.move(Albatross[2,1])
elseif @.name ="Komad_39"
    MaliR.cont.move(Albatross[3,1])
end
return
```

Slika 5.40 Programski kod male robotske ruke

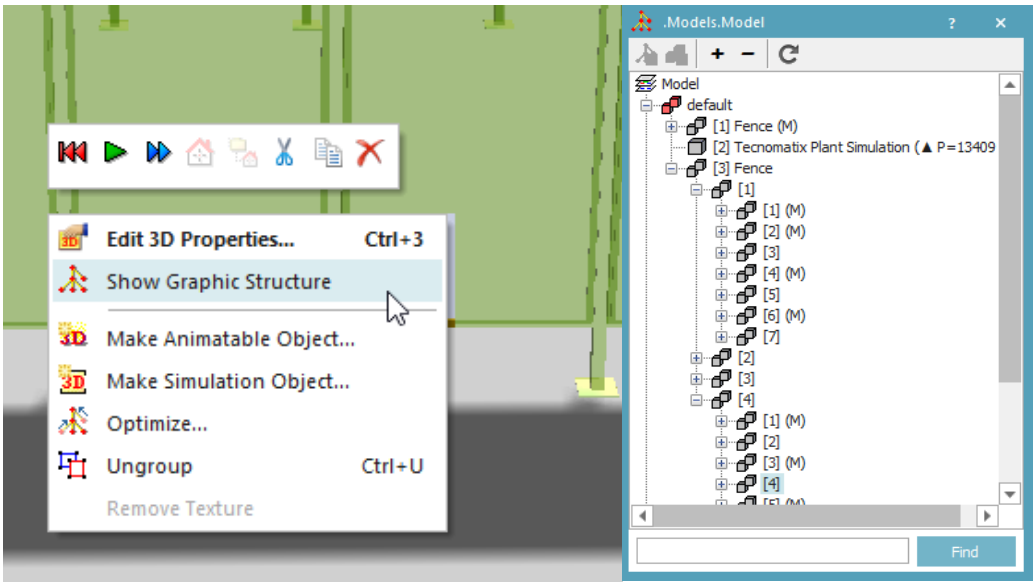
Kako bi poboljšali vizualni dojam da robotska ruka zaista je unutar stroja dodana je ograda. Ograda kao objekt postoji unutar softvera pod karticom „*Edit*“ unutar padajućeg izbornika s ostalim objektima (slika 5.41). Kada se odabere otvara se prozor za postavljanje iste. Dimenzije kao i materijal i boje ograde je potrebno postaviti kako je prikazano na slici 5.42. Kada su postavke definirane potrebno je pritisnuti na „*Create*“ nakon čega softver stvara ogradu unutar radnog prostora koju je potrebno postaviti na željeno mjesto lijevim klikom miša. Sljedeći korak je brisanje nekoliko panela ograde kako bi se omogućio neometan tok komada po pokretnim trakama. To se radi tako da se stisne desni klik miša na istu te u padajućem izborniku odabere „*Show Graphic Structure*“. Nakon tog otvara se novi prozor u kojem su svi dijelovi ograde prikazani u obliku stabla, prikazano slikom 5.43.



Slika 5.41Objekt - Fence/ograda

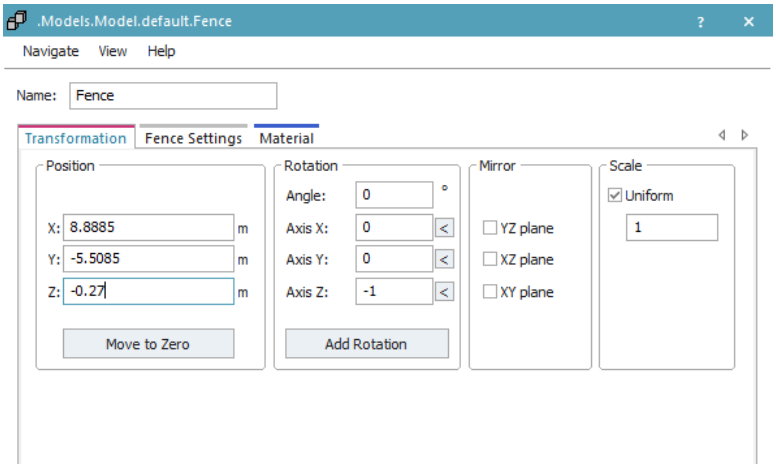


Slika 5.42 Postavljanje ograde – prije i poslije



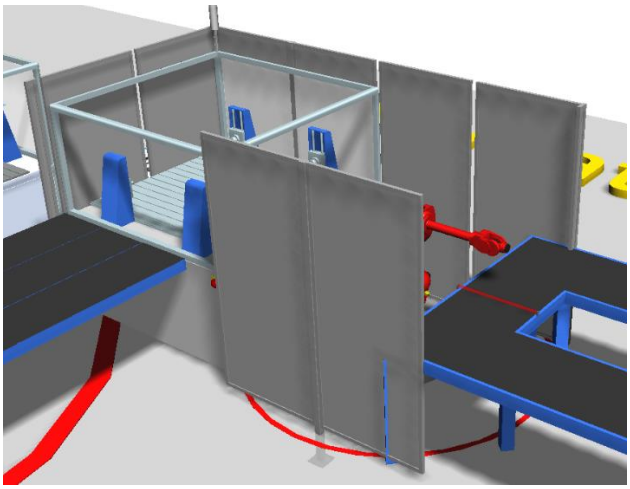
Slika 5.43 Struktura ograde

U strukturi ograde potrebno je proširiti pod brojem tri stavku „*Fence*“, potom stavke „[1]“ i „[4]“. unutar prve stavke potrebno je izbrisati sve od navedenog, dok u stavki „[4]“ potrebno je obrisati samo prvih pet. Time je otvorena ograda. Zatim otvaraju se postavke ograde zbog spuštanja iste, a to se postiže smanjivanjem „z“ vrijednosti. Postavke su prikazane na slici 5.44 te ih je potrebno prema tome definirati.



Slika 5.44 Postavke ograde

Kada su postavke namještene preostaje postaviti ogradu prema prikazu rasporeda strojeva s početka potpoglavlja te rezultat izgleda kao na prikazu 5.45. Kada je sve podešeno simulacijski model bi trebao izgledati prema prikazu s početka potpoglavlja 5.2.



Slika 5.45 Krajnji izgled montažnog uređaja

5.3. Izrada simulacijskog modela za automatizirani proizvodni sustav – verzija s kolaborativnim robotom (kobotom)

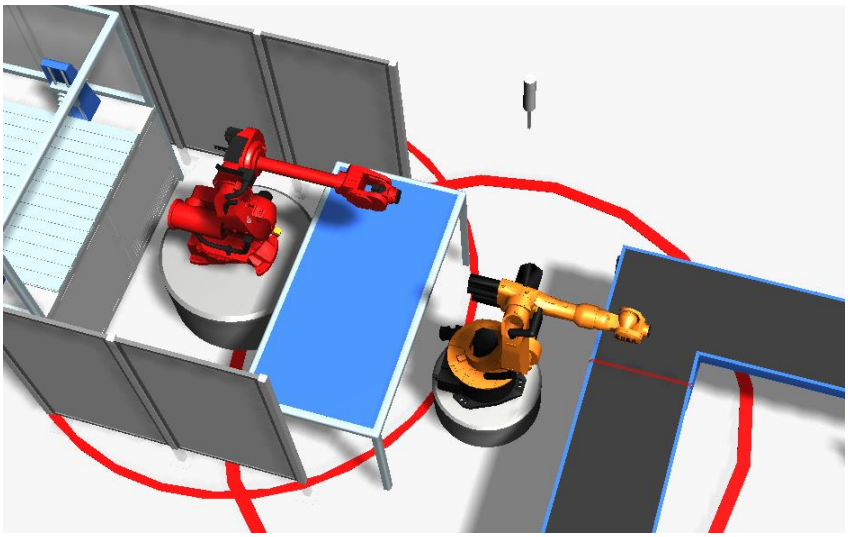
Kroz ovo potpoglavlje bit će objašnjena izrada druge verzije automatizacije promatranog dijela proizvodnog pogona. U ovoj verziji se umjesto produljene trake koristi kobot. Kobot je po svojoj definiciji kolaborativni robot (skraćeno kobot) koji je jedan od nekoliko vrsta robota. Njihovo glavno obilježje je sigurna i učinkovita suradnja s ljudskim radnicima. Bilo je potrebno izraditi više verzija automatizacije kako bi se moglo razmotriti koja verzija je pogodnija. Prva verzija je jeftinija i jednostavnija, no daleko je manje fleksibilna od ove s kobotom. Razlog leži u tome što se ne zna hoće li se produljivati ugovor o proizvodnji navedenih komada dalje od ugovorenog. A verzija s linijom zauzima više prostora u pogonu te može biti neprilagodljiva u slučaju promjena. U verziji s kobotom, koja je znatno skuplja, dobiva se na fleksibilnosti. Prvi razlog je što u slučaju da se nakon dogovorenog razdoblja proizvodnje komada prekine isti potrebno je jedino ukloniti stol s kobotom kako bi se oslobodio prostor. Drugi razlog je što se taj stol s kobotom može prenamijeniti za druge svrhe ovisno o potrebi.

Verzija s kobotom uključuje stol na koji je smješten kobot, a koji bi se nalazio između velike pokretne trake i montažnog uređaja. Na montažnom uređaju potrebno je ukloniti vanjski zid prema kobotu kako bi se omogućio tok komada kako je zamišljen. Dakle, komadi dolaze na namjenskoj paleti iz perilice te se zaustavljaju na senzor na traci, tada kobot preuzima komade i postavlja ih na stol na koji je pričvršćen. Na stolu postoji šest mjesta za odlaganje komada, po dva za svaku poziciju. U svaki od tih šest mjesta je ugrađen senzor koji prepoznaje prisutnost komada te isto javlja maloj robotskoj ruci unutar montažnog uređaja. Tada mala robotska ruka uzima komade sa stola te ih dalje prenosi na obradu.

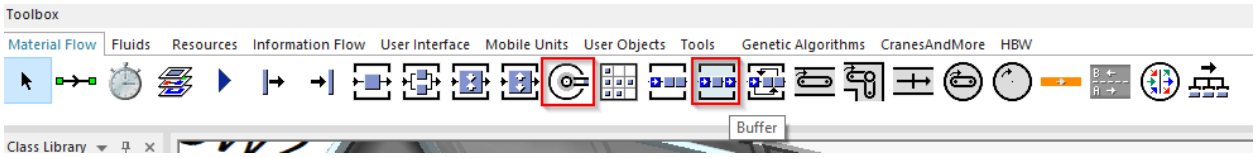
Simulacijski model ove verzije je potpuno isti kao i prethodni, izuzev dijela modela gdje se komadi prenose s namjenskih paleti, na izlazu iz perilice, na daljnju obradu. Stoga će se u daljnjem dijelu teksta objasniti samo taj dio. Na sljedećem prikazu (slika 5.46) vidljiv je izgled dijela simulacijskog modela s kobotom.

Velika pokretna traka je ista kao u osnovnom simulacijskom modelu objašnjenom u potpoglavlju 4.2, te ju je potrebno napraviti prema tim postavkama. Za verziju s kobotom potrebno je dodati, iz alatne trake u radni prostor, jedan *Buffer* te jednu robotsku ruku (*PickAndPlace*) (slika 5.47). Procedura s kobotom je ista kao i u prethodnim slučajevima. Dakle, kobot se postavlja na željenu poziciju, u ovom slučaju pokraj velike pokretne trake na izlazu iz perilice. Izgled se mijenja po prethodno objašnjenom principu, „*Exchange graphics*“, te se odabire „*PickAndPlace Kuka*“. Zatim je potrebno napraviti senzor na velikoj pokretnoj traci po drugom principu, odnosno potrebno je mišem povući kobota na željenu poziciju na velikoj pokretnoj traci gdje se želi stvoriti

senzor. Na novootvorenom prozoru potrebno je odabrati vrstu operacije „*Unload part*“. Time se kobotu daje naredba za skidanje komada s namjenske palete. Sljedeće se namještaju postavke senzora, dakle dvostrukim lijevim klikom na senzor otvaraju se postavke istog te ih je potrebno definirati prema slici 5.48

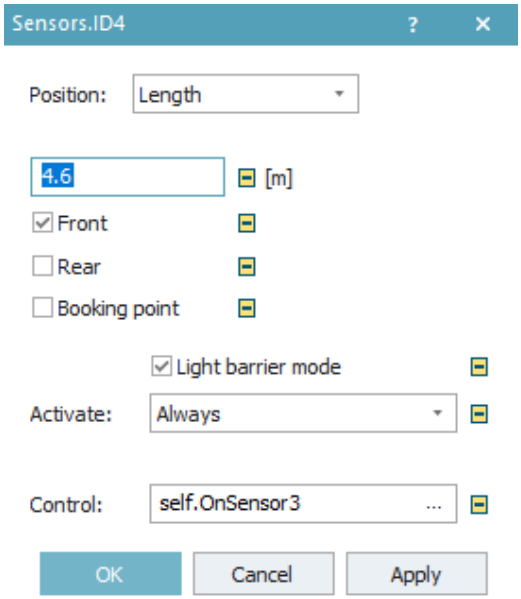


Slika 5.46 Verzija s kolaborativnim robotom - kobotom



Slika 5.47 Dodavanje objekata - Buffer i PickAndPlace

5.46



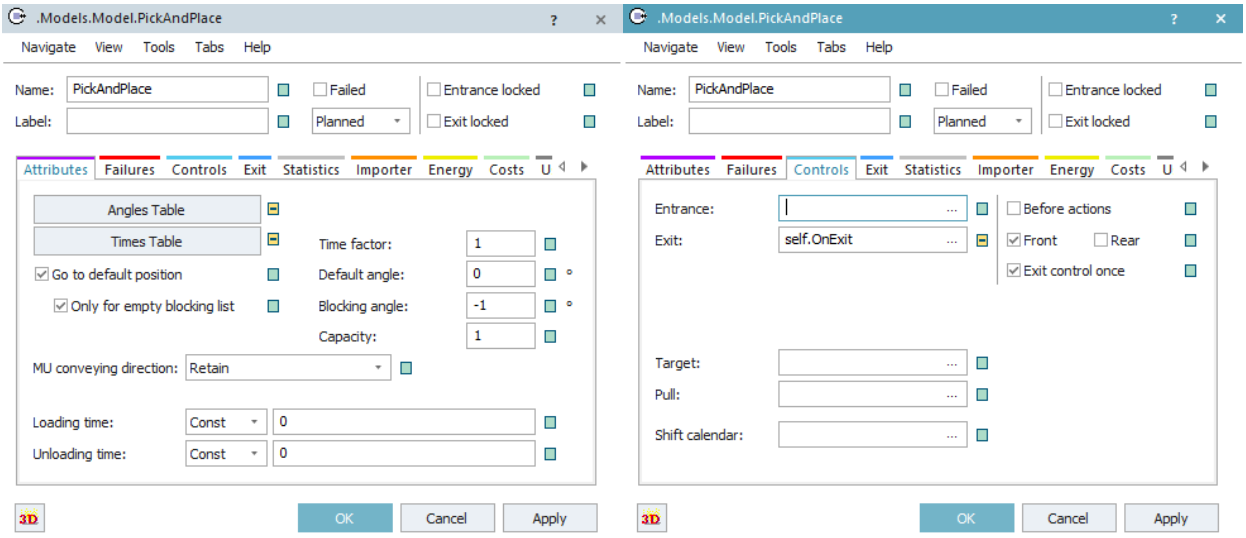
Slika 5.48 Postavke senzora

Također, dodaje se programski kod unutar senzora kako je prikazano na slici 5.49.

```
param SensorID: integer, Front: boolean


@.stopped := true

var Destination:object := PickAndPlace

while not @.Empty
    var mu: object := @.cont
    if not mu.move(Destination)
        stopuntil mu.location /= @ and not Destination.isLoading
    end
end

@.stopped := false
```

Slika 5.49 Programski kod unutar senzora

Programski kod unutar senzora je isti kao i u slučaju male robotske ruke u verziji simulacijskog modela s produljenom trakom. Shodno tome, objašnjenje istog nalazi se u prethodnom potpoglavlju. Nakon toga postavljaju se postavke kobota. Dvostrukim lijevim klikom miša na istog otvaraju se postavke. Njih je potrebno postaviti kako je prikazano na slici 5.50.



Slika 5.50 Postavke kobota

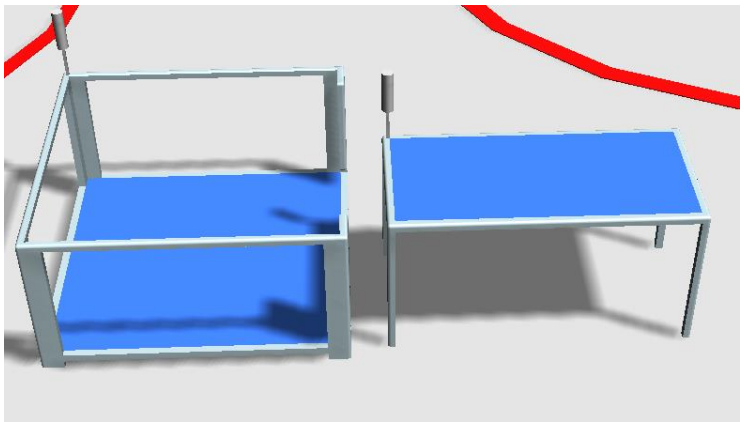
Vidljivo je iz prikazanog da se dodaje izlazna kontrola unutar kobota. Nju se dodaje pritiskom na tri točkice u desnom kraju prozora pod stavkom „*Exit*“ te u padajućem izborniku

odabire se „*Create control*“. U novootvoreni prozor dodaje se programski kod prikazan na slici 5.51. Programski kod je isti kao i u prethodnom slučaju male robotske ruke. Radi se o programskom kodu s pomoću kojeg robotska ruka razaznaje o kojem je komadu riječ te ga postavlja na za to predviđenu poziciju. To je važno zato što su po dva mjesta predodređena za svaku poziciju te bi u slučaju pogrešnog postavljanja komada od strane kobota rezultiralo havarijom unutar montažnog uređaja. Dakle, programski kod se sastoji od *if-elseif-end* uvjetne petlje koja za uvjet postavlja ime (*@.name*=“ime komada“) te za svaki uvjet postavlja naredbu kobotu na koje mjesto u *Bufferu* (*PickAndPlace.cont.move(buffer[3,3])*) treba postaviti isti. Petlja završava s naredbom *end*, a kontrola završava s naredbom *return*. Posljednja naredba govori kobotu da nakon izvršenja petlje da se vrati na početak iste i izvršava ju iznova.

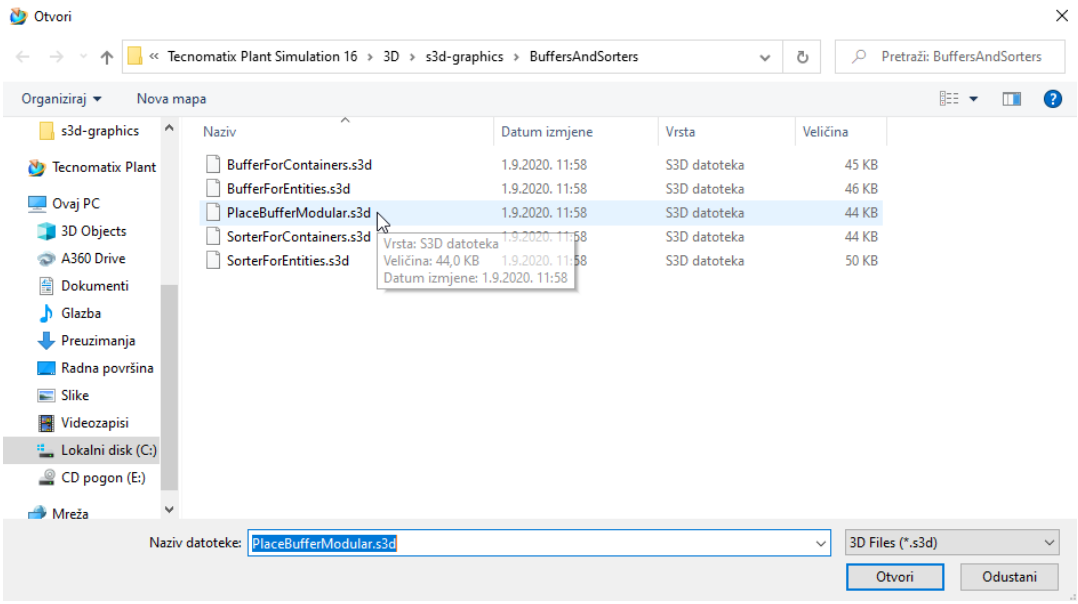
```
if @.name="Komad_41"
    PickAndPlace.cont.move(Buffer[3,3])
elseif @.name="Komad_40"
    PickAndPlace.cont.move(Buffer[1,3])
elseif @.name="Komad_39"
    PickAndPlace.cont.move(Buffer[2,3])
end
return
```

Slika 5.51 Izlazna kontrola kobota

Sljedeće se namješta *Buffer*. Njemu će se promijeniti izgled kako bi izgledao kao stol, a postupak je isti kao i za robotsku ruku osim što se odabire stavka „*PlaceBufferModular*“ (slika 5.52). Novi izgled *Buffera* se nalazi u mapi „*BuffersAndSorters*“ unutar mape sa svim mogućim izgledima zadanih objekata/strojeva (slika 5.53).

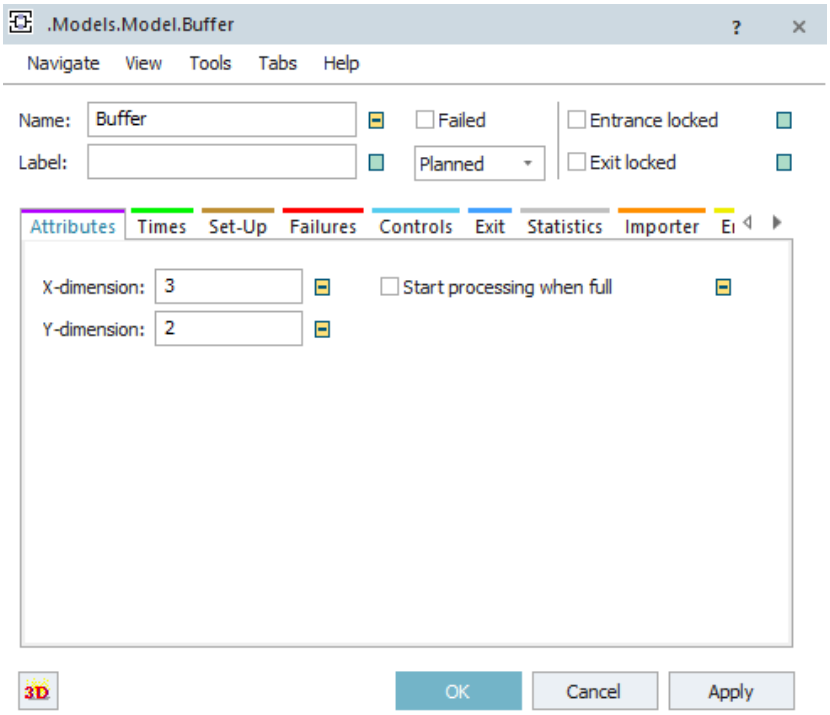


Slika 5.52 Izgled objekta Buffer - prije i poslije



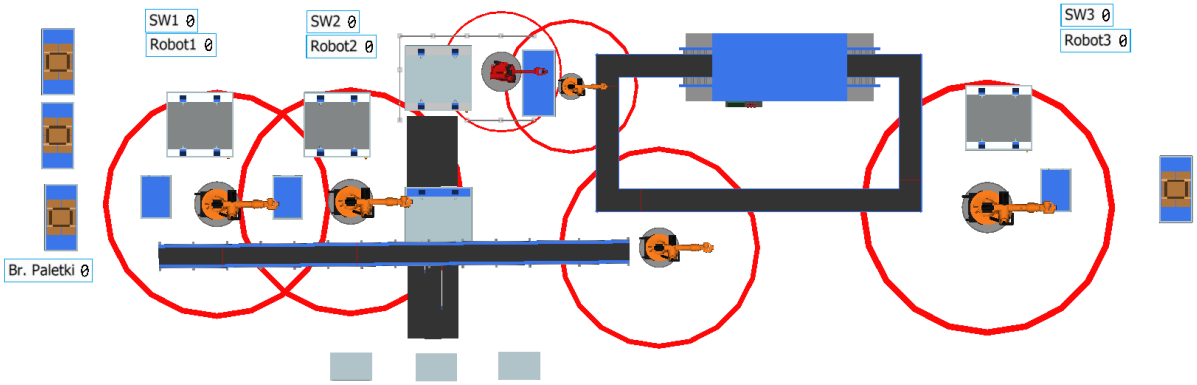
Slika 5.53 Promjena izgleda objekta Buffer

Nakon što je promijenjen izgled objekta *Buffer* potrebno je namjestiti postavke istog. Dvostrukim lijevim klikom miša otvaraju se postavke. U novootvorenom prozoru prvo što se namješta jesu dimenzije stola, odnosno *Buffera*. Kako smo već napomenuli, potrebno je da stol posjeduje po dva mjesta za svaku poziciju odnosno šest sveukupno, stoga ćemo dimenzije postaviti prema slici 5.54 dok ostatak postavki nije potrebno mijenjati.



Slika 5.54 Postavke objekta Buffer - dimenzije

Kada su dimenzije postavljenje potrebno je stol postaviti na lokaciju prema prikazu s početka potpoglavlja 5.3, te sa svime tim tlocrt proizvodnog pogona izgleda kako je prikazano na slici 5.55.

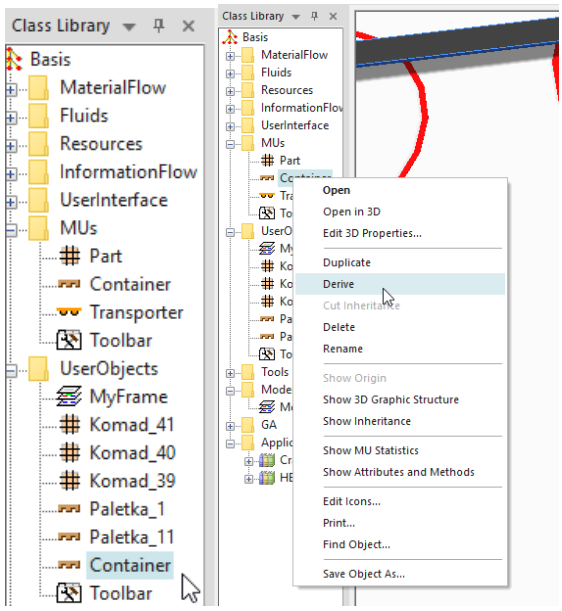


Slika 5.55 Tlocrt simulacijskog modela verzije s kobotom

6. ODREĐIVANJE OPTIMALNOG BROJA NAMJENSKIH PALETA

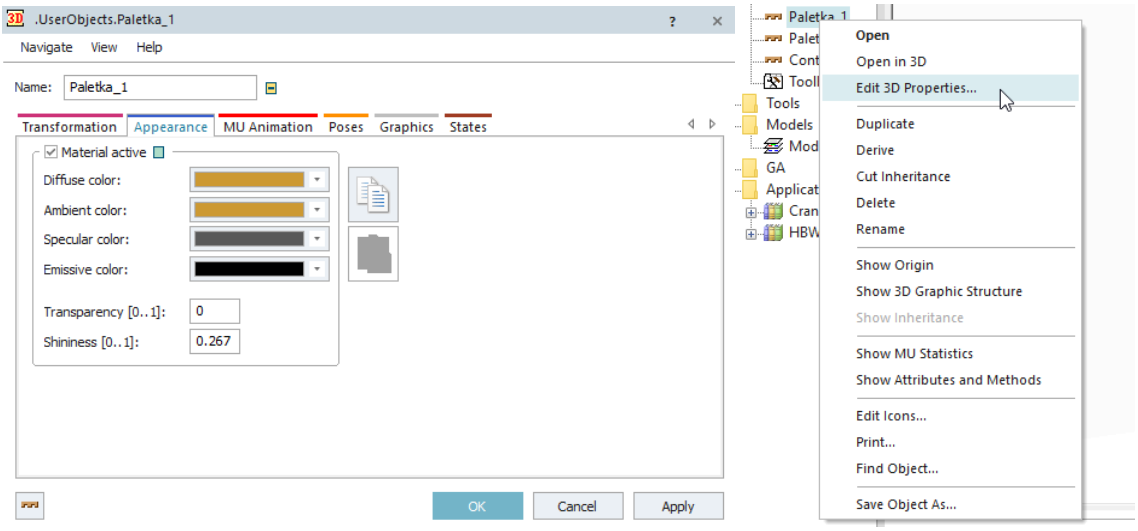
Kroz ovo poglavlje bit će objašnjeno određivanje optimalnog broja namjenskih paleta, no prije toga potrebno je osvrnuti se na problematiku. Na namjenske palete se postavljaju po dva komada te je potrebno da komadi dolaze na montažni uređaj pravilnim redoslijedom kako se ne bi stvaralo usko grlo. Drugim riječima potrebno je da, npr., prvi komad bude 39, 40 pa 41 ili nekim drugim redoslijedom, ali je važno da ne dođu dva komada iste vrste za redom. Kao rješenje problema odlučilo se koristiti dvije vrste namjenskih paleta. Na prvu vrstu, „Paletka_1“, stavljat će se komadi 39 i 40, a na drugu vrstu, „Paletka_11“, stavljat će se po jedan komad 41 i drugi će biti ili komad 39 ili 40. U simulacijskom modelu, u obje verzije, vidljivo je da su namjenske palete različitih boja, narančasta i zelena. Promijenjena im je boja radi lakšeg praćenja i ispitivanja. Kroz ispitivanje je potrebno odrediti za koji broj namjenskih paleta će se proizvesti najviše komada unutar sat vremena.

Namjenske palete dodaju se na sljedeći način, u „*Class libraryu*“ pod stavkom „MUs“ nalazi se objekt „*Container*“ koji predstavlja namjenske palete u ovom slučaju. Na njega je potrebno stisnuti desnim klikom miša te iz padajućeg izbornika odabrati „*Derive*“ (slika 6.1). Kada se to odabere stvara se novi objekt u *Class Library-u* pod stavkom „*User objects*“ i pod nazivom „*Container*“. Taj postupak je potrebno ponoviti još jedan put kako bi se stvorile dvije namjenske palete. Nadalje, potrebno je promijeniti ime prvog *Containera* u „Paletka_1“, a ime drugog u „Paletka_11“.

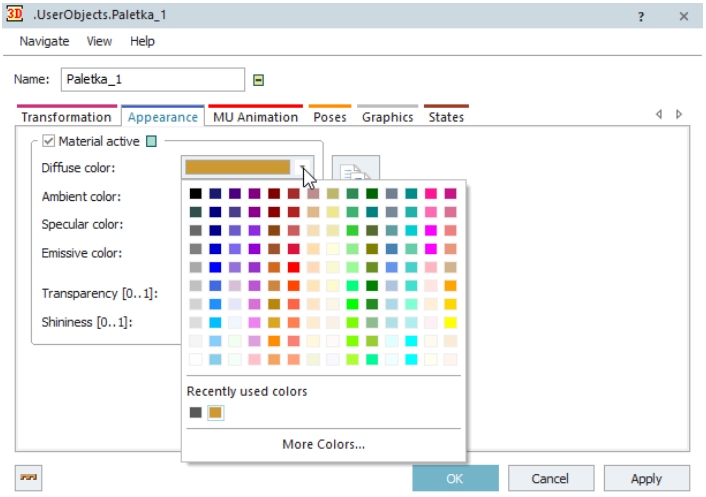


Slika 6.1 Dodavanje namjenskih paleta; promjena imena

Za promjenu boje potrebno je otvoriti postavke namjenskih paleta. One se otvaraju pritiskom desne tipke miša na željenu namjensku paletu te se iz padajućeg izbornika odabere „*Edit 3D Properties*“. U novootvorenom prozoru će se mijenjati boja namjenskih paleta te je za to potrebno prijeći na karticu „*Appearance*“.Prema slici 6.2 potrebno je namjestiti postavke namjenske palete. Prvi korak je aktivirati materijal „*Material active*“ kako bi se otvorila mogućnost promjene boje. Zatim pod stavkom „*Diffuse colo*r“ se mijenja boja. Desno od te stavke postoji strelica te nakon pritiska na istu otvara se novi prozor s cijelom paletom boja. Odabire se boja po želji, a u ovom slučaju je odabrana narančasta (slika 6.3). Nakon odabira klikne se na „*Apply*“ te „*OK*“ kako bi se primijenila promjena boje. Isti postupak se ponavlja za drugu namjensku paletu. Naziv druge namjenske palete, je „Paletke_11“, a boja je zelena.



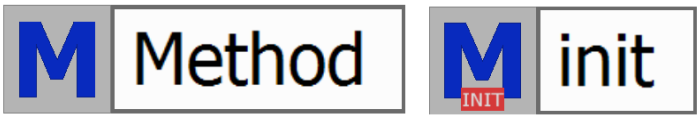
Slika 6.2 Izmjena boje namjenskih paleta



Slika 6.3 Izmjena boje namjenske palete -Paleta boja

Kada bi u ovom trenutku pokrenuli simulaciju, komadi bi se krenuli generirati u isto vrijeme te ne bi mogli osigurati ispravan redoslijed komada. Na prva dva obradna centra komadi se kreću obrađivati u isto vrijeme, što znači da će se ista završiti u isto vrijeme te bi ih robotske ruke prenosile na viseću pokretnu traku u isto vrijeme. Do robotske ruke na kraju viseće pokretne trake dolaze četiri komada 39 pa četiri komada 40. Robotska ruka ih prenosi na namjenske palete te isti, prolazeći kroz protočnu perilicu, dolaze na namjenskim paletama do montažnog uređaja sljedećim redoslijedom: 39-41, 39-41, 39-39, 40-41, 40-41, 40-40.

Da bi osigurali redoslijed komada na namjenskim paletama 39-40, 41-39, 40-41, 39-40, 41-39 potrebno je osigurati različiti početak obrade na obradnim centrima. To se postiže preko „*Init*“ metode. To je obična metoda koja, kada se preimenuje, postaje metoda koju softver pokreće po inicijalizaciji odnosno pri pokretanju. Dakle, metoda se unosi iz alatne trake *drag and drop* metodom u radni prostor. Nakon što je unesena potrebno ju je preimenovati u „*init*“, tim činom metodu pretvaramo u metodu koju softver pokreće po inicijalizaciji (slika 6.4). Zatim ju je potrebno otvoriti dvostrukim lijevim klikom miša. Kada se otvori novi prozor unosi se željeni programski kod, koji se nalazi na slici 6.5.



Slika 6.4 Init metoda – prije i poslije

```
&M_Robot1.executeIn(0)
&M_Robot2.executeIn(7)
&M_Robot3.executeIn(0)
```

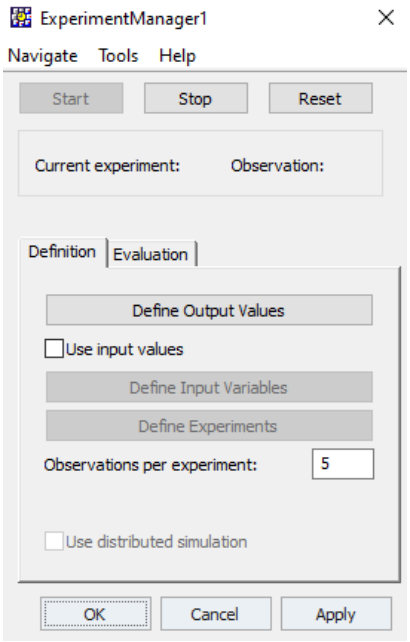
Slika 6.5 Programski kod - Init metoda

U programski kod unesena je nova naredba, *executeIn*, koja govori da se pokrenu metode: „M_Robot1“, „M_Robot2“ i „M_Robot3“. Ona se sastoji od simbola &, naziva metode nakon koje slijedi točka, sam naziv metode „*executeIn*“, te vremena u zagradi. Vrijeme u zagradi je vrijeme odgode nakon koje simulacija će izvršiti navedene metode. Druga metoda, koja je mogla biti i prva, ima odgodu zbog osiguranja pravilnog redoslijeda komada na ulazu u montažni uređaj. Vrijednost od 7 sekundi dobivena je metodom pokušaja i pogrešaka.

Slijedi definiranje postavki izvora namjenskih paleta i postavljanje tablice namjenskih paleta. Kako je već prikazano postavke izvora namjenskih paleta su namještene da proizvode namjenske palete iz tablice slijedno ciklički. To znači da će izvor proizvoditi namjenske palete prema rasporedu iz tablice, a pod stavkom „*Amount*“, u postavkama izvora, postavljaju se količine koje se želi proizvesti. Količina će se trenutno zanemariti, a u nastavku će se objasniti i zašto. Sljedeće se otvara tablica namjenskih paleta u koju se unose količine koje se želi proizvesti. Prethodno je objašnjen potreban redoslijed komada, a kako na Paletku_1 idu komadi 39 i 40 te na Paletku_11 kombinacija komada 41 s komadom 39 ili 40, količina namjenskih paleta mora iznositi kako je prikazano na slici 6.6. S time se osigurava pravilan redoslijed komada na ulazu u montažu. Kako ne bi „pješke“ ispitivali optimalan broj namjenskih paleta korišten je objekt „*Experiment Manager*“. Softver nudi taj objekt pomoću kojeg se mogu vršiti ispitivanja. On se nalazi u alatnoj traci pod karticom „*Tools*“, a unosi se jednostavnom metodom *drag and drop*. Nakon što je ubačen u radni prostor definirane su njegove postavke (slika 6.7).

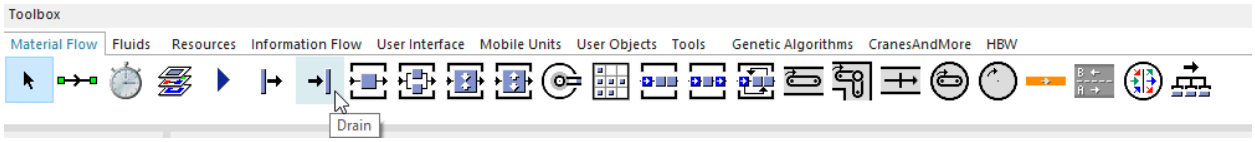
	object 1	integer 2	string 3	table 4
string	MU	Number	Name	Attrib...
1	.UserObjects.Paletka_1	1		
2	.UserObjects.Paletka_11	2		
3				
4				

Slika 6.6 Količina namjenskih paleta



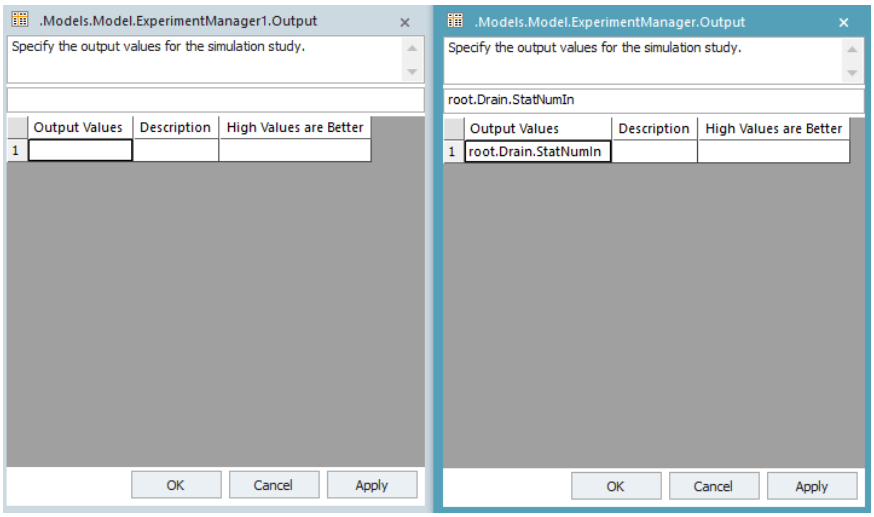
Slika 6.7 Sučelje objekta Experiment manager

Prvi korak je definiranje izlaznih vrijednosti (*Define Output Values*). Pritiskom na tu stavku otvara se novi prozor u koji je potrebno unijeti vrijednost koju se želi da objekt mjeri tijekom izvođenja eksperimenata. Za praćenje je odabran broj komada na izlazu. Da bi se to moglo pratiti potrebno je u radni prostor ubaciti objekt „*Drain*“ koji se nalazi u kartici „*Material Flow*“, a unosi se *drag and drop* metodom (slika 6.8).



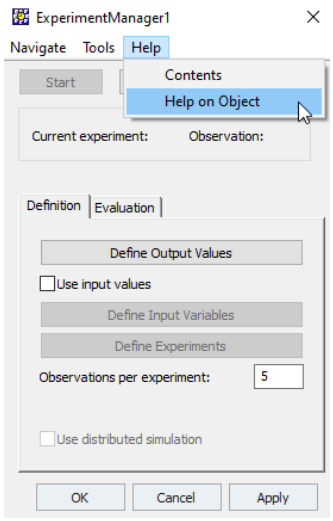
Slika 6.8 Položaj objekta drain u alatnoj traci

Drain je objekt koji u svojoj biti predstavlja izlaz komada proizvodnog pogona, odnosno iz simulacije. Njegove postavke nije potrebno mijenjati. Sljedeće, unosi se *Drain* kao izlazna vrijednost u objekt *Experiment manager* (slika 6.9).



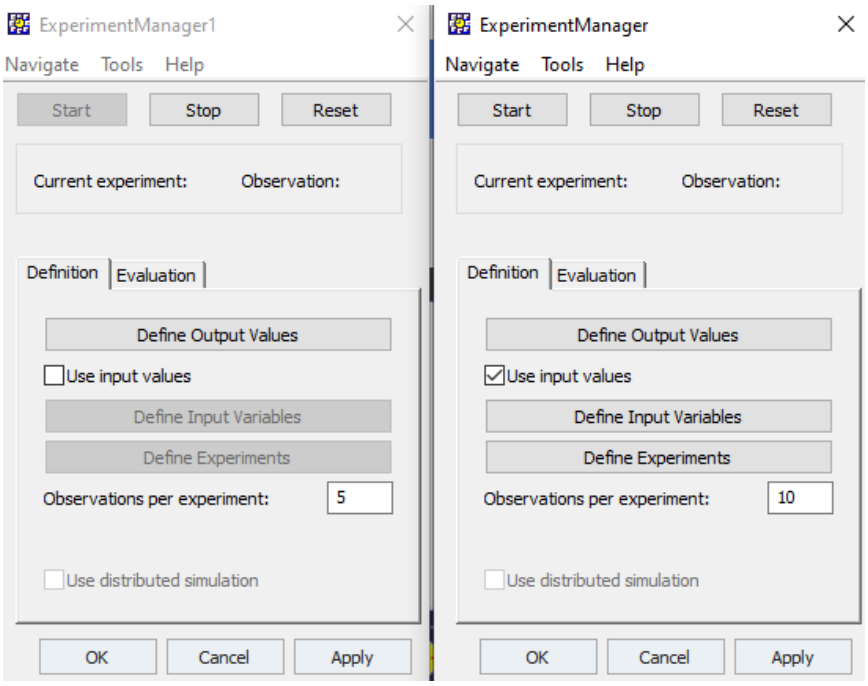
Slika 6.9 Definiranje izlaznih vrijednosti objekta Experiment manager

U prvom polju unosi se vrijednost „*root.Drain.StatNumIn*“ koja govori objektu da broji objekte koji ulaze u objekt *Drain*. Dio „*root*“ označava da se objekt *Drain* nalazi unutar radnog prostora, slijedi naziv objekta te „*StatNumIn*“ koji govori koliko je entiteta ušlo u ovaj objekt. Izuzev naredbe *StatNumIn* postoje i razne druge koje se mogu pronaći unutar „*Help*“ kartice pod stavkom „*Help on object“.* Osim naredbi postoje i sve ostale potrebne informacije o navedenom objektu (slika 6.10).



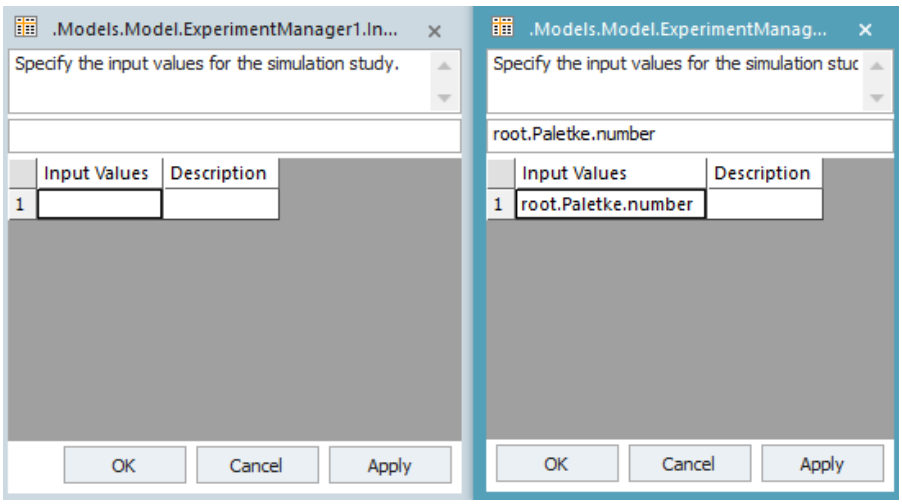
Slika 6.10 Informacije o *objektu Experiment manager*

Zatim se unose ulazne vrijednosti (*Define Input Values*). Da bi se otvorila mogućnost unošenja ulaznih vrijednost potrebno je mali prozorčić označiti s kvačicom koji se nalazi odmah pored „*Use input values*“ (slika 6.11).



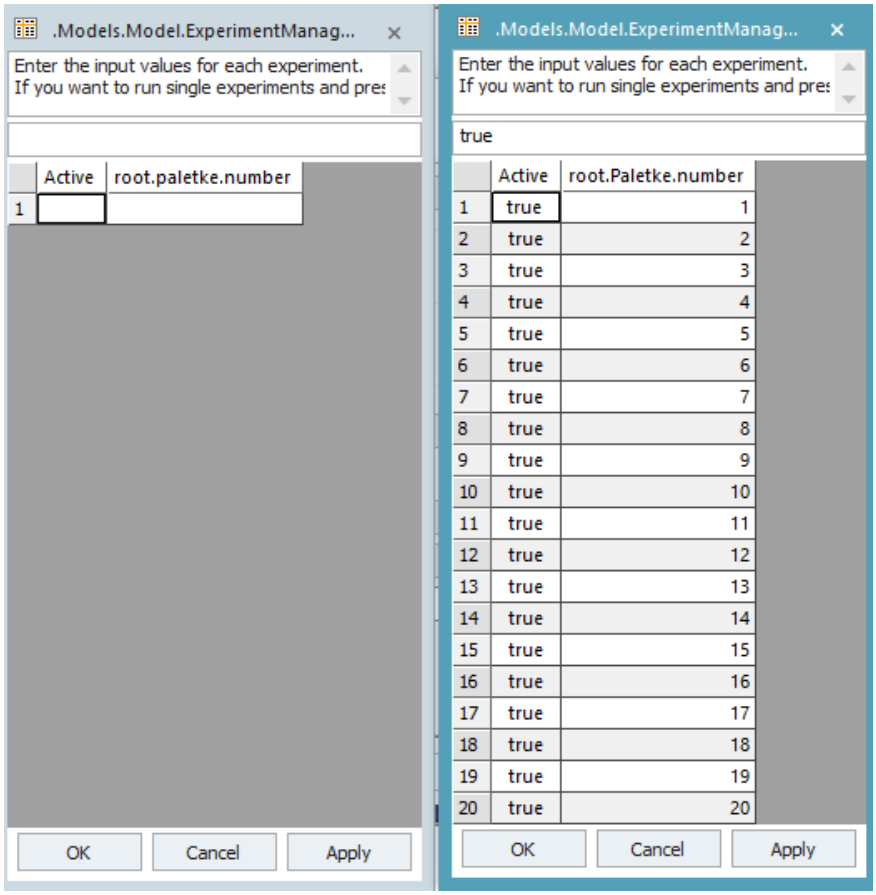
Slika 6.11 Ulazne vrijednosti objekta Experiment manager

Pritiskom na tu stavku otvara se sljedeći prozor vidljiv na slici 6.12.



Slika 6.12 Definiranje ulaznih vrijednosti objekta Experiment manager

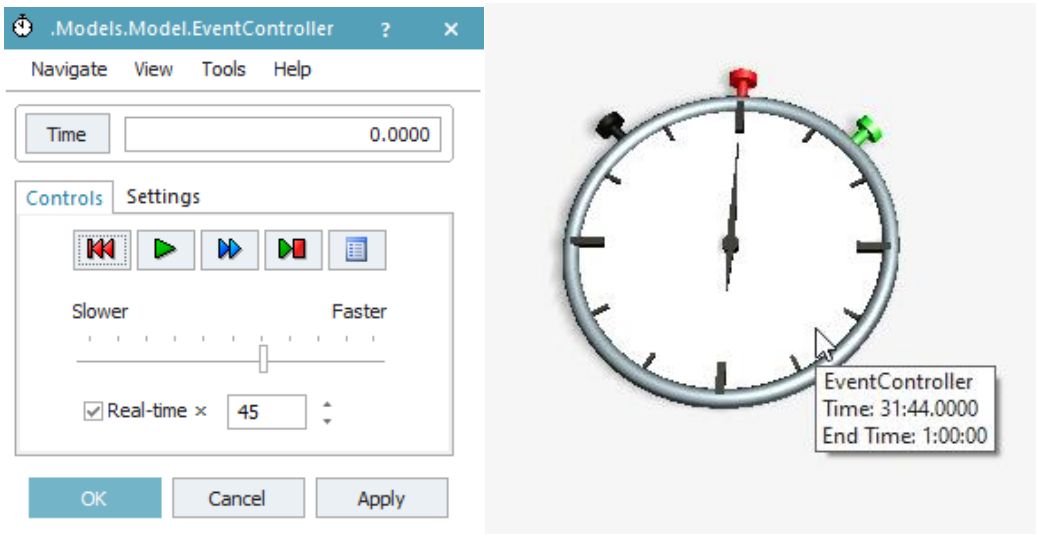
U prvo polje unosi se ulazna vrijednost koja je „*root.Paletke.number*“. Ona govori objektu da kao ulazne vrijednosti uzima broj namjenskih paleta (*number*). Potom se namješta broj eksperimenata koje će objekt izvršiti, stavka „*Define Experiments*“. Kada pritisnemo na nju otvara nam se novi prozor, kao na slici 6.13., u koji se unose eksperimenti koje se želi izvršiti.



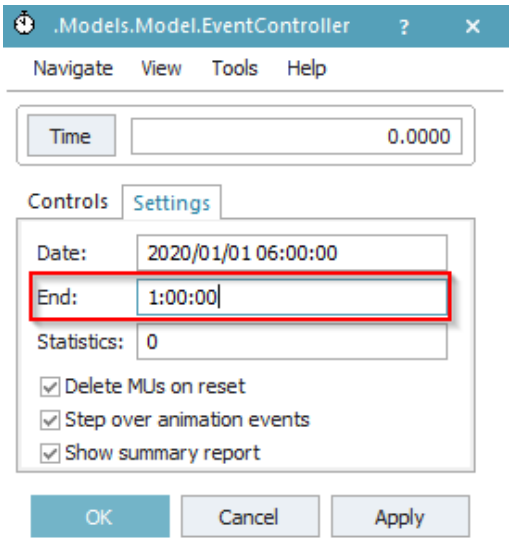
Slika 6.13 Definiranje plana eksperimenata unutar objekta Experiment manager

U prvom stupcu unose se vrijednosti *true* ili *false* te nije moguće unijeti niti jednu drugu. Koju vrijednost će se unijeti ovisi o tome da li se želi određeni eksperiment isključiti iz statistike ili ne. Ako se odabere *false* isključuje ga se, a ako se odabere *true* referentni eksperiment će biti uključen u statistiku. U stupcu pokraj se nalazi ulazna vrijednost koju će objekt mijenjati prema zadanom tijekom izvođenja eksperimenata, u ovom slučaju to je broj namjenskih paleta. Probali se unijeti svaku namjensku paletu posebno ali softver ne dozvoljava tu mogućnost, stoga je unesen ukupan broj namjenskih paleta. Kada se sve unese stisne se na „*Apply*“ te na „*OK*“.

Iz ovog razloga nije unesen broj namjenskih paleta u izvoru za palete jer će objekt *Experiment manager* sam mijenjati vrijednosti tijekom izvođenja eksperimenata. U prethodnom prikazu (slika 6.13) vidljivo je da je postavljeno 20 eksperimenata te broj namjenskih paleta iz eksperimenta u eksperiment raste linearno za po jedan. Sada su povezani objekti: *Experiment manager,* izvor za namjenske palete te tablica paleta. Povezani su tako da kada *Experiment manager* izvodi prvi od dvadeset testova izvor za namjenske palete će proizvesti samo jednu namjensku paletu i to prvu navedenu unutar tablice. Prilikom izvođenja sljedećeg eksperimenta izvor će proizvesti dvije palete, jednu paletku 11 i jednu paletku 1. U trećem eksperimentu izvor za namjenske palete će proizvesti jednu paletku 1 te dvije paletke 11, a u posljednjem eksperimentu će proizvesti sedam paletki 1 i trinaest paletki 11. Postoji mogućnost uključivanja stavke „*Generate as batch*“ kojom će izvor za namjenske palete proizvoditi iste u jediničnim serijama prema zadanim količinama u tablici. Drugim riječima, izvor će, u slučaju s uključenom stavkom, za prvi eksperiment proizvoditi jednu palektu1, u drugom eksperimentu će proizvesti jednu paletku 1 i dvije paletke 11, u trećem slučaju će proizvesti dvije paletke 1 i dvije paletke 11 te u krajnjem eksperimentu proizvesti će 10 x Paletka_1 čija količina u tablici je 1 što iznosi 10 namjenskih paleta 1 i 10 x Paletka_11 čija količina iznosi 2 što nam daje 20 namjenskih paleta 11 i ukupno to je 30 paleta. Dakle, glavna razlika je u tome što izvor bez uključene opcije (*Generate as batch*) proizvodi po jednu namjensku paletu za navedenu količinu (*Amount*; unutar postavki izvora) prema postavljenom rasporedu u tablici, a u slučaju uključene opcije proizvodi entitete prema redoslijedu zadanom u tablici prema količinama iz tablice. Posljednji korak je namještanje vremenskog okvira za izvođenje eksperimenata, koje se postavlja na sljedeći način. Prvi korak je otvaranje „*EventControllera*“ (6.14). Otvaraju se postavke istog prelaskom na karticu „*Settings*“. U postavkama je potrebno namjestiti stavku „*End*“. Ona označava kraj simulacije, odnosno kada se unese željena vrijednost, željeno trajanje simulacije, te pokrene ista ona će završiti za zadano vrijeme. U ovom slučaju postavljeno vrijeme je sat vremena, dano slikom 6.15, jer je to vremenski okvir u kojem je potrebno ispitati broj namjenskih paleta za najveću količinu proizvedenih komada.



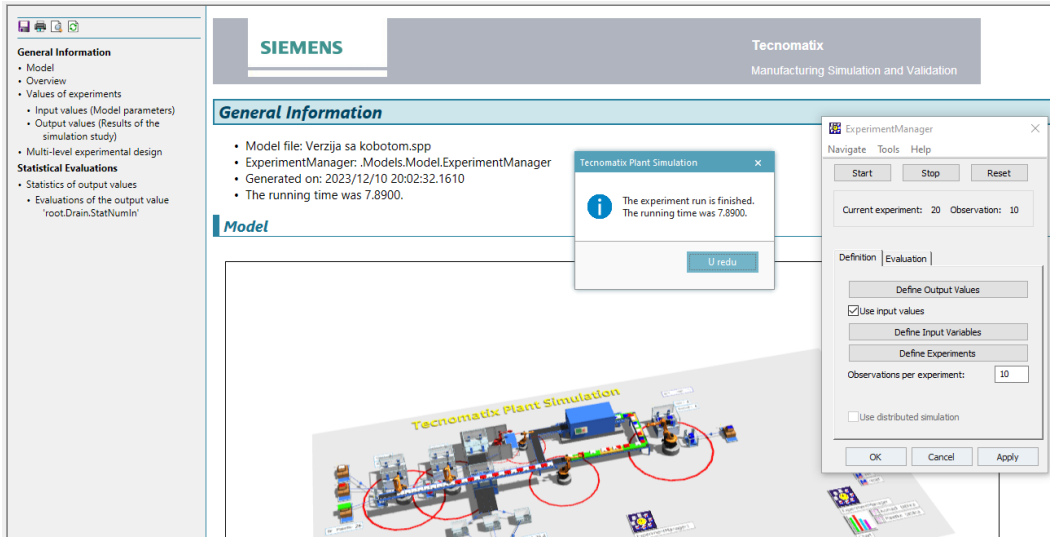
Slika 6.14 Objekt - Event controller



Slika 6.15 Event controller - postavljanje završetka simulacije

Kada su sve vrijednosti postavljene u *Eksperiment Manageru* može se pokrenuti izvođenje eksperimenata. To se postiže pritiskom na stavku „*Start*“ unutar istog objekta (slika 6.16). Pritiskom na tipku *Start* objekt pokreće izvođenje eksperimenata čiji napredak je vidljiv pokraj stavke „*Current experiment*“ ispod tipke *Start*. Na kraju otvara se novi prozor unutar programa. Rezultati ispitivanja (slika 6.16) se sastoje od više stavki od kojih će se prikazati samo dvije, osvrt te statistička evaluacija. Važno je za napomenuti da je u ovom slučaju bila uključena opcija „Generate as batch“. Testovi će se ponoviti i za slučaj kada je opcija unutar izvora isključena, te će ih se na kraju usporediti. Prvi naveden je osvrt na eksperimente u obliku tablice, prikazan slikom 6.17. Tablica je podijeljena na eksperimente (*exp*) po redovima te na broj namjenskih paleta i

količinu komada unutar *Draina* po stupcima. Prema tome, može se iščitati da unutar eksperimenta 9 je napravljeno najviše komada u sat vremena, a iznosi 196 komada. Statistička evaluacija je prikazana u vidu točkastog dijagrama, dano slikom 6.18.



Slika 6.16 Rezultati ispitivanja - uključena opcija „Generate as batch“

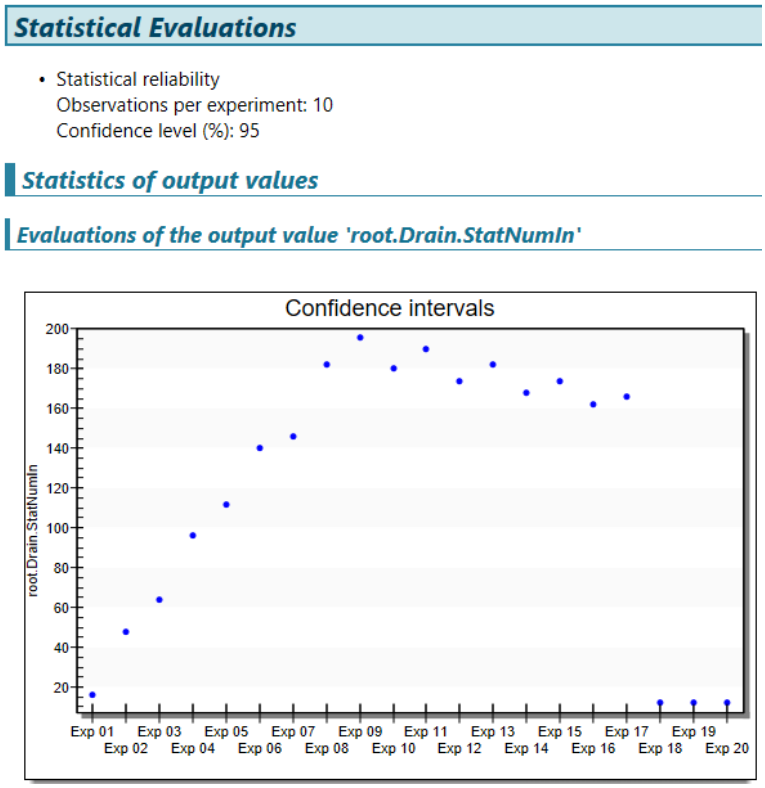
Overview

Overview of all executed experiments, their parametrizatior
The number of decimal places of the results is -1.

	root.Paletke.number	root.Drain.StatNumIn
Exp 01	1	16
Exp 02	2	48
Exp 03	3	64
Exp 04	4	96
Exp 05	5	112
Exp 06	6	140
Exp 07	7	146
Exp 08	8	182
Exp 09	9	196
Exp 10	10	180
Exp 11	11	190
Exp 12	12	174
Exp 13	13	182
Exp 14	14	168
Exp 15	15	174
Exp 16	16	162
Exp 17	17	166
Exp 18	18	12
Exp 19	19	12
Exp 20	20	12

Simulation effort: 20 experiments with 200 simulation runs
No special diagrams

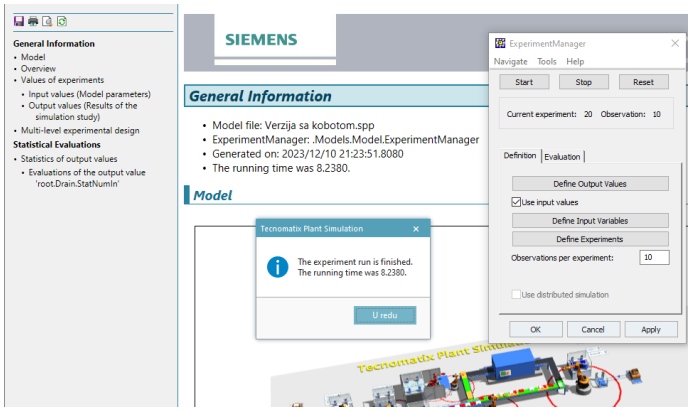
Slika 6.17 Rezultati provođenja 20 eksperimenata kroz 200 pokrenutih simulacija - uključena opcija „Generate as batch“



Slika 6.18 Statistička evaluacija 20 provedenih eksperimenata - uključena opcija „Generate as batch“

Iz dijagrama je vidljiv trend rasta broja komada do devetog eksperimenta nakon kojeg vrijednosti opadaju. Rast i pad vrijednosti ne prati pravac zato što postoje dvije vrste namjenskih paleta na koje se ne postavljaju komadi jednako. U slučaju kada je više paletki 11, na pokusima parnih brojeva, stvara se usko grlo na prijelazu s viseće pokretne trake na veliku pokretnu traku. Posljednja tri eksperimenta imaju vrijednosti 12, vidljivo iz slike 6.17, zato što se stvara velik broj namjenskih paleta te dolazi do opterećenja velike pokretne trake i perilice.

Slijedi provođenje eksperimenata bez uključene opcije „*Generate as batch*“(slika 6.19).



Slika 6.19 Rezultati ispitivanja - isključena opcija „Generate as batch“

Rezultati se razlikuju u odnosu na prethodni slučaj, no opet se poklapaju određene vrijednosti, što je bilo za pretpostaviti. Slikom 6.20 dani su dobiveni rezultati nakon provođenja 20 eksperimenta, a najveći iznos se nalazi pod eksperimentom 13 i također iznosi 196 komada.

Overview

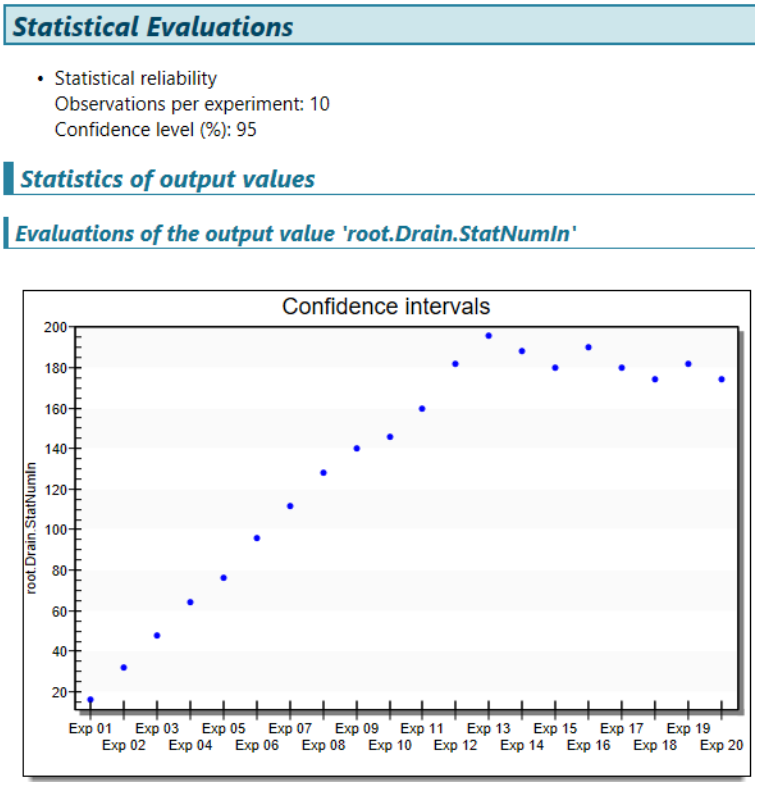
Overview of all executed experiments, their parametrizations
The number of decimal places of the results is -1.

	root.Paletke.number	root.Drain.StatNumIn
Exp 01	1	16
Exp 02	2	32
Exp 03	3	48
Exp 04	4	64
Exp 05	5	76
Exp 06	6	96
Exp 07	7	112
Exp 08	8	128
Exp 09	9	140
Exp 10	10	146
Exp 11	11	160
Exp 12	12	182
Exp 13	13	196
Exp 14	14	188
Exp 15	15	180
Exp 16	16	190
Exp 17	17	180
Exp 18	18	174
Exp 19	19	182
Exp 20	20	174

Simulation effort: 20 experiments with 200 simulation runs
No special diagrams

Slika 6.20 Rezultati provođenja 20 eksperimenata kroz 200 simulacija isključena opcija „Generate as batch“

U točkastom dijagramu danom na slici 6.21, vidljiv je pravilniji rast komada na objektu *Drain*. Razlog leži u načinu stvaranja namjenskih paleta. U ovom slučaju je isključena opcija stvaranja u jediničnim serijama te se namjenske palete stvaraju jedna po jedna čime se omogućuje podrobnije ispitivanje optimalnog broja namjenskih paleta. No, najveći broj dobivenih komada se dobiva za isti broj namjenskih paleta, a to je 13. U prvom slučaju najveći broj komada ostvaren je pod eksperimentom 9, odnosno 5 x Paletka1 i 4 x 2 Paletka11. Navedeno u sumi daje 13 namjenskih paleta. U drugom slučaju je suma isti broj namjenskih paleta, odnosno 5 x Paletka1 i 8 x Paletka11.



Slika 6.21 Statistička evaluacija 20 provedenih eksperimenata - isključena opcija „Generate as batch“

Dakle, na kraju se može zaključiti da optimalan broj namjenskih paleta iznosi 13 te se za taj broj namjenskih paleta ostvaruje najbolji rezultat, a to je proizvodnja 196 komada u jednom radnom satu.

7. ZAKLJUČAK

Iako zastupljenost korištenja računalnih simulacija u Hrvatskoj se nalazi na niskom nivou, u svijetu je implementacija istih uzela maha. Kroz svoje dobre strane unaprijedila je određene aspekte proizvodnje. Brže i jednostavnije dolazimo do rezultata ispitivanja, ubrzani su procesi nadogradnji proizvodnih sustava, itd. Općenito, ova mlada tehnologija je pouzdana, brza pri ispitivanju i sve jednostavnija za uporabu.

Kao veliki plus upravo je to što je jako mlada tehnologija koja je još u razvoju te napreduje iz dana u dan, a granice za napredovanje nema. Kao i svaka tehnologija tako i ova dolazi s određenim manama. Neke od tih mana utjecale su na samu izradu simulacijskih modela kao i kvalitetu ispitivanja. Konkretno, kroz eksperimentalni dio ovog rada to su bile: vremenski zahtjevi koje se prolongiralo, potreba za podacima koji su kasnili te se zbog istih prolongirala izrada simulacijskih modela te ograničenje validnosti simulacijskih modela koji vrijedi samo za zadane uvjete rada za koji su kreirani.

Planiranje proizvodnog procesa je dug i kompleksan proces u kojem se više strana mora dogovoriti oko zajedničkog cilja. Kako i oko najmanjih stvari iz svakodnevnog života gdje je teško sklopiti dogovor, tako je i u planiranju proizvodnog procesa. Sam prostorni raspored pogona se može mijenjati i do nekoliko puta zbog čega dolazi do kašnjenja krajnjih i točnih podataka a što utječe na prolongiranje izrade samog simulacijskog modela. Nažalost, to je faktor na koji se ne može utjecati uslijed konstantnog širenja proizvodnje čime se ne mogu osigurati potrebne informacije u najkraćem mogućem vremenskom roku. Još jedan faktor koji je imao jak utjecaj je iskustvo korištenja softvera te je bitno za napomenuti da je zadatak bio jako kompleksan. Stručnjaci u ovom području bi izradili simulacijski model i proveli ispitivanje u puno kraćem roku.

Iskustveno, kako bi se olakšalo i ubrzalo korištenje računalnih simulacija, u proizvodnji, potrebno je stvoriti lakši pristup u vidu objekata koji će imati mogućnost vršenja više mogućih akcija.

LITERATURA

[1] Ištoković, D.: „Predavanje 1“ dodatna nastavna literatura iz kolegija „Računalna simulacija proizvodnih procesa“

[2] Banks, J.: „INTRODUCTION TO SIMULATION“, s interneta, https://ieeexplore.ieee.org/document/899690, 20.09.2023

[3] Wellmann, J.; Borrelli, A.: „Computer Simulations Then and Now: an Introduction and Historical Reassessment“, s interneta, Computer Simulations Then and Now: an Introduction and Historical Reassessment | NTM Zeitschrift für Geschichte der Wissenschaften, Technik und Medizin (springer.com), 27.08.2023.

[4] Nance, R.; Wilson, R.J.; Goldsman, D.: „A brief history of simulation“, s interneta, (PDF) A brief history of simulation (researchgate.net), 29.08.2023.

[5] Lozanovski, B. i dr.:“ A Monte Carlo simulation-based approach to realistic modelling of additively manufactured lattice structures“, s interneta, https://www.sciencedirect.com/science/article/abs/pii/S2214860419313818 (ScienceDirect), 28.09.2023.

[6] Trigo, A. i dr.: „Simulation – Concepts and Applications“, s interneta, (PDF) Simulation - Concepts and Applications. (researchgate.net), 10.09.2023.

[7] Antoniades, I.P. i dr.: „Dynamical Simulation Models of the Open Source Development Process“, s interneta, https://www.researchgate.net/publication/316792667_Dynamical_Simulation_Models_of_the_Open_Source_Development_Process, 05.10.2023.

[8] Robinson, S.:“Discrete-event simulation: from the pioneers to the present, what next?“, s interneta, https://link.springer.com/article/10.1057/palgrave.jors.2601864 (SPRINGER LINK), 04.10.2023.

[9] Varga, A.: „Discrete Event Simulation System“, s interneta, https://www.sfu.ca/~ljilja/ENSC835/Spring08/News/Presentations/OMNeT++/usman.pdf (USMAN),04.10.2023.

[10] Goti, A.:“Discrete Event Simulations“, s interneta, https://books.google.hr/books?hl=hr&lr=&id=kAehDwAAQBAJ&oi=fnd&pg=PA1&dq=discrete+event+simulation+in+manufacturing&ots=cC52DLGC-T&sig=lxzmtBRB2mq6rjIr29X0DMGaA9c&redir_esc=y#v=onepage&q=discrete%20event%20simulation%20in%20manufacturing&f=false (RESEARCHGATE), 05.10.2023.

[11] Mes, R.K.M.: „Simulation Modelling using Practical Examples: A Plant Simulation Tutorial“ dodatna nastavna literatura iz kolegija „ Računalne simulacije proizvodnih procesa“

[12] Bangsow, S.: „Manufacturing Simulation with Plant Simulation and Sim Talk“, dodatna nastavna literatura iz kolegija „Računalna simulacija proizvodnih procesa“

[13] Bangsow, S.: „Tecnomatix Plant Simulation“, dodatna nastavna literatura iz kolegija „Računalna simulacija proizvodnih procesa“

POPIS SLIKA

SAŽETAK

U diplomskom radu osmišljena je automatizacija postojećeg dijela proizvodnog pogona za izradu poklopca mjenjača te je provedeno ispitivanje optimalnog broja namjenskih paleta s pomoću računalnih simulacija. Najprije je napravljen simulacijski model postojećeg dijela pogona, koji je verificiran i validiran. Nakon toga je osmišljena automatizacija dijela pogona u vidu dva prijedloga - verzija s produljenom pokretnom trakom i verzija s kolaborativnim robotom (kobotom). Izrađeni su simulacijski modeli za osnovnu varijantu kao i za dva osmišljena prijedloga unutar softvera *Tecnomatix Plant Simulation* tvrtke *Siemens*. Prilikom izrade istih naišlo se na nekoliko problema koji su uspješno riješeni. Kada su izrađeni simulacijski modeli za oba prijedloga izvršeno je ispitivanje optimalnog broja namjenskih paleta s pomoću objekta *Experiment manager*. Rezultati ispitivanja su prikazani u obliku tabličnog prikaza i točkastog dijagrama.

KLJUČNE RIJEČI: računalna simulacija, proizvodni proces, simulacija diskretnih događaja, Tecnomatix Plant Simulation, Experiment manager

SUMMARY

Automatization and testing of optimal number of pallets through computer simulations were performed in this graduate thesis. First, the simulation model was made of existing part of the plant which was after verified and validated. After that, the automatizaton of part of the plant is done conceptually and through several ideas the most favourable versions, the version with extended conveyor belt and the version with the cobot were chosen. The next step was to create a simulation model for the automatisation of part of the plant in the software „Siemens Tecnomatix Plant Simulation“. We encountered several problems that were successfully solved when models were being created. When both models were made, the optimal number of pallets was tested using an object offered by the software „Experiment manager“ itself. This facility established the optimum number of pallets for both versions of the simulation model of the automated part of the plant. The test results are presented in the form of a table and a dotted diagram.

KEY WORDS: computional simulation, manufacturing process, discrete event simulation, Tecnomatix Plant Simulation, Experiment manager