TEHNIŠKI ŠOLSKI CENTER MARIBOR
VIŠJA STROKOVNA ŠOLA
STROJNIŠTVO

Luka LORBEK

UČNE SITUACIJE ZA ŠOLSKO ROBOTSKO CELICO ABB V SIMULATORJU ROBOTSTUDIO

DIPLOMSKO DELO

Višješolski strokovni študij

Maribor, 2026

TEHNIŠKI ŠOLSKI CENTER MARIBOR
VIŠJA STROKOVNA ŠOLA
STROJNIŠTVO

Luka LORBEK

UČNE SITUACIJE ZA ŠOLSKO ROBOTSKO CELICO ABB V SIMULATORJU ROBOTSTUDIO

DIPLOMSKO DELO

Višješolski strokovni študij

LEARNING SCENARIOS FOR AN ABB EDUCATIONAL ROBOTIC CELL IN THE ROBOTSTUDIO SIMULATOR

GRADUATION THESIS

Higher vocational studies

Maribor, 2026

ZAHVALA

Iskreno se zahvaljujem mentorju Iztoku Milošiču, univ. dipl. inž. el., za strokovno vodenje, pomoč in nasvete pri izdelavi diplomskega dela. Njegova strokovnost, izkušnje in pripravljenost za pomoč so bili ključnega pomena pri uspešni izvedbi raziskave ter pri razumevanju zahtevnejših strokovnih vsebin. Posebno zahvalo namenjam svojim staršem, ki so mi ves čas nudili podporo, razumevanje in spodbudo skozi celoten študij. Njihova vztrajnost, potrpežljivost in zaupanje vame so bili nepogrešljiv del mojega uspeha. Zahvaljujem se tudi vsem profesorjem in predavateljem, ki so mi med študijem posredovali znanje ter me spodbujali k razmišljanju in samostojnemu delu. Pridobljeno znanje in izkušnje so pomembno prispevali k mojemu strokovnemu in osebnemu razvoju. Na tem mestu bi želel izreči zahvalo tudi vsem, ki so mi na kakršen koli način pomagali pri pripravi in izvedbi diplomskega dela. Vsaka oblika pomoči, spodbude ali nasveta mi je bila v veliko oporo.
Hvala vsem, ki ste pripomogli k nastanku tega dela.

IZJAVA O AVTORSTVU

Podpisani Luka Lorbek, rojen 21. 10. 2003 v Mariboru, študent Tehniškega šolskega centra Maribor, Višje strokovne šole, programa strojništvo izjavljam, da je diplomsko delo z naslovom *Učne situacije za šolsko robotsko celico ABB v simulatorju RobotStudio* avtorsko delo.

V diplomskem delu so vsi uporabljeni viri in literatura konkretno navedeni; teksti niso prepisani brez navedbe avtorjev.

Diplomsko delo je lektorirala Anka Jemenšek, prof. ang. in slov. j. , ključno dokumentacijsko informacijo sem prevedel Luka Lorbek.

Kraj in datum: _____

Lastnoročni podpis študenta/-ke: _____

MENTORSTVO

Diplomsko delo je zaključek Višješolskega strokovnega študija, smer strojništvo, opravljeno je bilo na Tehniškem šolskem centru Maribor, Višji strokovni šoli.

Študijska komisija Tehniškega šolskega centra Maribor, Višje strokovne šole je za mentorja diplomskega dela imenovala Iztoka MILOŠIČA, univ. dipl. inž. el.

Komisija za oceno in zagovor:

Predsednik: _____

Član/mentor: _____

Član: _____

Član/somentor: _____

Datum diplomskega izpita: _____

POVZETEK

V okviru diplomskega dela sem v programu RobotStudio izdelal programe za industrijskega robota ABB. Z uporabo simulacijskega okolja bom omogočil prikaz delovanja in natančno analizo gibanja robota ter preveril pravilnost in učinkovitost posameznih postopkov. Poseben poudarek bo namenjen optimizaciji gibanja in pravilni izvedbi vseh operacij znotraj virtualnega okolja. Končni rezultat diplomskega dela bodo v celoti izdelani in preizkušeni robotski programi v simulaciji, ki bodo pripravljeni za morebitno kasnejšo implementacijo na dejanskega robota.

KLJUČNA DOKUMENTACIJSKA INFORMACIJA

ŠD Dd

DK 37.02:004.43:621.865.8(043.2)

KG učne situacije, ABB RobotStudio, robotska roka ABB, programiranje, simulacija

AV LORBEK, Luka

SA MILOŠIČ, Iztok (mentor)

KZ SI-2000 Maribor, Zolajeva 12

ZA Tehniški šolski center Maribor, Višja strokovna šola

LI 2026

IN UČNE SITUACIJE ZA ŠOLSKO ROBOTSKO CELICO ABB V SIMULATORJU ROBOTSTUDIO

TD Diplomsko delo (višješolski strokovni študij)

OP XIII, 87 str., 169 sl., 12 vir.

IJ sl

JI sl/en

AI *Diplomsko delo obravnava pripravo učnih situacij za šolsko robotsko celico ABB v simulacijskem okolju RobotStudio. Namen dela je bil razviti vaje za učenje osnov programiranja industrijskega robota v virtualnem okolju, ki omogoča varno in ponovljivo delo brez uporabe dejanske opreme. V delu je opisan postopek namestitve programske opreme, aktivacije licence ter priprave simulacijske celice, sledijo pa učne situacije z jasno določenimi koraki za nastavitev robota, delovnih objektov, orodij in programiranja gibanja. Pripravljene vaje so bile preverjene v simulatorju in uspešno prenesene tudi na dejanskega robota, kar potrjuje njihovo uporabnost pri pouku in nadaljnjem izobraževanju.*

KEY WORDS DOCUMENTATION

DN Dd

DC 37.02:004.43:621.865.8(043.2)

CX learning scenarios, ABB RobotStudio, ABB robotic arm, programming, simulation

AU LORBEK, Luka

AA MILOŠIČ, Iztok (mentor)

PP SI-2000 Maribor, Zolajeva 12

PB Technical School Centre Maribor, Higher Vocational College

PY 2026

TI LEARNING SCENARIOS FOR AN ABB EDUCATIONAL ROBOTIC CELL IN THE ROBOTSTUDIO SIMULATOR

DT Graduation Thesis (Higher vocational studies)

NO XIII, 87p., 169 fig., 12 ref.

LA sl

AL sl/en

AB *This diploma thesis addresses the development of learning scenarios for an ABB educational robotic cell in the RobotStudio simulation environment. The aim of the thesis was to develop exercises for learning the basics of industrial robot programming in a virtual environment that enables safe and repeatable operation without the use of actual equipment. The thesis describes the installation of the software, license activation, and preparation of the simulation cell, followed by learning scenarios with clearly defined steps for setting up the robot, work objects, tools, and programming robot motion. The developed exercises were tested in the simulation environment and successfully transferred to a real industrial robot, which confirms their applicability in teaching and further education.*

KAZALO VSEBINE

KAZALO SLIK

1 UVOD

1.1 OPREDELITEV PROBLEMA

V okviru diplomskega dela bom obravnaval razvoj in izdelavo programov za industrijskega robota ABB GoFa CRB 15000-5/0,95 s pomočjo simulacijskega okolja RobotStudio. Glavni cilj naloge je izdelati in preizkusiti delujoče programe v virtualnem okolju, kjer je mogoče natančno analizirati gibanje, delovanje in odziv robota na posamezne ukaze. S tem bo omogočena popolna simulacija delovnega procesa brez potrebe po prenosu programov na realnega robota. Takšen pristop omogoča varno, stroškovno učinkovito in fleksibilno preizkušanje robotskih postopkov ter hkrati zmanjšuje možnost napak pri kasnejši implementaciji. Končni rezultat bo zbirka pravilno delujočih programov, ki bodo služili kot osnova za nadaljnjo uporabo ali učenje.

Ker je robot, za katerega so programi namenjeni, dostopen tudi drugim študentom višje strokovne šole in dijakom srednjih šol, bo rezultat naloge prispeval k širjenju znanja in omogočil lažje razumevanje delovanja industrijskih robotov v praksi.

1.2 NAMEN IN CILJI DIPLOMSKEGA DELA

Namen diplomskega dela je pripraviti učne situacije za ABB robotsko roko in hkrati pridobiti znanje z upravljanjem programa RobotStudio ter se seznaniti z novim načinom programiranja robota, ki ga do sedaj še ne poznam.
Cilji so:

- Modeliranje vseh potrebnih pripomočkov
- Seznaniti se z delovanjem programa RobotStudio
- Naučiti se programirati robotsko roko v drugem programskem jeziku
- Podati poročilo o uspešno opravljenem delu.

2 PREGLED STANJA

2.1 PODJETJE ABB

ABB (Asea Brown Boveri) je mednarodno tehnološko podjetje, ki ima sedež v Zürichu v Švici. Nastalo je leta 1988, ko sta se združili dve podjetji: švedska ASEA in švicarska Brown, Boveri & Cie (BBC). ABB deluje v več kot 100 državah po svetu in zaposluje več deset tisoč ljudi. Podjetje se ukvarja predvsem z elektrifikacijo, avtomatizacijo ter digitalizacijo. To pomeni, da razvija opremo in tehnologije, ki pomagajo pri prenosu in upravljanju električne energije, ter sisteme, ki podjetjem omogočajo avtomatsko in učinkovitejše delovanje. Med njihove ključne izdelke spadajo transformatorji, stikalne naprave, motorji, pogoni, industrialna avtomatika ter programska oprema za vodenje procesov. ABB je znan tudi po napredni robotiki (ABB Robotics), kjer razvija industrijske robote in rešitve za avtomatizirano proizvodnjo. Podjetje veliko vlaga v inovacije, trajnost in razvoj tehnologij, ki zmanjšujejo porabo energije ter omogočajo bolj varno in učinkovito industrijo. Glavni cilj ABB je pomagati industriji, podjetjem in skupnostim pri prehodu v bolj pametno, digitalno in okolju prijazno prihodnost (ABB, 2025).

Slika 1: Logotip podjetja ABB

Vir: (ABB, 2025)

2.2 ROBOTSKE ROKE ABB

ABB Robotics je skozi zgodovino dosegel pomembne prodajne mejnike, kar odraža njegovo vodilno vlogo na trgu industrijskih robotov. Že leta 2002 je ABB poročal o prodaji 100.000 robotov, kar je bil velik mejnik in znak, da imajo njihove rešitve široko bazo uporabnikov. Kasneje, po skoraj štirih desetletjih delovanja, je ABB dosegel prodajo že 250.000 robotskih rok, kar potrjuje dolgoletno rast in pomembnost njihovega robotskega segmenta. ABB ima v svojem portfelju različne vrste robotskih rok, ki se uporabljajo v tovarnah in podjetjih po vsem svetu. Te robotske roke pomagajo pri težkih, ponavljajočih ali natančnih opravilih, kot so premikanje materiala, pakiranje, varjenje, montaža ali barvanje. Podjetje ponuja tako klasične industrijske robote kot tudi kolaborativne robote, ki lahko varno sodelujejo z ljudmi (ABB, 2025).

Eden najbolj zanimivih kolaborativnih robotov je ABB GoFa CRB 15000-5/0,95 na katerem je temeljilo moje diplomsko delo. Je šest osni industrijski robot, namenjen varnemu sodelovanju z operaterji v skupnem delovnem prostoru. Robot ima nosilnost 5 kg in doseg približno 0,95 m, kar omogoča uporabo pri manipulacijskih, montažnih in strežnih nalogah v proizvodnih sistemih. Robot je opremljen s senzorji sile in navora v vseh sklepih, kar omogoča zaznavanje stika in varno zaustavitev gibanja. V kombinaciji z naprednimi varnostnimi funkcijami krmilnega sistema omogoča kolaborativno delovanje brez uporabe zaščitnih ograj, če to dopušča ocena tveganja. Zaobljena mehanska zasnova dodatno prispeva k varnosti pri delu. Krmiljenje robota temelji na platformi ABB OmniCore, ki omogoča več načinov programiranja, vključno z ročnim vodenjem, programiranjem prek upravljavskega terminala ter uporabo programske opreme RobotStudio za simulacijo in offline programiranje. Robot odlikujeta dobra ponovljivost in relativno visoka hitrost gibanja, kar omogoča učinkovito izvajanje cikličnih procesov. Zaradi svoje kompaktne zasnove, prilagodljive montaže in enostavne integracije je ABB GoFa CRB 15000-5/0,95 primeren za aplikacije, kot so strežba strojev, manipulacija materiala, pobiranje in odlaganje izdelkov ter enostavna montaža (ABB, 2025).

Slika 2: GoFa CRB 15000-5/0,95

Vir: (ABB, 2025)

Poleg CRB 15000-5/0,95 ABB izdeluje tudi klasične industrijske robote, kot sta IRB 2600, ki je močna in hitra roka za težja dela, ter IRB 1200, ki je majhna, hitra in zelo natančna roka za montažo in precizna opravila. CRB 15000-5/0,95 je poseben, ker združuje prednosti kolaborativnega in industrijskega robota. Lahko sodeluje z ljudmi, hkrati pa izvaja delo hitro, natančno in učinkovito, zaradi česar je primeren za sodobne tovarne in proizvodne linije (ABB, 2025).

Slika 3: IRB 2600

Vir: (ABB, 2025)

2.3 ROBOTSTUDIO

RobotStudio je programska oprema, ki jo je razvilo podjetje ABB in je namenjena programiranju, simulaciji in optimizaciji industrijskih robotov. Glavna prednost tega programa je, da omogoča načrtovanje in testiranje robotov na računalniku, še preden jih namestimo v dejansko proizvodnjo. To pomeni, da lahko podjetja preizkusijo različne scenarije, optimizirajo gibanje robota in preverijo, ali bo robot opravil nalogo pravilno, brez tveganja za poškodbe ali napake v proizvodnji. Program temelji na 3D simulaciji. Uporabnik lahko v računalniškem okolju zgradi robotsko celico, postavi robota, delovno območje in predmete, s katerimi bo robot delal. Nato z enostavnimi ukazi ali grafičnim vmesnikom nastavi pot, hitrost, kotne gibe in naloge robota. RobotStudio uporablja enako programsko logiko in ukaze kot dejanski robot, kar pomeni, da se program, ki ga izdelamo v simulaciji, lahko prenese neposredno na fizičnega robota. Program deluje na principu “offline programiranja”, kar pomeni, da ni potrebno prekiniti delovne linije ali izklopiti drugih strojev. Program lahko predvidi morebitne trke, preveri čas izvajanja naloge in optimizira gibanje robota za večjo hitrost ali natančnost. Poleg tega RobotStudio omogoča tudi programiranje sodelujočih robotov, kot je GoFa, kjer je mogoče določiti meje sile in varnostne funkcije, da robot varno sodeluje z ljudmi. Uporaba RobotStudia poteka približno tako: najprej se ustvari 3D model robota in njegovega delovnega okolja, nato se določijo naloge in poti gibanja, simulira se delovanje, preveri optimizacija in varnost, na koncu pa se pripravljeni program prenese na robota. To podjetjem omogoča, da hitro testirajo različne nastavitve in naloge, zmanjšajo napake, povečajo natančnost in varnost ter prihranijo čas in stroške pri uvajanju novih robotskih rešitev (ABB, 2025).

3 PRIPRAVA UČNIH SITUACIJ

Učne situacije so vnaprej pripravljeni učni primeri, pri katerih se udeleženci učijo z reševanjem nalog v realnih ali simuliranih okoliščinah. Njihov namen je povezati teoretično znanje s praktičnim delom ter omogočiti boljše razumevanje učne snovi (Sigh, 2025).

V tem poglavju so predstavljene učne situacije, ki so bile pripravljene in izvedene v simulacijskem okolju ABB RobotStudio, kjer je bil uporabljen virtualni industrijski robot. Pred izvedbo vaj je opisan postopek namestitve programskega okolja, aktivacije programske licence ter način pridobitve licence na osebni računalnik, kar je omogočilo nemoteno delo v simulatorju. Pred samim začetkom dela v simuliranjem okolju sem iz strani mentorja prejel datoteko, v kateri so bila osnovana navodila za učenje, namenjeno začetnikom. Podrobno sem pregledal vsebino te datoteke, saj mi je ta bila v pomoč pri nadaljnjem delu (ABB, 2025).

Poudarek praktičnega dela je na simuliranem delu z virtualnim robotom, pri čemer so bile vse naloge izvedene s pomočjo programa RobotStudio. Za vsako učno situacijo je razložen njen namen ter predstavljeni osnovni koraki za pripravo simulacijskega okolja, kot so nastavitev virtualnega robota, delovnega objekta, orodja in referenčnih točk. Za vsako učno situacijo so pripravljena jasna in zaporedna navodila, ki omogočajo samostojno izvedbo vaj v simulacijskem okolju.

3.1 NAMESTITEV PROGRAMA ROBOTSTUDIO IN PRIDOBITEV LICENCE

Za začetek programiranja sem najprej potreboval program RobotStudio, zato sem ga prenesel in namestil na računalnik. V šoli sem na glavnem računalniku aktiviral uradno enoletno licenco, ki je omogočala uporabo vseh funkcij programa. Na svoj osebni računalnik pa sem namestil enako različico RobotStudia, vendar sem imel na voljo začasno 90-dnevno licenco, kar je bilo dovolj za potrebe učenja in izvajanja projekta. Pri namestitvi in prvih korakih sem si pomagal s priročnikom, ki sem ga prejel v šoli pri mentorju. Priročnik je vseboval osnovna navodila za namestitev in začetno uporabo programa, zato sem ga najprej temeljito prebral, da sem bolje razumel postopek in se izognil morebitnim napakam. Ko sem se seznanil z navodili, sem brez večjih težav uspešno namestil program in ga pripravil za nadaljnje delo. S tem sem lahko začel z učenjem programiranja robotov in s samim postopkom izdelave simulacij v RobotStudiu (Jerman, 2024).

3.1.2 Navodila za namestitev programa

Postopek namestitve in aktivacije simulacijske programske opreme ABB RobotStudio na lokalni server in uporabniške PC-je.

1. RobotStudio za izobraževalne ustanove deluje tako, da SLP Distributor deli 100 licenc med uporabniške PC-je. SLP Distributor je nameščen na lokalnem serverju, do katerega morajo imeti dostop uporabniški PC-ji. Postopek namestitve je naveden spodaj in v nadaljevanju opisan po posameznih korakih:
   • posredovanje navodil za namestitev ter aktivacijske kode,
   • prenos datotek potrebnih za namestite RS z ABB-jeve spletne strani,
   • namestitev SLP Distributorja na server,
   • enkratni vnos aktivacijske kode v SLP Distributor,
   • namestitev programske opreme RS na posamezne uporabniške PC-je,
   • navedba IP naslova SLP Distributorja na posameznih uporabniških PC-jih,
   • potrditev uspešne namestitve RS-ja ABB-ju.

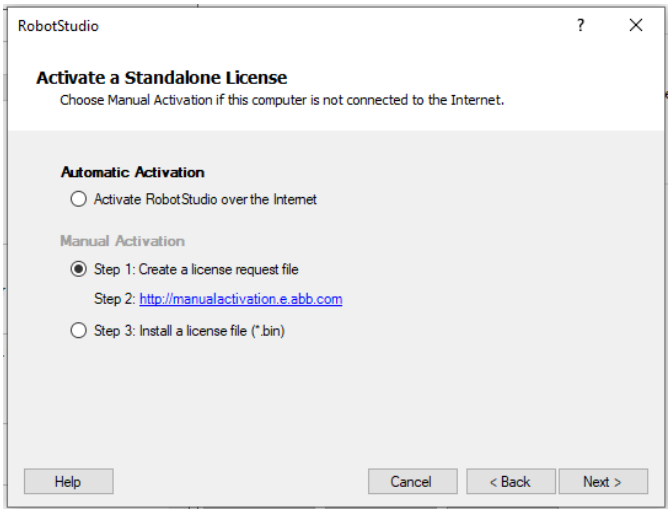
Aktivacijska koda se vpisuje le v SLP Distributor, nikoli na uporabniške PC-je!

2. Za prenos datotek potrebnih za namestitev RS kliknite na spodnjo povezavo. **https://new.abb.com/products/robotics/robotstudio/downloads**
   Nato İzberite ikono za prenos, izpolnite svoje podatke ter kliknite na »Downloads«.
   V datoteki, ki jo boste prenesli »RobotStudio 2024.11.zip«, se nahaja več map.
3. Na lokalni server sedaj namestite SLP Distributor, ki ga najdete v istoimenski mapi (SLP Distributor).
   Zaženite datoteko »SLP Distributor for ABB.exe« in dokončajte namestitev.
4. Vnos aktivacijske kode, ki je za vsako šolo unikatna, se izvedete samo na serverju v SLP Distributor. Podrobna navodila so poglavju »Activating Multi-user license« na 25 strani dokumenta. Aktivacijsko kodo ste prejeli skupaj s navodili v e-mail-u.
5. Sedaj namestite RobotStudio tudi na enega izmed uporabniških računalnikov, na katerih boste izvajali pouk. V preneseni mapi RobotStudio izberite datoteko »Setup.exe« in jo zaženete. Predlagamo namestitev RS-ja v polni velikosti (opcija complete).
6. Ko je programska oprema RS nameščena na uporabniškem PC-ju, jo je potrebno povezati preko IP naslova s SLP Distributorjem. Ko bo RS prvič zagnan, se bo odprlo okno »Licensing«, ki je sicer vedno na voljo v meniju »Options«. Izberete »Activate RobotStudio license«, »I want to specify a network license server or manage server licenses« ter nato navedite IP naslov do license serverja. Po novem zagonu RS-ja bi morala licenca delovati. Enako kadar koli ponovite za vse ostale uporabniške PC-je.
   V kolikor želite profesorju ali dijaku licenco za največ 90 dni prenesti na prenosnik (za samostojno uporabo brez povezave do SLP Distributorja) na izbranem uporabniškem PC-ju izberite opcijo »I want to check out commuter license.« (Jerman, 2024).

3.1.1 Aktivacija Licence

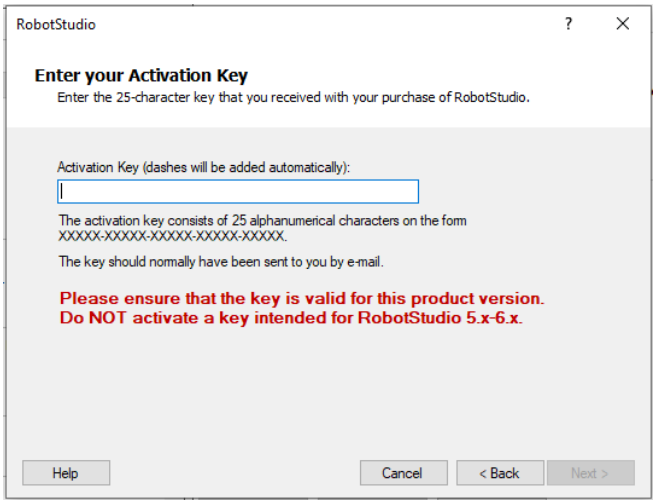
Da sem lahko aktiviral licenco, sem na šolskem računalniku moral v oknu Standalone license izbrati možnost Manual activation. Ko sem potrdil izbiro s klikom na Next, se je odprlo okno Enter your activation key, kamor sem moral vnesti 25-mestno aktivacijsko kodo. To kodo sem prejel od svojega mentorja. Po vnosu kode je program ustvaril licenčno datoteko, ki sem jo moral shraniti. Nato je bilo potrebno izvesti še drugi korak postopka ročne aktivacije (Manual activation – Step 2), kjer sem moral to datoteko ponovno naložiti v program, da se je licenca pravilno registrirala. Ko sem oddal licenčno datoteko in program potrdil njeno veljavnost, sem lahko licenco uspešno namestil na računalnik. S tem je bila aktivacija zaključena in RobotStudio je postal pripravljen za uporabo.

Slika 4: Aktiviranje licence



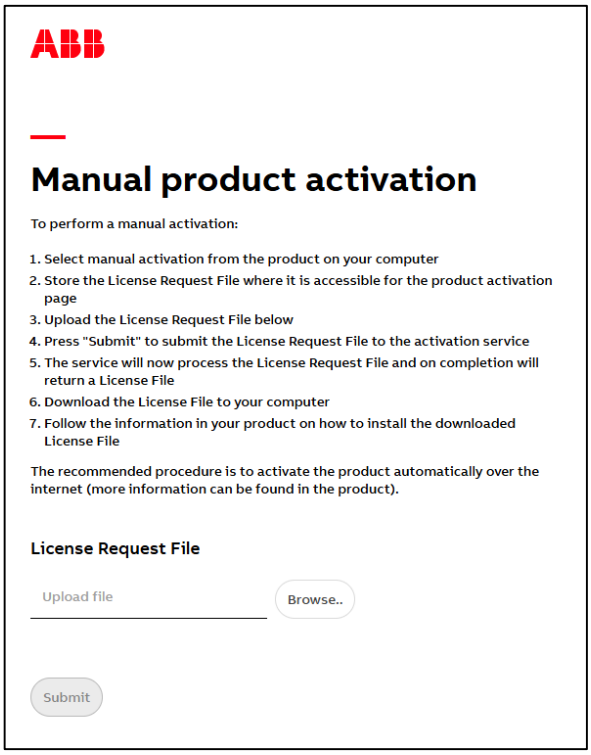
Vir: (ABB, 2025)

Slika 5: Vnos 25-mestne aktivacijske kode



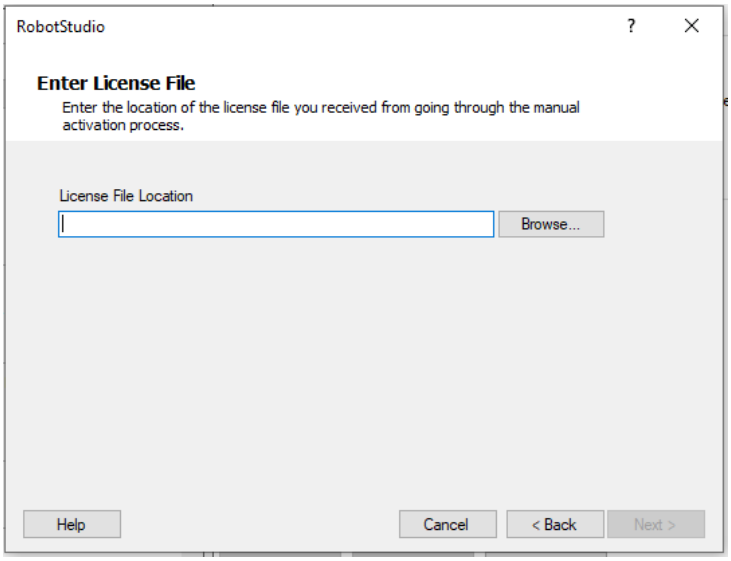
Vir: (ABB, 2025)

Slika 6: Drugi korak za aktivacijo licence



Vir: (ABB, 2025)

Slika 7: Namestitev licence v računalnik

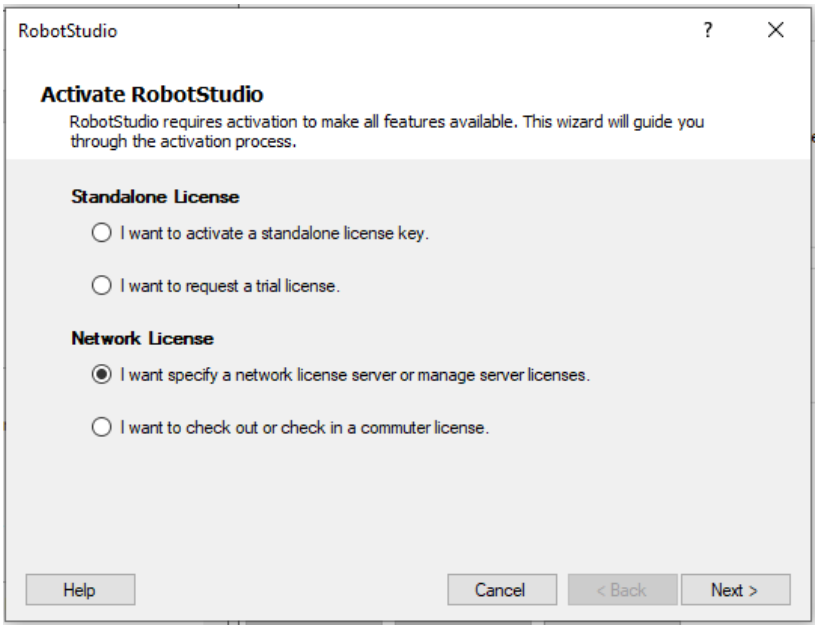


Vir: (ABB, 2025)

Ko sem program namestil, sem v RobotStudiu odprl meni Options in nato zavihek Licensing, kjer se urejajo vse nastavitve povezane z licencami. V delu Active RobotStudio License sem izbral možnost I want to specify a network license server or manage server licenses. Tam sem moral ročno vnesti IP-naslov šolskega računalnika, ki je gostil licenco.

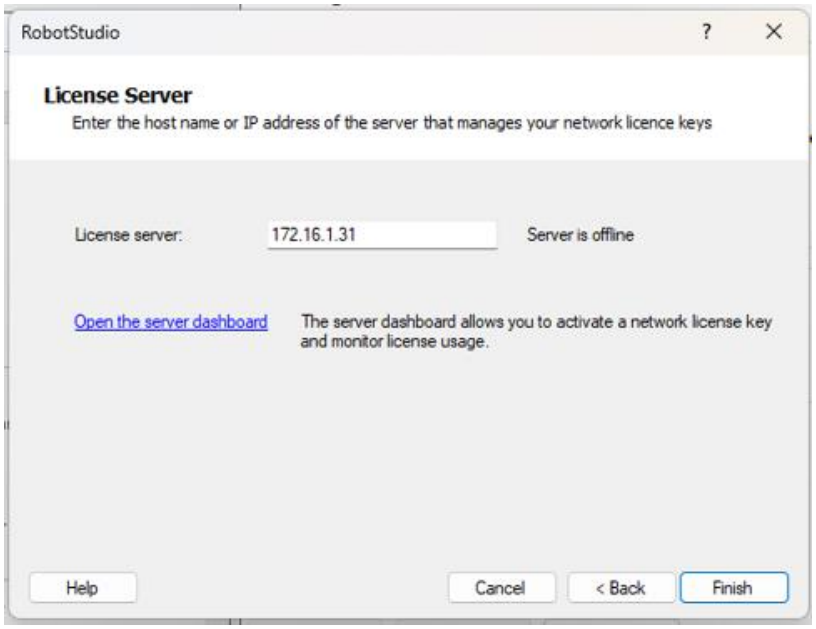
Po potrditvi povezave je moj računalnik samodejno zaznal veljavno licenco in program se je aktiviral. Ker ima računalnik notranji IP naslov, se lahko na njega povežeš lokalno, preko Ethernet kabla, ali pa od zunaj preko VPN povezave.

Slika 8: Povezava z licenco



Vir: (ABB, 2025)

Slika 9: IP naslov šolskega računalnika

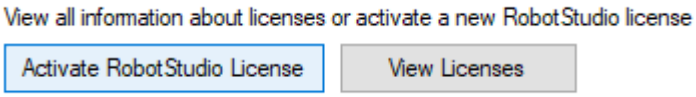


Vir: (ABB, 2025)

3.1.2 Aktivacija licence na osebnem računalniku

Na spletni strani ABB sem si najprej prenesel najnovejšo razpoložljivo različico programa RobotStudio. Ker sem na šolskem računalniku že predhodno aktiviral uradno enoletno licenco, sem lahko to licenco uporabil tudi na svojem računalniku. Šolski računalnik je tako deloval kot licenčni strežnik, kar je omogočalo, da se lahko drugi računalniki v šoli ali doma povežejo nanj in pridobijo dostop do licence.
Ko sem program namestil, sem v RobotStudiu odprl meni Options in nato zavihek Licensing, ter izbral možnost za aktiviranje licence.

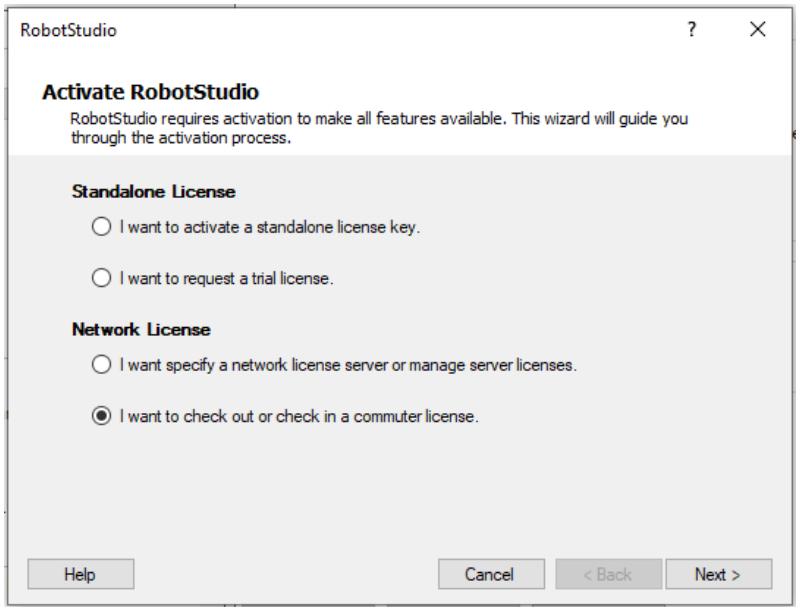
Slika 10: Aktiviranje licence



Vir: (ABB, 2025)

V pojavnem oknu sem izbral možnost I want to check out or check in a commuter license. Ta možnost omogoča, da si uporabnik začasno izposodi licenco iz licenčnega strežnika in jo uporablja na računalniku tudi takrat, ko ni povezan v omrežje šole.

Slika 11: Povezava s strežnikom

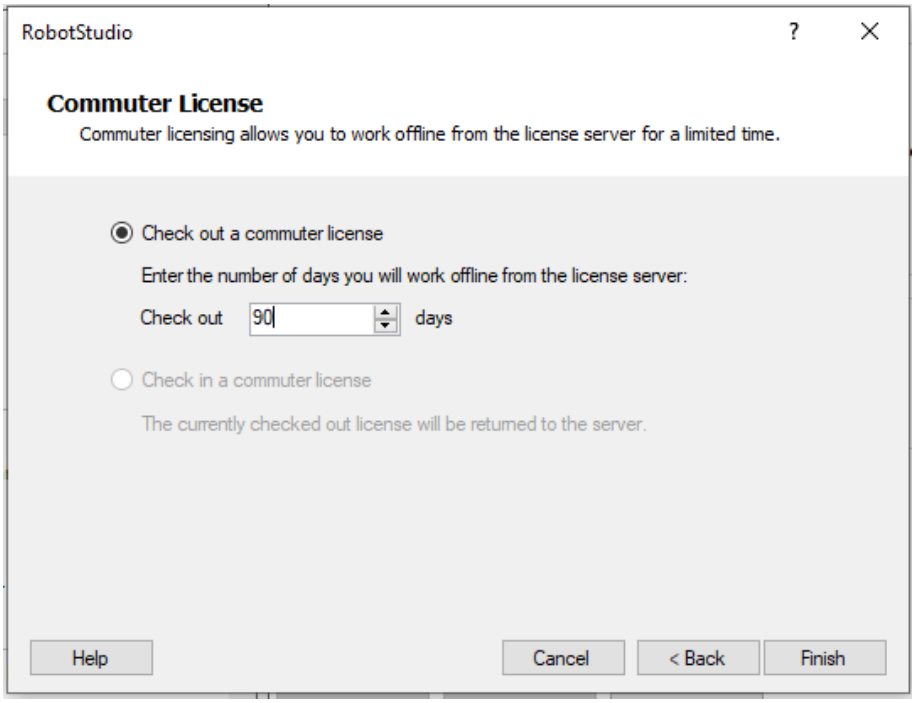


Vir: (ABB, 2025)

S tem postopkom sem si zagotovil popoln dostop do vseh funkcij RobotStudia, kar je bilo ključno za nadaljnje programiranje robota in izvajanje simulacij. Prednost tega pristopa je tudi ta, da lahko licenco uporablja več uporabnikov, dokler so povezani v isto omrežje ali imajo dostop do strežnika. To je zelo praktično, saj omogoča učenje in delo na daljavo, brez potrebe

po več ločenih licencah.

Slika 12: Določitev veljavnosti začasne licence



Vir: (ABB, 2025)

Na prikazani sliki je videti okno, v katerem sem lahko izbral, za koliko dni si želim izposoditi licenco s šolskega strežnika. Najdaljše dovoljeno obdobje je 90 dni, zato sem v ustrezno polje vpisal 90. S tem sem si zagotovil, da lahko program uporabljam tudi doma, brez stalne povezave s šolskim računalnikom, ki deluje kot licenčni strežnik.

Ko sem potrdil nastavitev s klikom na Finish, se je program ponovno naložil in na začetni strani sem lahko videl točen datum poteka moje licence. Ta pregled je zelo uporaben, saj omogoča, da uporabnik pravočasno preveri veljavnost licence in si po potrebi ponovno izposodi licenco za dodatno obdobje.

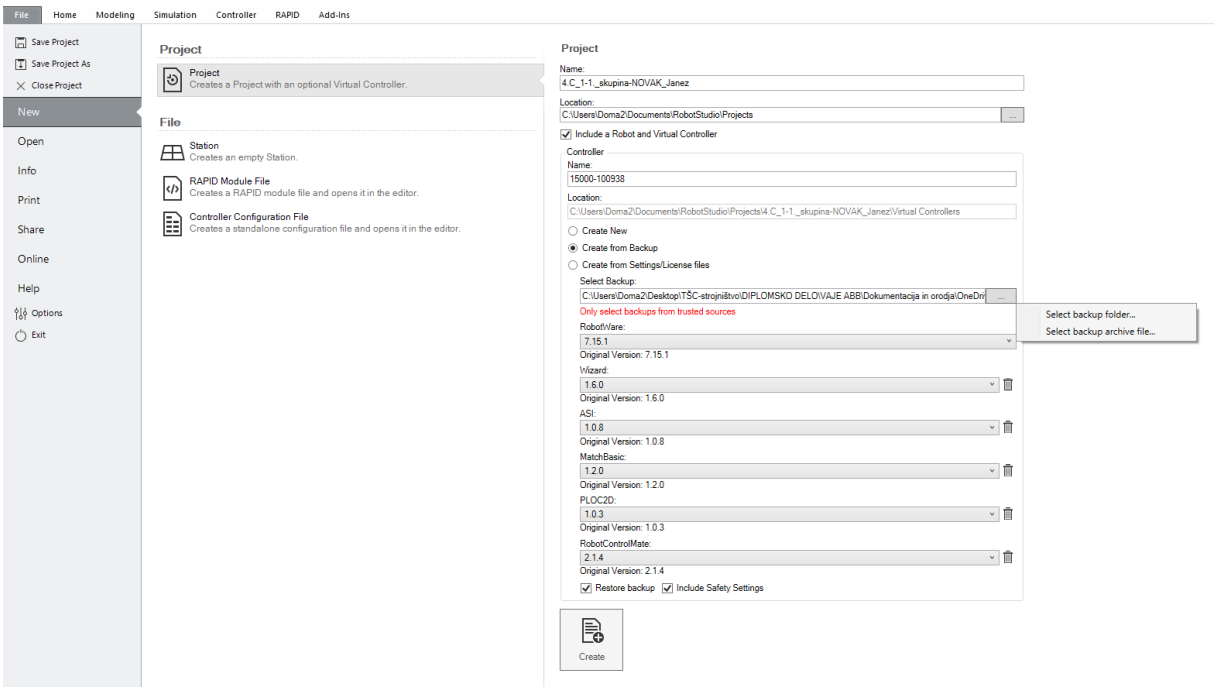
3.2 VAJA 1

Izdelava robotske celice v simulacijskem programu RobotStudio

Izdelajte robotsko celico in dodajte vse potrebne komponente (podstavek robota, prijemalo s kamero, delovno ploščo in oviro). Spoznajte se z osnovami programiranja v RobotStudi ter osnovnim delom v programu.

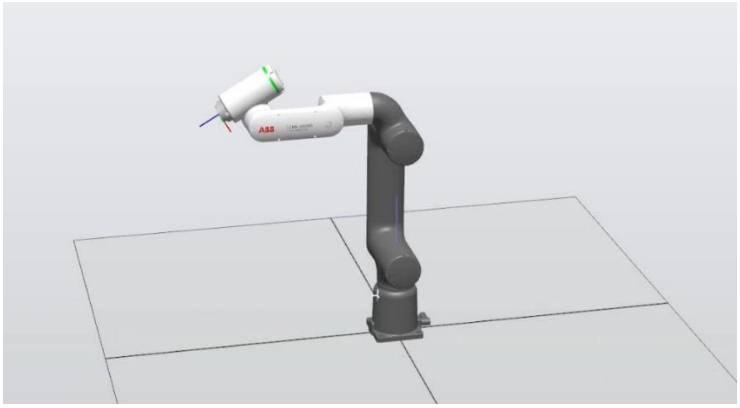
1. Zaženete program RobotStudio.
2. V meniju File izberemo okno New in nato Project.
3. Vpišete ime RC. Ime vpišete po sistemu: **razred-skupina-PRIIMEK_Ime**. Dovoljeni znaki so črke abecede (brez šumnikov, preglasov ...), števila, pika (.) in podčrtaj (_) ter vezaj (-). Vse ostalo ni dovoljeno (presledki, šumniki, preglasi, posebni znaki …), ker lahko pride do napak v RC.
   Primer: **4._C_1-1._skupina-NOVAK_Janez**
   Controller izberete iz backup-a pod izbiro Select backup archive file. S tem izberemo varnostno kopijo krmilnika dejanskega robota. Pri vseh dodatkih robotske celice izberemo najvišjo verzijo (če nam nobene ne ponudi na izbiro, jih je potrebno inštalirati v zavihku Add-ins). Ko imamo vse izbrano, potrdimo s Create.

Slika 13: Ustvarjanje projekta



4. Ko ustvarimo projekt, ustvarimo tudi virtualni krmilnik, katerega status se prikaže v desnem spodnjem robu. Počakajte, da se vam pojavi RC.

Slika 14: Ustvarjena robotska celica



Delo z miško

1. Pomoč za delo z miško glejte v razdelku Mouse Commands.
2. Za približanje in pomanjšanje uporabite koleščck.
3. Za vrtenje pogleda na robotsko celico uporabite CTRL + SHIFT + levi klik.
4. Za premikanje po zaslonu uporabite CTRL + levi klik.

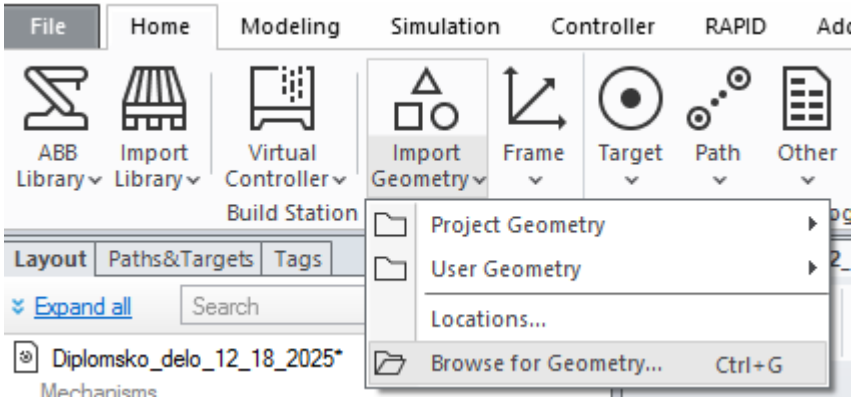
Slika 15: Ukazi za miško

Dodajanje podstavka

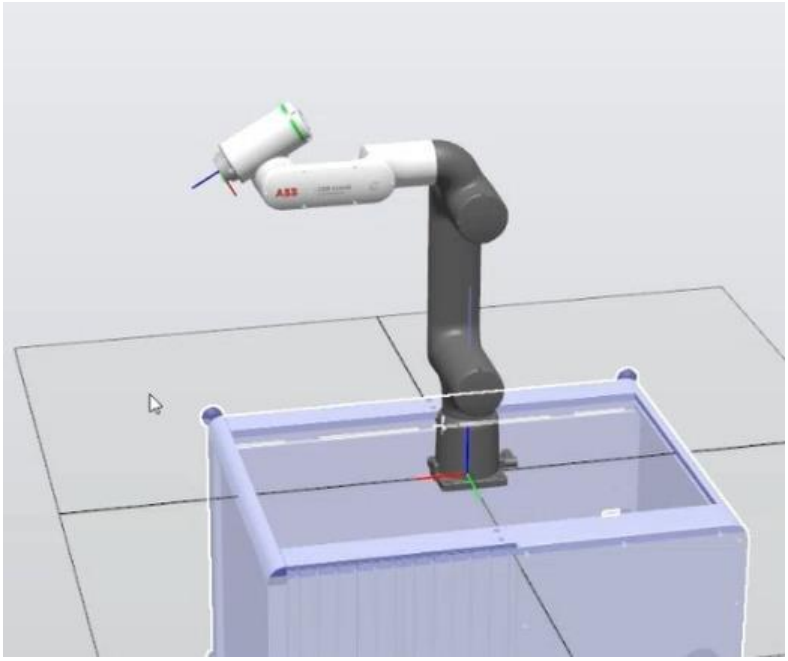
Predpogoj za začetek te vaje je, da imate izdelano osnovno RC z ustreznim robotom.

1. Kliknite Import Geometry in nato Browse for Geometry, izberite Podstavek_robota.SAT ter ga uvozite.

Slika 16: Uvoz podstavka



Slika 17: Uvožen podstavek

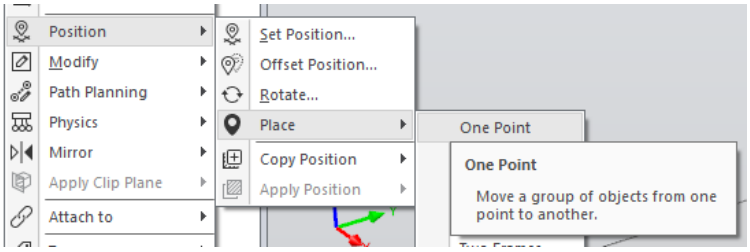


2. Izberite podstavek in v zavihku Home funkcijo Move and Rotate. S tem lahko podstavek premaknete v poljuben položaj in rotacijo. S puščico, premikamo komponento linearno, s krogom pa spreminjamo rotacijo.
3. V pomožnem orodju izberemo funkcijo Snap Object.

Slika 18: Funkcija snap

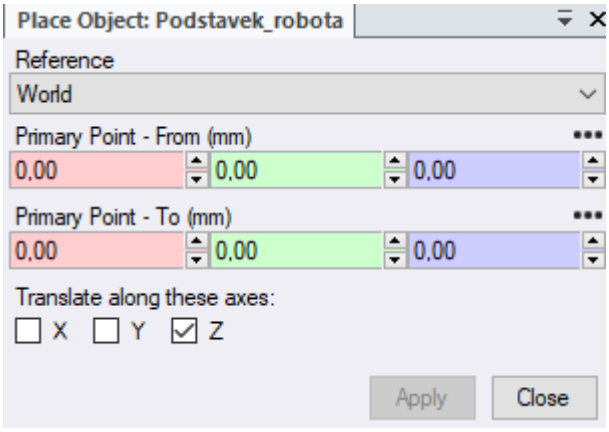
4. V layout-u izberemo podstavek, position, place, one point.

Slika 19: Pozicioniranje komponent

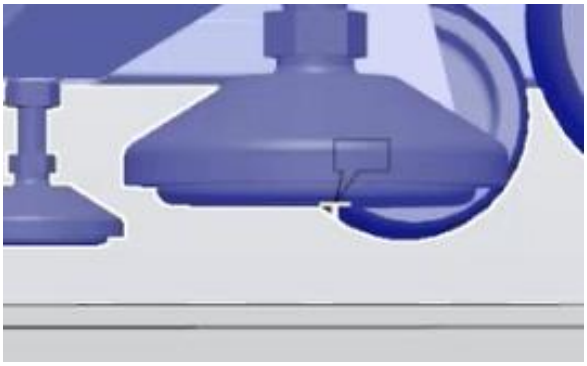


5. V pojavljenem oknu označimo vrednosti od trenutne pozicije točke in do želene. Ker želimo nastaviti samo višino, pustimo obkljukano samo Z os. Izhodiščno točko označimo tako, da bo na spodnjem delu ene izmed nog mize.
   Ker želimo da bo podstavek postavljen na površini RC, pustimo želeno točko na 0 in potrdimo z Apply. Podstavek se premakne na nivo tal.

Slika 20: Postavljanje komponent



Slika 21: Izhodiščna točka podstavka



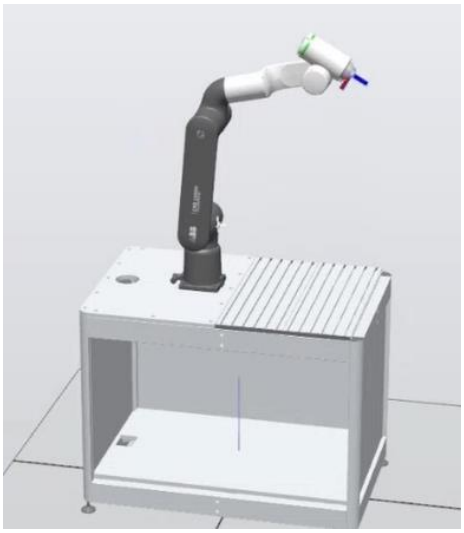
Pozicioniranje robotske roke

1. V zavihku layout izberite robotsko roko in jo z uporabo funkcije Move and Rotate premaknite nad podstavek robota.
2. V pomožnem orodju, izberemo funkcijo za Snap Center (desno od Snap Object).
3. V layout-u izberemo robotsko roko, position, place, one point.
4. Izberemo center ene izmed lukenj na nosilcu robota kot izhodiščno točko in eno izmed lukenj na podstavku kot želeno točko.
5. Ob tem se pojavi okno, ki sprašuje, če želimo premakniti Task Frame in potrdimo z DA.

Slika 22: Pozicija robotske roke

Končna postavitev mora biti enaka (slika 23).

Slika 23: Robotska roka na podstavku

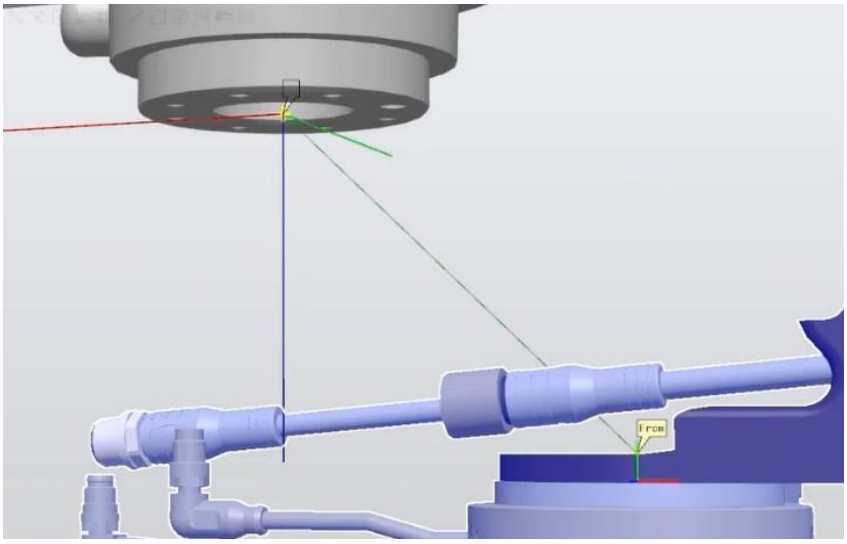


Namestitev prijemala s kamero

1. Kliknite Import Geometry in nato Browse for Geometry, izberite Prijemalo_s_kamero.SAT ter ga uvozite.
2. Na robotski roki 5. os nastavite pravokotno. To storite tako, da v zavihku Layout ob kliku na robotsko roko izberete Mechanism Joint Jog. Vse osi postavite na 0°**,** razen 5. os na 90°**.**
3. Z ukazom Move and Rotate premaknite prijemalo s kamero bližje zadnji osi robota.
4. Postavite se na prijemalo in v zavihku Layout izberemo Podstavek → Position → Place → One point.
5. S funkcijo Snap Center postavimo prijemalo na zadnjo os robotske roke. Premaknemo jo po vseh treh oseh.
6. Z desnim klikom na prijemalo v zavihku Layout izberemo Attach to in nato CRB15000_5_95__02(TROB1)/Link6. S tem zaklenemo pozicijo prijemala na robotski roki.

7. Vpraša nas, če želimo osvežiti pozicijo in izberemo NE, ker smo pozicijo ročno nastavili.

Slika 24: Pozicioniranje prijemala

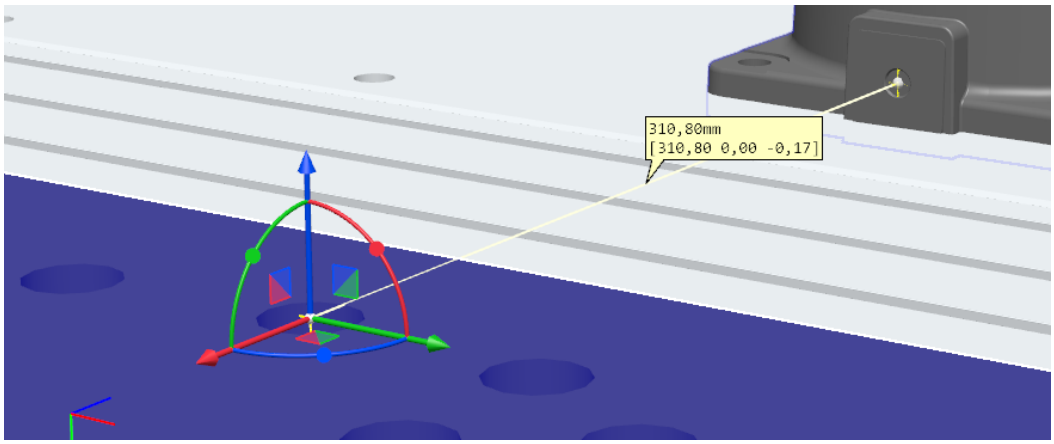


Postavitev delovne plošče

1. Z ukazom Import Geometry uvozite Delovna_plošča.SAT.
2. Z ukazom Move and Rotate jo postavite nad podstavek robota.
3. Z ukazom Snap Object in Position, Place, One point jo odložite na podstavek.
4. Postavite jo tako, da bo sredina robota poravnana s sredino plošče (os Y) in da bo po osi X odmaknjena za 310.
   S funkcijama Snap Center in Measure Point to Point lahko izmerite razdaljo točk.
   Upoštevajte točke za merjenje razdalje (Slika *26*).
5. Nato še pozicijo plošče zaklenemo na podstavek robota z ukazom Attach to.

Slika 25: Orodje Snap in Measure

Slika 26: Točke za merjenje pozicije

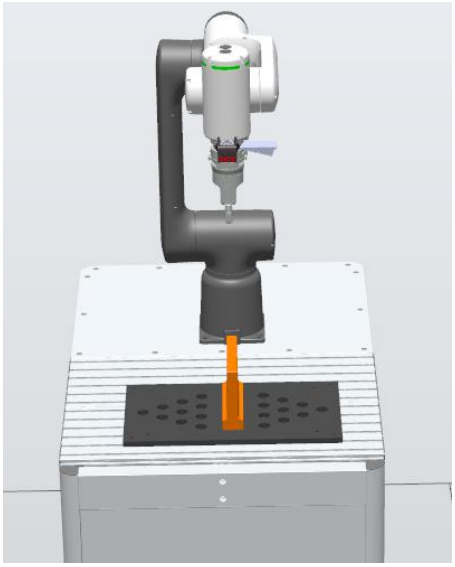


Postavitev ovire

1. Z ukazom Import Geometry uvozite Ovira.SAT.
2. Z ukazom Move and Rotate jo postavite nad delovno ploščo.
3. Z ukazom Snap Object/Center in Position, Place, One point jo odložite v sredinske luknje plošče (Slika *27*).
4. Z ukazom Attach to se zaklene pozicija ovire na delovni plošči.
5. Z desnim klikom na oviro in ukazom Modify lahko po želji spreminjamo barve komponentam.

Po tem postopku se uvažajo in postavljajo vse komponente, ki jih boste potrebovali pri vajah.

Slika 27: Pripravljena robotska celica

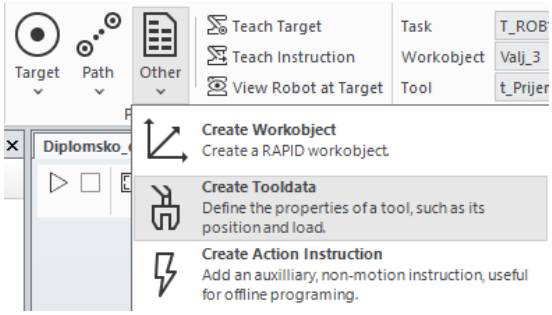


Ustvarjanje Tooldata

Tooldata lahko nastavimo na dva načina:

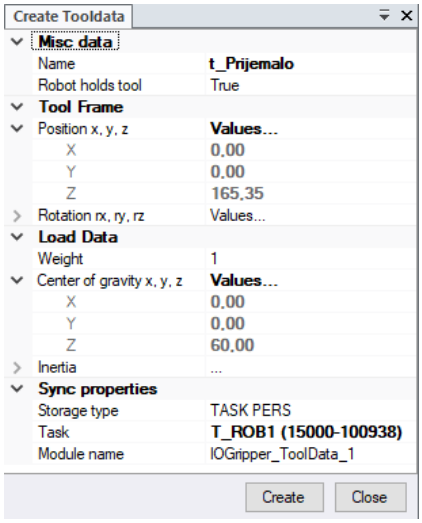
1. Z vpisovanjem pozicije:
   - V zavihku Home izberemo Other in nato Create Tooldata.

Slika 28: Create Tooldata



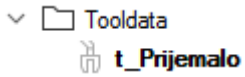
- V oknu za kreiranje orodja, vnesemo ime orodja, pozicijo po osi Z, ki znaša 165,35 mm in maso po osi Z za 60 kg.

Slika 29: Podatki za Tooldata



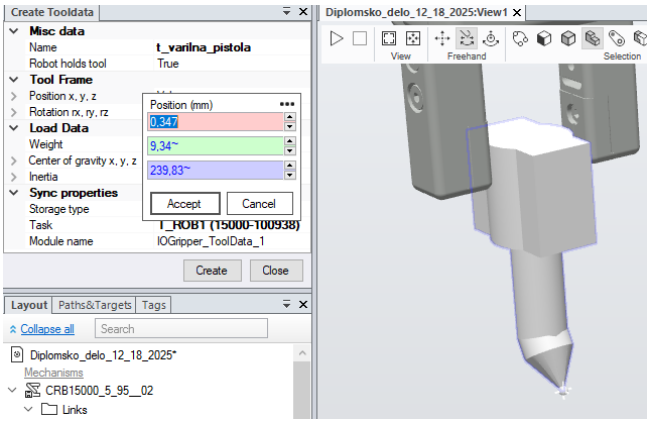
- V zavihku Paths&Targets lahko vidite ustvarjen Tooldata.

Slika 30: Ustvarjen Tooldata



2. S klikom na točko:
   - V zavihku Home izberemo Other in nato Create Tooldata.
   - V oknu za kreiranje orodja se postavimo na rdeče polje in z ukazom Snap Object izberemo konico orodja.

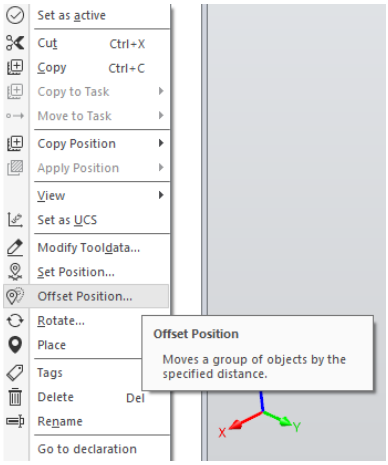
Slika 31: Nastavitev Tooldata s klikom na točko



- Ko smo označili pozicijo, še vnesemo podatek za maso po osi Z (60 kg) in ustvarimo Tooldata.

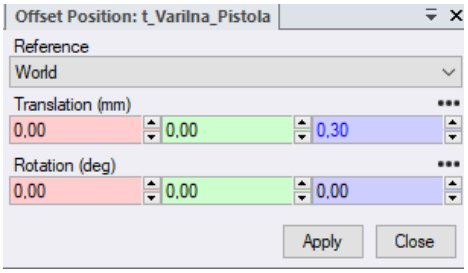
- Nato izberemo ta ustvarjen Tooldata in izberemo Offset Position.

Slika 32: Offset Tooldata



- Offset po osi Z nastavimo na 0,3 mm, da preprečimo stik orodja z obdelovancem in zagotovimo ustrezen razmik med njima.

Slika 33: Zamik TCP

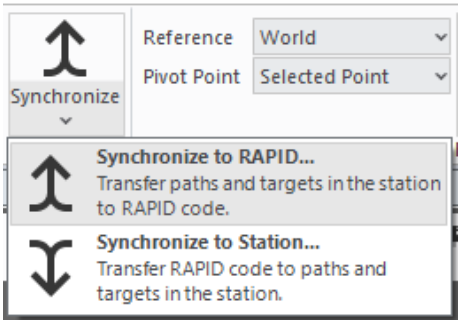


Sinhroniziranje podatkov

Če želimo, da so podatki v celici enaki kot so v Rapidu – del kjer se pišejo programi, jih moramo sinhronizirati oziroma pošiljati med enim in drugim.

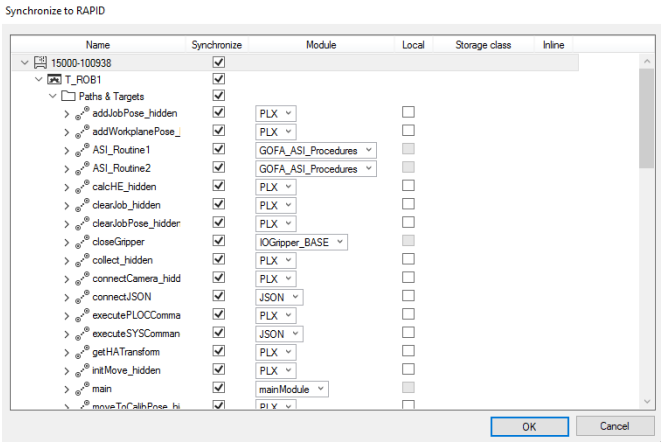
Če želimo podatke, kot so točke, Tooldata, WorkObject zabeležiti v Rapidu moramo:
1. Pritisniti ukaz Synchronize.
2. Nato izbrati Synchronize to RAPID.

Slika 34: Sinhronizacija v Rapid



3. Odpre se okno, v katerem mora biti obkljukan prvi kvadrat.
4. Potrdite z OK. Robotska celica se je prenesla v Rapid.

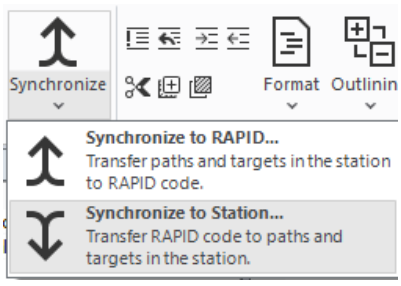
Slika 35: Sinhronizacija v Rapid



Če želimo programske kode in spremembe v teh imeti v robotski celici, moramo:

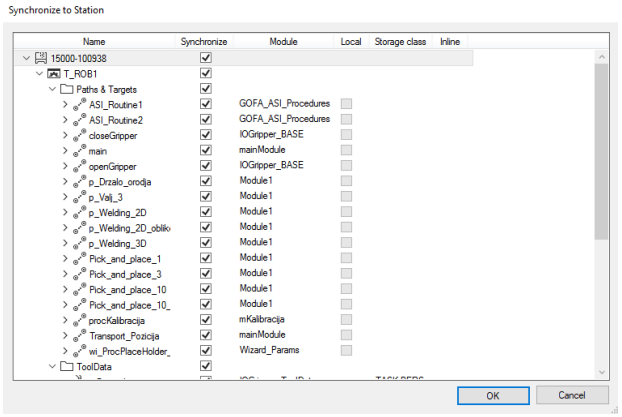
1. V zavihku Rapid pritisniti Synchronize.
2. Nato izbrati Synchronize to Station.

Slika 36: Sinhronizacija v robotsko celico



3. Odpre se okno, v katerem obkljukajte prvi kvadrat.
4. Potrdite z OK. Rapid se je prenesel v robotsko celico.

Slika 37: Sinhronizacija v robotsko celico



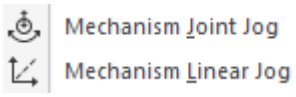
5. V kolikor se v Rapidu ali v sami robotski celici kaj popravlja ali spreminja in se po tem ne sinhronizira, se popravljen ukaz ne bo shranil. Sinhronizira se lahko vsaka stvar posebej ali pa zgolj določene, ki si jih obkljukamo sami.

Vodenje robotske roke

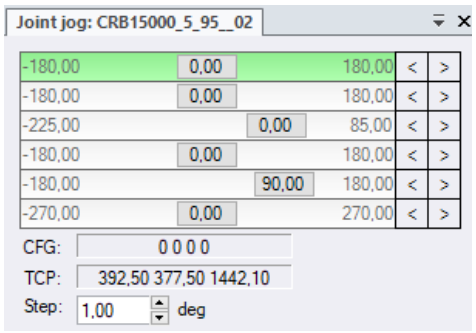
Robotsko roko lahko vodimo na več načinov:

1. Z desnim klikom na robota lahko robotu določimo pozicijo kartezičnih ali osnih gibov. To storite tako, da enega izberete in vnesete vrednosti. Robotska roka se bo premaknila v željen položaj.

Slika 38: Premik robotske roke

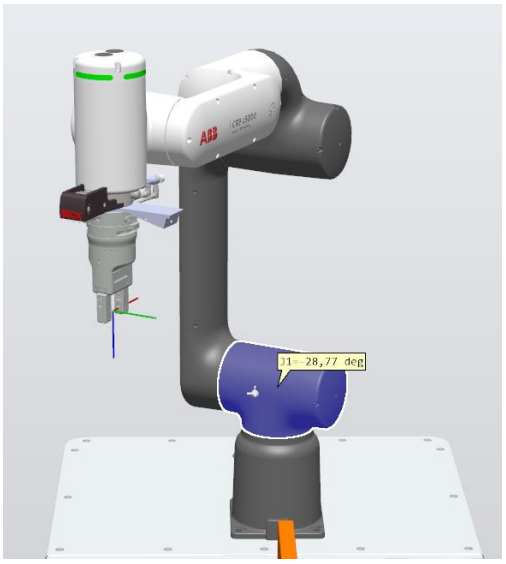


Slika 39: Premik robotske roke z osnimi gibi



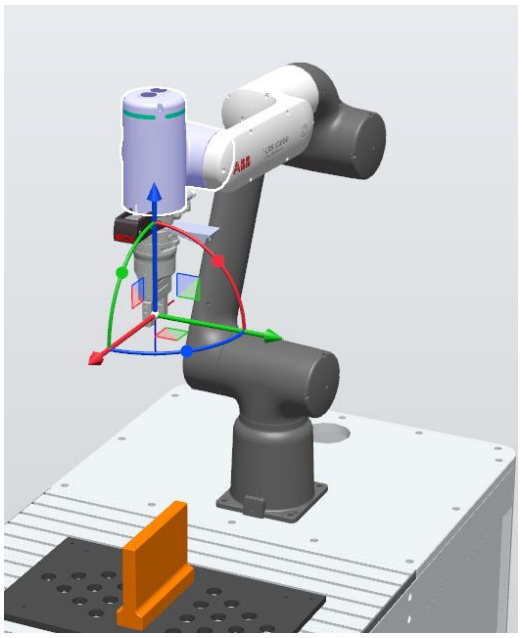
2. V Home meniju pritisnite ukaz za Jog Joint. Z miško se postavite na robotsko roko in izberite posamezno os, ki bi jo želeli premakniti. Pritisnite levi klik na miški in premikajte vsako os posebej.

Slika 40: Jog Joint



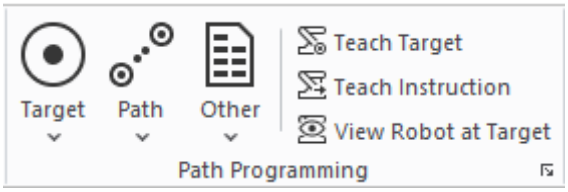
3. V Home meniju izberite ukaz Jog TCP. Nato klikni na robotsko roko in prikazale se bodo puščice za premik. Prav tako lahko napišete tudi vrednost zamika TCP-ja.

Slika 41: Jog TCP



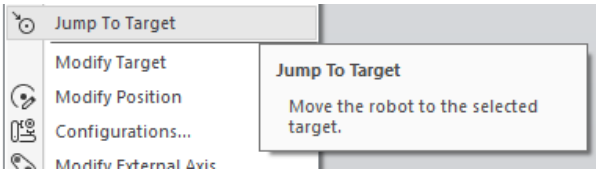
4. TCP lahko premaknete tudi direktno do točke z ukazom View Robot at Target. Ta ukaz deluje tako, da z desnim klikom kliknete na točko, ki bo shranjena v oknu Paths&Targets, nato pa izberete ta ukaz. Ko imaš TCP v željeni točki, poskrbite, da si ta ukaz ugasnete, drugače se bo robotska roka ob kliku na naslednjo točko premaknila.

Slika 42: Premik robotske roke do točke



5. Z desnim klikom na točko izberite Jump To Target. S tem ukazom se robotska roka premakne do željene točke.

Slika 43: Skok do točke



3.3 VAJA 2

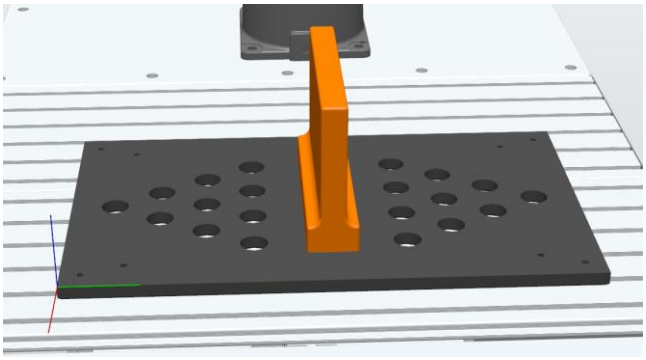
Pobiranje in odlaganje preko ovire enega valja.

Ustvarite program, v katerem bo robotska roka valj iz ene strani ovire prestavila na drugo stran. Program se mora začeti in končati v začetni poziciji. Robotska roka je v začetni poziciji takrat, ko so vse osi postavljene na 0°. Vsebovati mora točko nad valjem, točko pobiranja, točko vračanja, točko nad oviro, točko nad odlagalnim mestom, točko odlaganja in točko vračanja. V Rapid-u dodajte še zapiranje in odpiranje prijemala.

Vajo simulirajte.

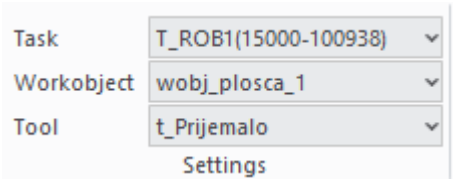
1. Ustvarite nov WorkObject v kotu delovne plošče in ga poimenujte wobj_plosca_1.

Slika 44: Nov Workobject



2. V nastavitvah v zavihku Home preverite, da imate izbran pravilen Tool in WorkObject.

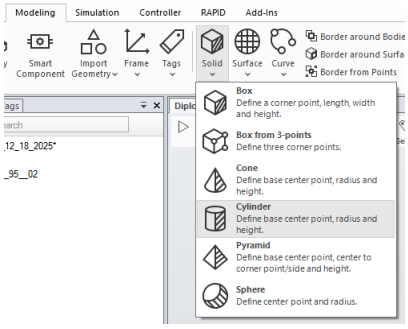
Slika 45: Izbira za Tool in WorkObject



Dodajanje valja

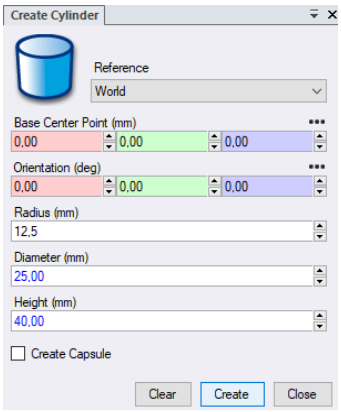
1. Odprite zavihek Modeling, nato okno Solid in izberite Cylinder.

Slika 46: Ustvarjanje valja



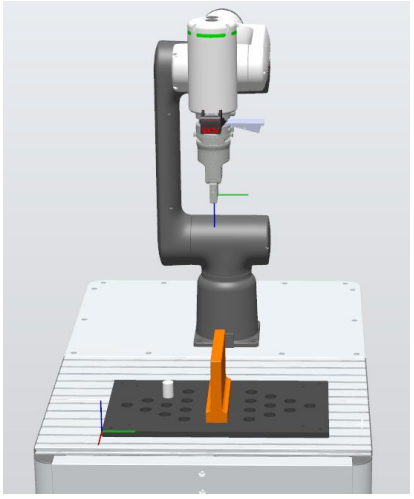
2. Odpre se okno, v katerega vpišete podatke o valju za višino 40 mm in premer 25 mm. Potrdite s Create.

Slika 47: Podatki o valju



3. V zavihku Layout ta valj preimenujte v Valj 1.
4. Z ukazom Move and Rotate ter Place valje premaknite na pobiralno mesto. Za pomoč za natančno postavitev uporabite orodje Measure in Move and Rotate.

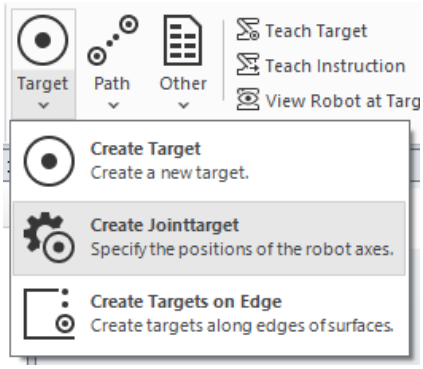
Slika 48: Izgled RC ob upoštevanih navodilih



Ustvarjanje skupne točke

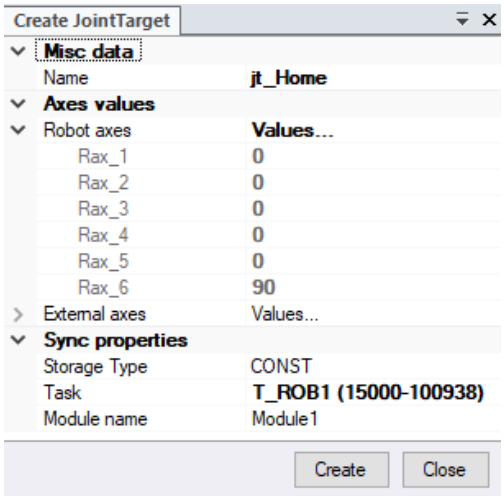
1. Najprej ustvarite izhodiščno točko gibanja. Točka mora biti tipa Jointtarget, saj jo bomo uporabili v več programih.
   Ustvarite jo tako, da v meniju Home izberete Target in nato Create Jointtarget.

Slika 49: Ustvarjanje Jointtarget



2. Odpre se pojavno okno, v katerega vnesete ime točke in pozicijo. Ime naj bo jt_Home. Za pozicije osi robota se nastavljajo joint vrednosti. Te postavite na 0°, razen 6. osi, ki jo nastavite na 90°.

Slika 50: Vrednosti Jointtarget

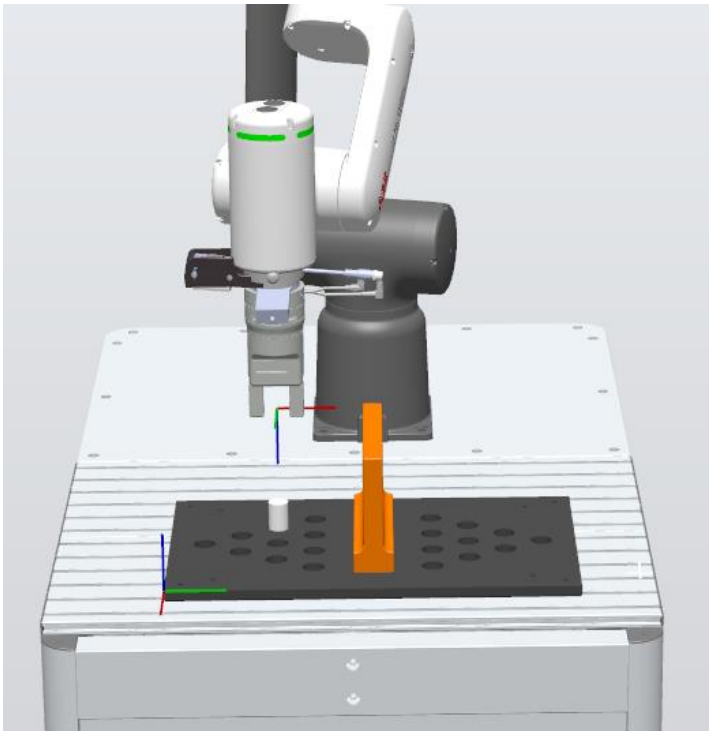


3. Potrdite s klikom na Create. Točka se shrani v oknu Paths&Targets v mapi Jointtargets.

Ustvarjanje točk pobiranja in odlaganja

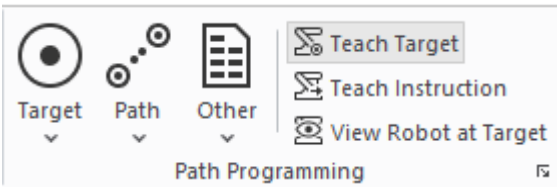
1. Robotsko roko z ukazom Jog TCP premaknite 50 mm nad valj, na mest kjer bomo ustvarili prvo točko, v kateri se prijemalo približa valju. Orodje naj bo postavljeno na sredino valja, pri tem si pomagajte z ukazi Measure.

Slika 51: Točka približevanja



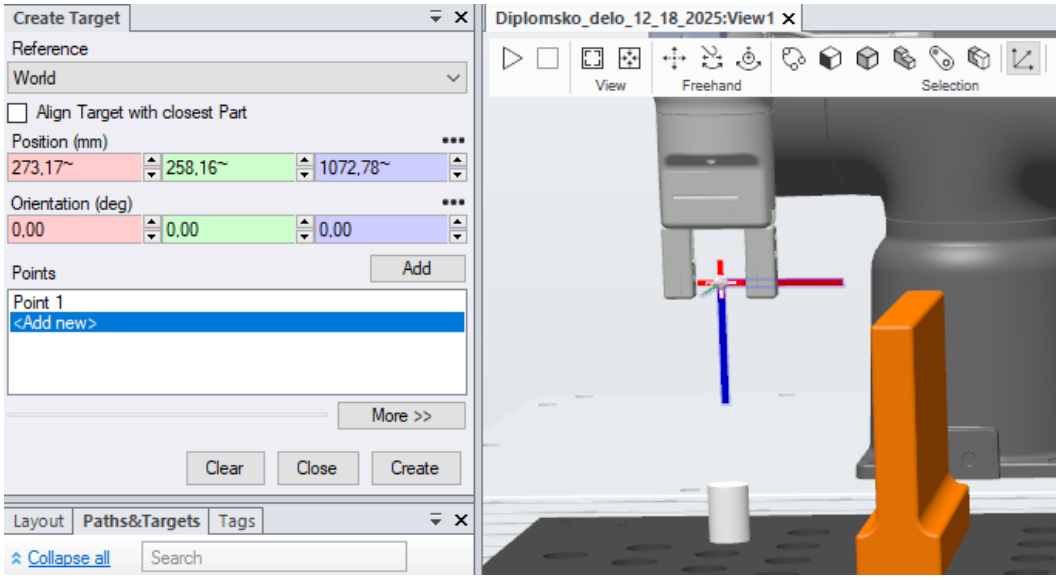
2. Ko je orodje na pravem mestu, v meniju Home izberemo ukaz Teach Target. Točka se bo shranila v oknu Paths&Targets v mapi wobj_plosca_1. Točko smiselno preimenujte.

Slika 52: Teach Target



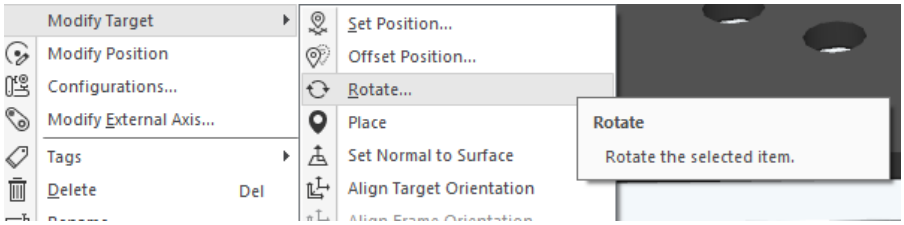
3. Drugi način, kako si lahko točko shranimo je, da v meniju Home izberemo Target in nato Create Target. Odpre se okno, v katerega vnesemo lokacijo trenutnega TCP-ja. Izberite ukaz Snap Object in nato se z miško postavite v rdeče polje za Position. Z miško nato označimo položaj TCP-ja in podatki se izpišejo v polja (Slika 53). Točko potrdimo z ukazom Create. Target. Točka se bo shranila v oknu Paths&Targets v mapi wobj_plosca_1. Točko smiselno preimenujte.

Slika 53: Ustvarjanje točke



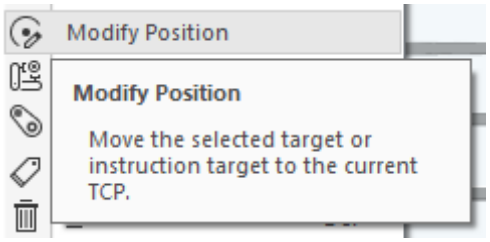
4. Če je orientacija shranjene točke napačna, jo je treba popraviti. Orientacija je pravilna takrat, ko je os Z usmerjena proti podstavku robota, X-os levo od robota, Y-os pa stran od robota. Če orientacija ne ustreza temu položaju, jo lahko spremenimo tako, da z desnim klikom miške kliknemo na točko, katere orientacijo želimo spremeniti, nato izberemo Modify Target in Rotate. V oknu, ki se odpre, izberemo os, okoli katere želimo izvesti rotacijo in določimo kot zasuka.

Slika 54: Rotiranje točk



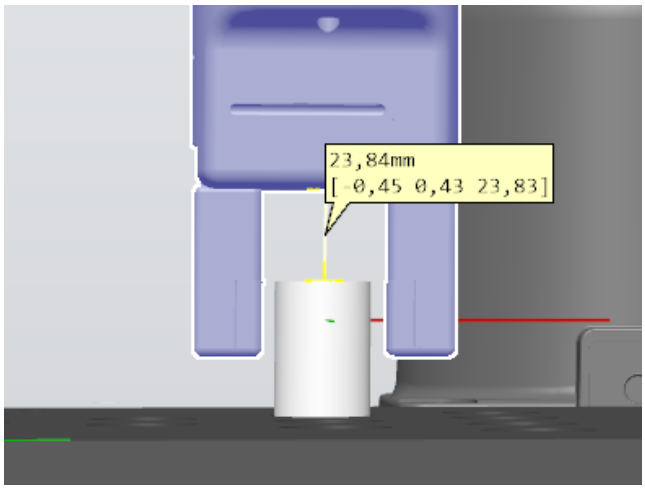
5. Če je postavitev robota še zmeraj v točki, v kateri smo točko shranili, lahko točke rotiramo z ukazom Modify Position. Pri tem bodite pozorni, da je TCP v točki, ki jo želite shraniti za robota, saj spremeni rotacijo in pozicijo točke. Ukaz je zelo uporaben takrat, ko želimo že narejeno točko premakniti na drugo pozicijo.

Slika 55: Urejanje pozicije



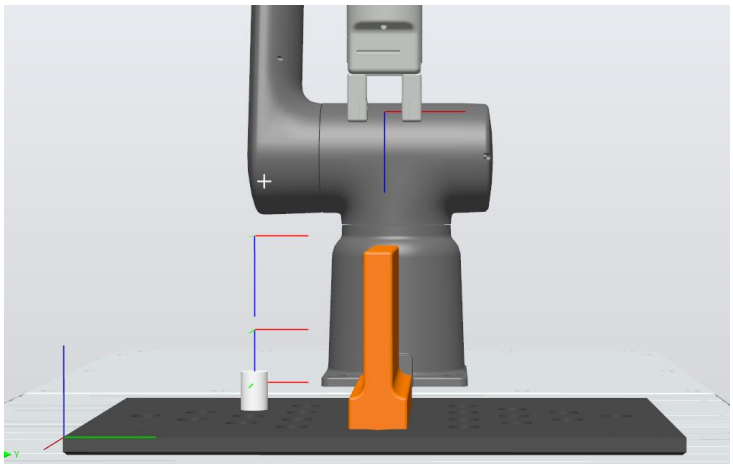
6. Točke lahko premikamo tudi vizualno. Levi klik na točko in nato ukaz Move and Rotate. Tako lahko točko vizualno zamaknete na želeno pozicijo.
7. Za ustvarjanje točke pobiranja premaknite orodje nad valj. Razdalja med valjem in orodjem mora biti 23,83 mm.

Slika 56: Razdaja za točko pobiranja



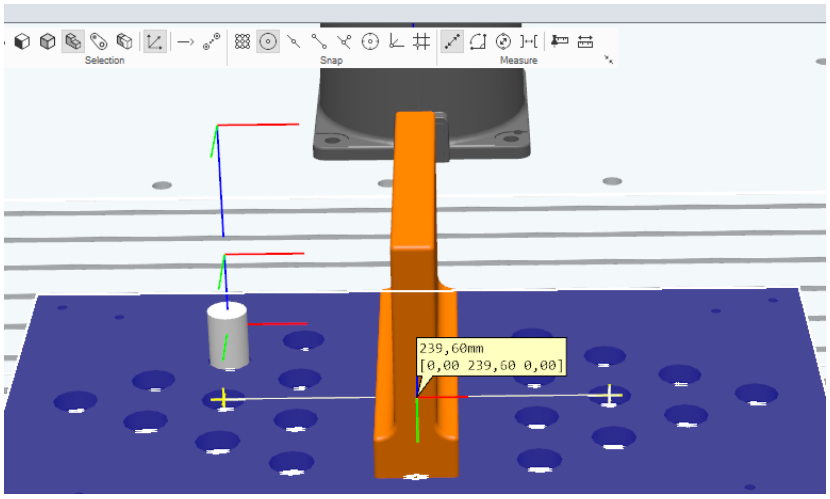
8. Ustvarite točko pobiranja. Ko je TCP na pravem mestu, točko shranite po enem izmed prej navedenih načinov.
9. Za točko vračanja premaknite orodje po osi Z za 100 mm in shranite točko.
10. Za premik nad oviro premaknite orodje po osi Y in Z nad oviro in ustvarite točko.

Slika 57: Točka pomika nad oviro



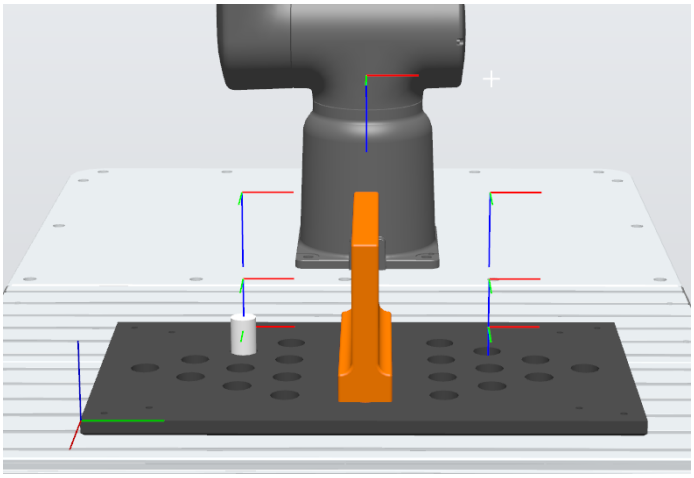
11. Točka nad mestom odlaganja valja mora biti 100 mm nad valjem. Da to točko najlažje ustvarimo, se premaknite na točko vračanja. Izmerite razdaljo med mestom za pobiranje in odlaganje in TCP premaknite po osi Y za izmerjeno vrednost. Ustvarite točko in jo poimenujte.

Slika 58: Razdalja med pobiranjem in odlaganjem



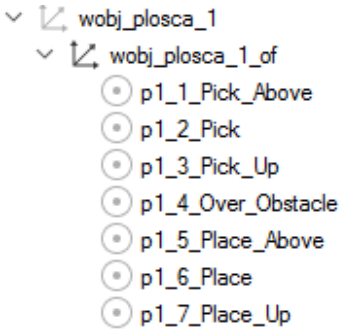
12. Točko, v kateri prijemalo odloži valj, ustvarite na enak način.
13. Točka vračanja mora biti 50 mm nad valjem. Ustvarite jo po enakem načinu.

Slika 59: Pogled na ustvarjene točke



14. Shranjene točke se izpišejo v oknu Paths&Targets v mapi aktivnega WorkObject-a.

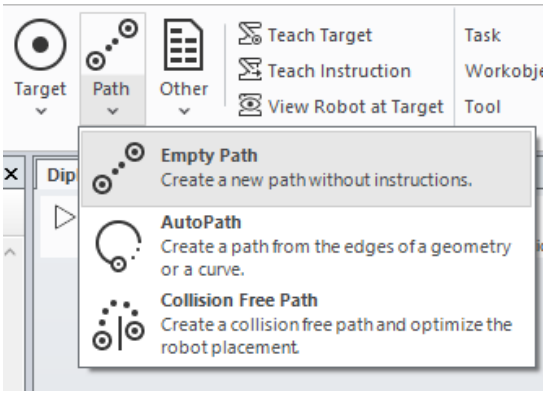
Slika 60: Točke za Vajo 2



Ustvarjanje poti gibanja

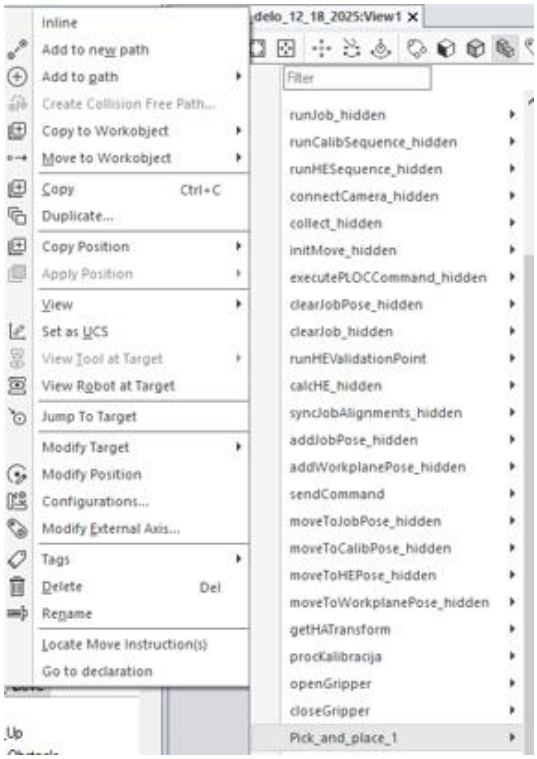
1. V meniju Home izberite ukaz Path in Empty Path. Ustvarila se bo prazna pot, v katero bomo dodali ukaze. Pot preimenujte.

Slika 61: Ukaz za Path



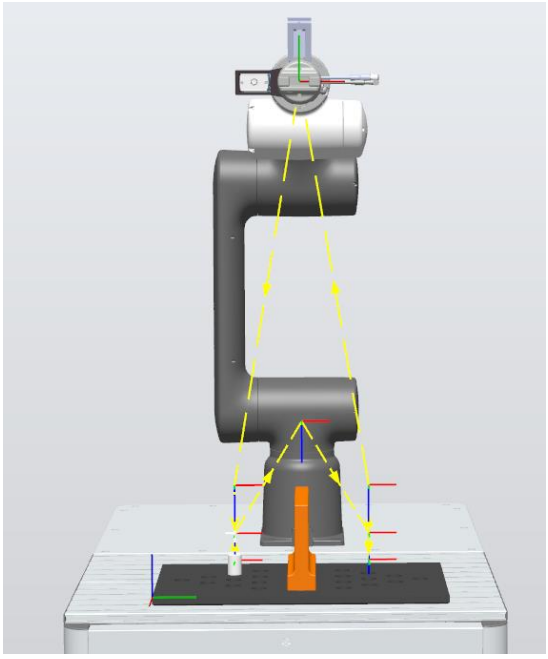
2. V zavihku Paths&Targets z desnim klikom kliknete na prvo točko. Na vrhu se izpiše Add to path in nato ponudi izbiro, da točko postavite v Path. Izberite tega, ki ste ga zdaj naredili in označite First, da bo prva točka v ustvarjeni poti.

Slika 62: Izbira poti



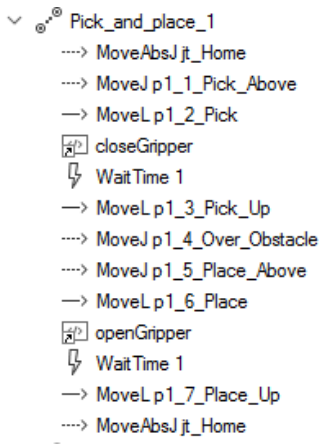
3. Enak proces uporabite pri ostalih točkah, vendar pri vsaki naslednji označite ukaz Last.
4. Na začetek in konec poti dodajte prej ustvarjen Jointtarget z Joint gibom.
5. Izriše se pot, po kateri se bo premikal TCP orodja.

Slika 63: Pot gibanja pri Vaji 2



6. V zavihku Home in nato Paths&Targets v mapi Paths&Procedures pod imenom, ki ste jo ustvarili, se izpiše celoten potek programa.

Slika 64: Ustvarjeni koraki poti



7. Ko so vse točke dodane v Path, celotno RC sinhronizirajte v Rapid.

Programiranje v Rapidu

1. V zavihku Rapid, v Modul 1, so se shranile točke in ustvarjena pot.
2. V programu dodajte komentarje in ukaze za odpiranje ter zapiranje prijemala. Določite, kateri gib v programu bo linearen in kateri joint. Program je narejen tako, da se robot ustavi samo v točki 2 in 6, torej v točkah, kjer pobira oziroma odlaga valj.

Slika 65: Program za Vajo 2

```
! Zacetek programa za 1 valj
PROC Pick_and_place_1()
    ! Zacetna pozicija
    MoveAbsJ jt_Home,v1000,z100,t_Prijemalo\WObj:=wobj_plosca_1;
    ! Premik nad valj 1
    MoveJ p1_1_Pick_Above,v300,z100,t_Prijemalo\WObj:=wobj_plosca_1;
    ! Spust do valja
    MoveL p1_2_Pick,v300,fine,t_Prijemalo\WObj:=wobj_plosca_1;
    ! Zapri prijemalo
    closeGripper;
    WaitTime 1;
    ! Dvig valja
    MoveL p1_3_Pick_Up,v200,z100,t_Prijemalo\WObj:=wobj_plosca_1;
    ! Pomik nad oviro
    MoveJ p1_4_Over_Obstacle,v200,z100,t_Prijemalo\WObj:=wobj_plosca_1;
    ! Pomik nad odlagalno mesto
    MoveJ p1_5_Place_Above,v200,z100,t_Prijemalo\WObj:=wobj_plosca_1;
    ! Spust valja
    MoveL p1_6_Place,v200,fine,t_Prijemalo\WObj:=wobj_plosca_1;
    ! Odpri prijemalo
    openGripper;
    WaitTime 1;
    ! Dvig prijemala
    MoveL p1_7_Place_Up,v300,z100,t_Prijemalo\WObj:=wobj_plosca_1;
    ! Zacetna pozicija
    MoveAbsJ jt_Home,v1000,fine,t_Prijemalo\WObj:=wobj_plosca_1;
ENDPROC
```

3. V zavihku poiščite in preverite IOGripper_BASE, če se ujema (Slika 66). V kolikor se ne, ga dopolnite in popravite.

Slika 66: Program za prijemalo

```
1  MODULE IOGripper_BASE
2      PROC openGripper()
3          SetDO ABB_Scalable_IO_0_DO2,0;
4          SetDO ABB_Scalable_IO_0_DO1,0;
5          WaitTime 0.5;
6          SetDO ABB_Scalable_IO_0_DO2,1;
7          WaitTime 1;
8      ENDPROC
9      PROC closeGripper()
10         SetDO ABB_Scalable_IO_0_DO1,0;
11         SetDO ABB_Scalable_IO_0_DO2,0;
12         WaitTime 0.5;
13         SetDO ABB_Scalable_IO_0_DO1,1;
14         WaitTime 1;
15     ENDPROC
16 ENDMODULE
```

4. V zavihku mainModule določimo, kateri program se bo izvajal. Znotraj njega kličete programe iz Module1.

Slika 67: Glavni program

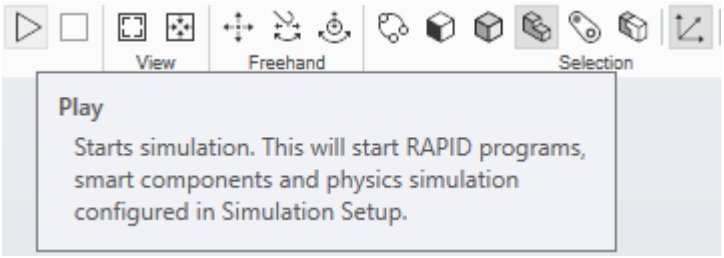
```
  ! Glavni program
PROC main()

  !Klic pick and place 1
  Pick_and_place_1;
  WaitTime 1;
```

5. Ko so programi urejeni, je potrebno Rapid del sinhronizirati v robotsko celico, da bodo spremembe vidne tudi tam.
6. Sinhronizirajte Rapid v robotsko celico.

Simulacija vaje

1. Vajo lahko simuliramo na dva načina:
   - V zavihku Home, v vrstici s pomožnim orodjem, pritisnemo Play. Simulacija se bo začela predvajati. Ko jo želimo ustaviti, pritisnemo Stop.

Slika 68: Simuliranje vaje



   - V zavihku Simulation, kjer lahko simulacijo tudi pavziramo in jo vodimo koračno.

Slika 69: Simuliranje vaje

2. Postopek najprej zaženemo v koračnem načinu, nato še v avtomatskem načinu. Po uspešni simulaciji vaje skrijte komponente in točke, ki jih ne potrebujete pri naslednji vaji. Izberite komponento, točko, WorkObject, orodje ali pot (Tool ali Path) in z desnim klikom izberite ukaz View, da odkljukajte Visible. Komponenta se bo skrila. Če jo želite ponovno videti, obkljukajte Visable.

3.4 VAJA 3

Sledenje 2D enostavni konturi z varilno pištolo.

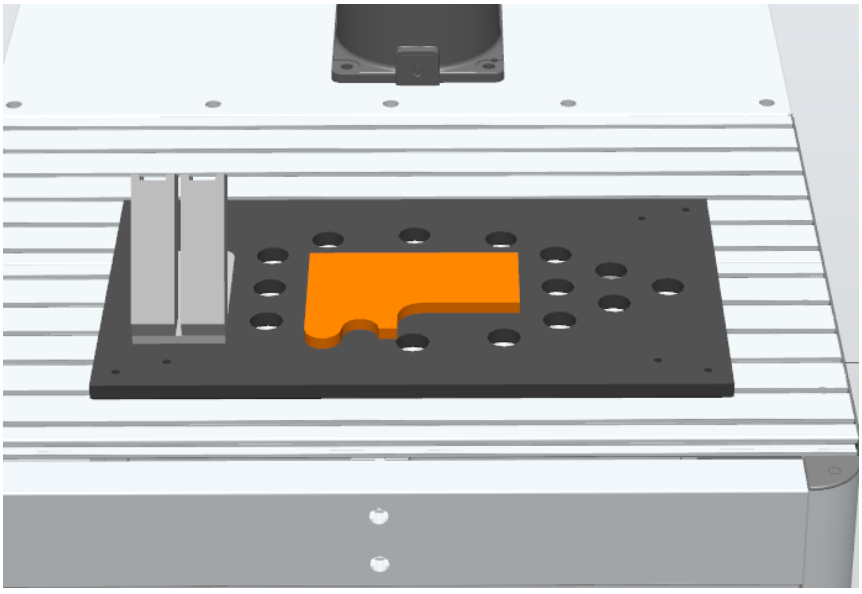
Robotska roka začne program v začetni poziciji. Pomakne se proti držalu orodja, kjer pobere varilno pištolo. Dvigne jo iz pobiralnega mesta in premakne do pozicije nad prvo točko sledenja. Ko konča s sledenjem konture, se v zadnji točki dvigne nad konturo in nato orodje vrne v mesto, kjer ga je pobralo. Kjer je vogal, se mora robotska roba ustaviti.
Ustvarite tri podprograme. Enega za pobiranje orodja, enega za sledenje konturi in enega za odlaganje orodja.

Upoštevajte, da mora biti hitrost premika ob sledenju konturi 20 mm/s.

Dodajanje komponent

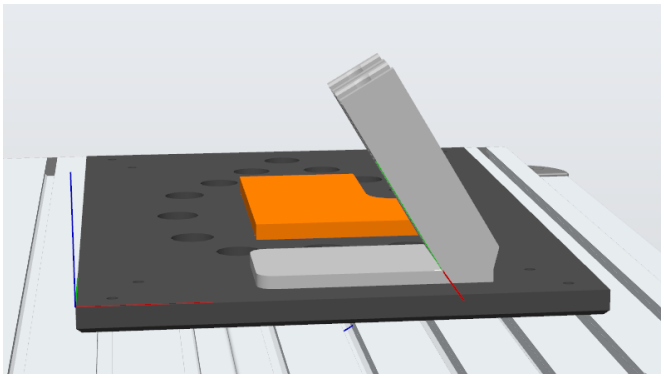
1. S funkcijo Import Geometry dodajte 2D_Kontura.SAT. Postavite jo s spodnjimi nogami na sredino delovne plošče.
2. S funkcijo Import Geometry dodajte Držalo_orodja.SAT. Na delovno ploščo jo postavite tako, kot je prikazano na Slika 70.

Slika 70: Postavitev komponent za Vajo 3



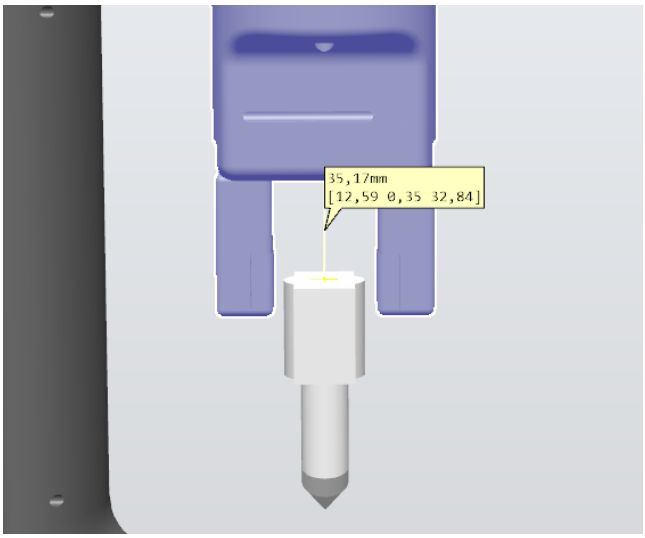
3. Ustvarite nov WorkObject v vogalu delovne plošče in ga poimenujte wobj_2D_kontura.
4. Ustvarite WorkObject za držalo orodja z metodo treh točk.

Slika 71: WorkObject za držalo orodja



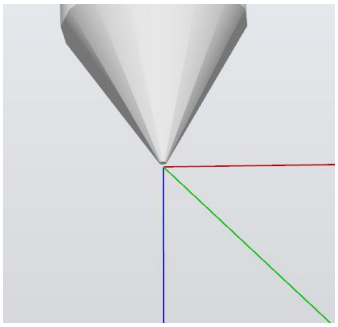
5. V RC naložite orodje za varjenje in ga postavite v sredino prijemala. Razdalja med varilno pištolo in orodjem mora biti 32,84 mm. Ko je varilna pištola na sredini prijemala, jo priključite na prijemalo s funkcijo Attach to.

Slika 72: Razdalja med varilno pištolo in prijemalom



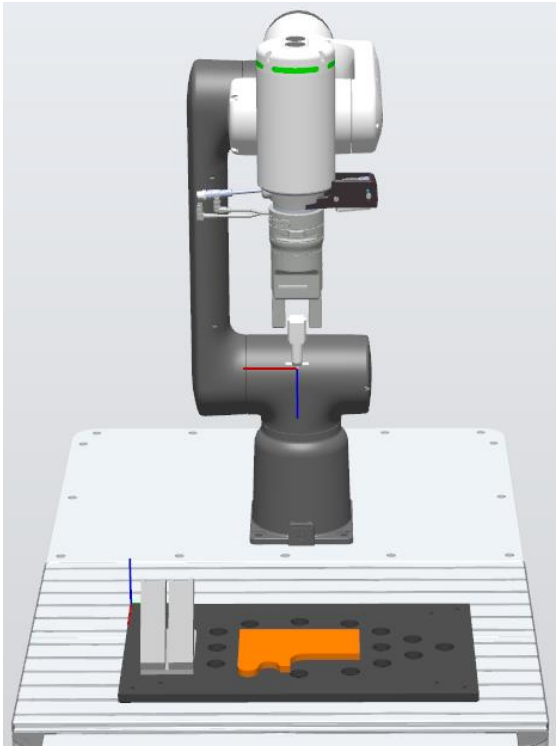
6. Ustvarite nov Tooldata in ga poimenujte t_Varilna_Pistola. Ustvarjen TCP premaknite za 0,25 mm po osi Z, da se ne dotika same varilne pištole. S tem se ob simulaciji orodje ne bo premikalo po komponenti, vendar bo sledilo robu nad to komponento.

Slika 73: Zamik TCP-ja



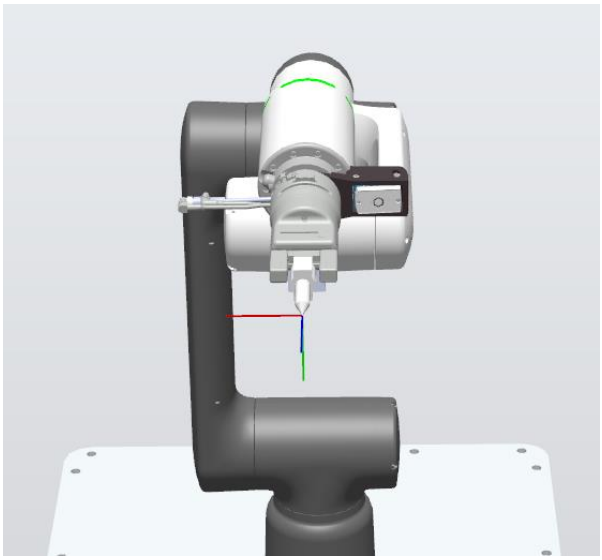
7. Ustvarjena robotska celica more vsebovati vse WorkObject, Tooldata in vse komponente.

Slika 74: RC za Vajo 3



8. Ustvarite nov Jointtarget, točko ki bo služila kot Home pozicija za orodje Varilna pištola in jo poimenujte. Upoštevajte, da mora biti konica varilne pištole obrnjena navzdol in naprej.

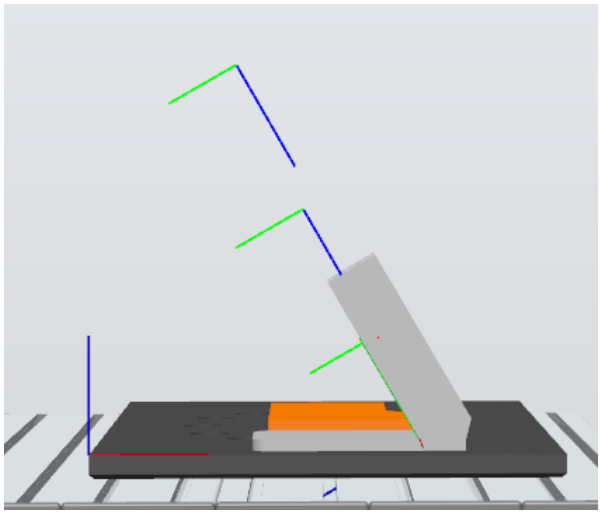
Slika 75: Nov Jointtarget



Ustvarjanje točk

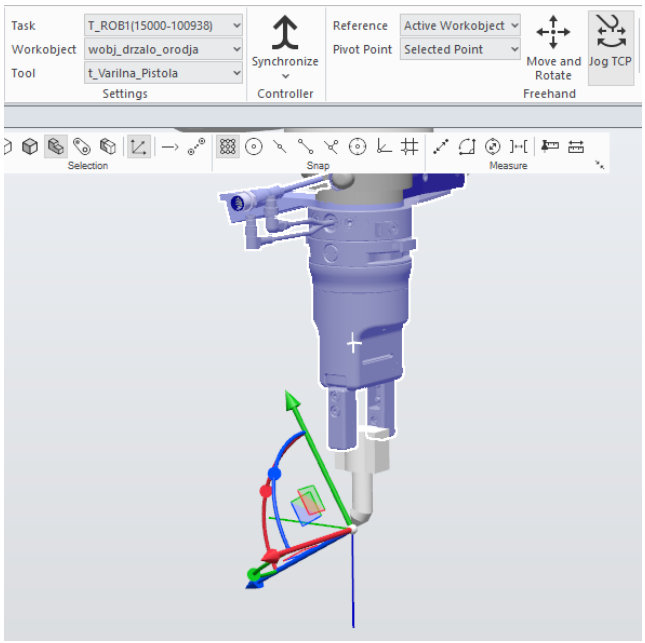
1. Za ustvarjanje točk podprograma za pobiranje in odlaganje varilne pištole uporabite metodo Jog TCP. Nastavite si tri točke in ustrezne korake pri vsaki. Bodite pozorni, da imate izbran ustrezen WorkObject in pravilen Tool.

Slika 76: Ustvarjene točke za pobiranje in odlaganje orodja



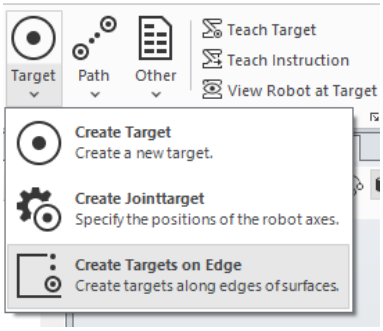
Ker držalo orodja ni postavljeno pravokotno na delovno ploščo, temveč je pod kotom, je pozicioniranje točk neustrezno z World referenco premikanja TCP-ja. V zavihku Home v oknu Reference, izberite Active WorkObject. S tem se spremeni koordinatni sistem premika TCP-ja in omogoča lažje pozicioniranje točk. Nato se z ukazom Jog TCP za izmerjeno vrednost premaknite.

Slika 77: Nastavitev reference



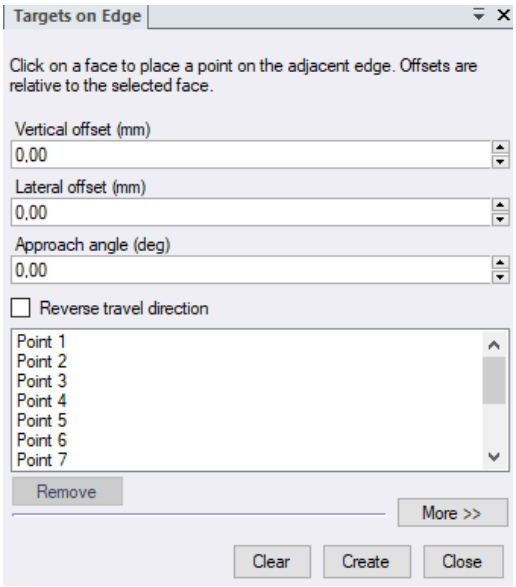
2. Pri ustvarjanju točk podprograma za sledenje konturi najprej ustvarite točko, v kateri se robotska celica približa konturi. Naj bo nad točko v kateri se bo začel izvajati program sledenja.
3. Za ustvarjanje točk na konturi, lahko uporabite dva načina.
   - Prvi način je s sledenjem roba konture. V meniju Home izberite ukaz Traget in nato Create Targets on Edge.

Slika 78: Ukaz za točke na robu konture



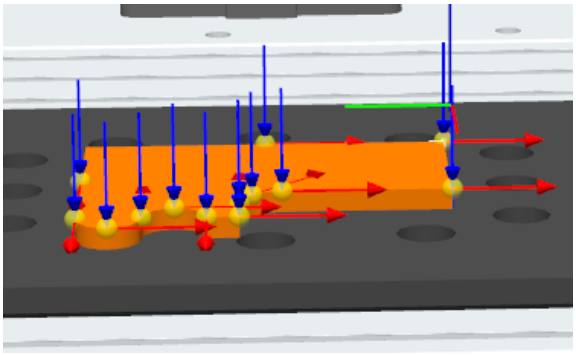
Odpre se okno, v katerega se bodo izpisovale točke, ki jih boste izbrali. Ko boste točke imeli ustvarjene, jih v tem oknu potrdite s Create.

Slika 79: Potrjevanje točk



Točke ustvarite tako, da imate v pomožnem orodju izbrano funkcijo Snap Object. Z miško se pomaknite nad konturo in vizualno se vam bodo prikazovale točke. Kjer so krožni gibi, je potrebno upoštevati, da se v točki pred samim lokom gib začne. Nato ustvarite dve točki, eno približno na sredini loka in eno na koncu. Le tako bo krmilnik pravilno izvedel krožni gib. Dodajte točke v vogalih. Pri tem upoštevajte, da boste v vsakem kotu potrebovali dve točki zaradi zasuka robotske roke iz enega položaja v drugega. Točke poimenujte. Prednost tega načina je, da se točke hitreje ustvarijo.

Slika 80: Točke na robu konture



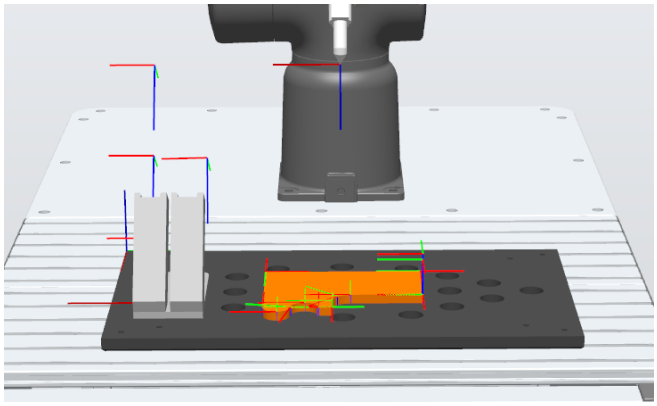
Ustvarite še točko, v kateri se orodje premakne pravokotno nad konturo.

Vsem točkam nastavite pravilno orientacijo. Stisnite točko in nato View Robot at Target, da boste lahko videli trenutno pozicijo robota v točki. Prilagodite jo tako, da bo ustrezal konturi.

Ustvarite še eno točko, v kateri se bo robot bolj naravno premaknil od mesta za pobiranje do konture. Naj bo nekje vmes med držalom orodja in konturo.

- Drugi način programiranja točk je z ročnim vodenjem TCP-ja do vsake točke. Pomagamo si lahko z orodji Measure in Snap, da dobimo podatek za koliko se moramo premakniti do katere točke in se nato z ukazom Move and Rotate premaknemo. Prednost tega načina je, da lahko sproti nastavljamo orientacijo orodja in vidimo, kaj se v kateri točki dogaja.

Slika 81: Ustvarjene točke programa za Vajo 3

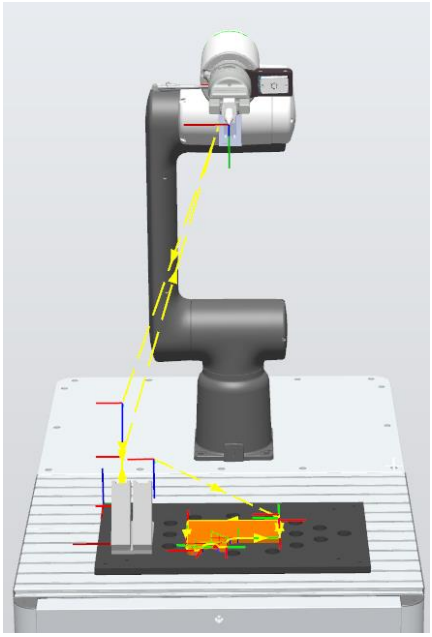


4. Vsaka točka se je shranila v izbran WorkObject.
5. Ustvarite poti gibanja robotske roke.
   Naredite tri poti.
   - Eno za pobiranje orodja, ki se mora začeti v Home poziciji in končati v točki dviga orodja.
   - Druga za odlaganje orodja, ki se začne v točki nad mestom odlaganja in konča v Home poziciji.

- Tretja pot za sledenje konturi, ki se začne v točki med držalom orodja in konturo ter konča v točki umika od konture.

Vsako pot smiselno poimenujte.

Slika 82: Pot gibanja za Vajo 3



6. Sinhronizirajte robotsko celico v Rapid.

Programiranje v Rapidu

1. V podprogramu dodajte komentarje in ukaze za odpiranje ter zapiranje prijemala. Določite, kateri gib v programu bo kartezični in kateri osni. Nastavite hitrost sledenja konturi in kako se v katerem koraku točki približa.

Slika 83: Program za pobiranje orodja

```
!Program za pobiranje orodja
PROC p_Drzalo_orodja()
    ! Zacetna pozicija
    MoveAbsJ jt_Home,v1000,z100,t_Prijemalo\WObj:=wobj_drzalo_orodja;
    ! Postavitev nad orodje
    MoveJ O_1_Pick_Above,v1000,z100,t_Prijemalo\WObj:=wobj_drzalo_orodja;
    ! Pobiranje orodja
    MoveL O_2_Pick,v1000,z100,t_Varilna_Pistola\WObj:=wobj_drzalo_orodja;
    ! Zapri prijemalo
    closeGripper;
    WaitTime 1;
    ! Dvig prijemala
    MoveL O_3_Pick_Up,v1000,z10,t_Varilna_Pistola\WObj:=wobj_drzalo_orodja;
ENDPROC
```

Slika 84: Program za odlaganje orodja

```
!Program za odlaganje orodja
PROC p_drzalo_orodjak()
    ! Postavitev nad drzalo orodja
    MoveJ O_1_Pick_Above, v1000, z100, t_Varilna_Pistola\WObj:=wobj_drzalo_orodja;
    ! Odlaganje orodja
    MoveL O_2_Pick, v500, fine, t_Varilna_Pistola\WObj:=wobj_drzalo_orodja;
    ! Odpri prijemalo
    openGripper;
    WaitTime 1;
    ! Dvig prijemala
    MoveL O_3_Pick_Up, v500, z10, t_Varilna_Pistola\WObj:=wobj_drzalo_orodja;
    ! Zacetna pozicija
    MoveAbsJ jt_Varjenje_HOME,v1000,z100,t_Varilna_Pistola\WObj:=wobj_drzalo_orodja;
ENDPROC
```

Slika 85: Program za Vajo 3

```
! Zacetek programa za varjenje 2D konture
PROC p_Welding_2D()
    MoveJ RT3, v1000, z50, t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveJ AP1, v1000, z50, t_Varilna_Pistola\WObj:=wobj_2D_kontura;

    MoveL k2D_1,v1000,fine,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveL k2D_2,v1000,fine,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveL k2D_3,v1000,fine,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveL k2D_4,v1000,fine,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveL k2D_5,v1000,fine,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveL k2D_6,v1000,z5,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveC k2D_7,k2D_8,v150,z10,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveC k2D_9,k2D_10,v150,z10,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveL k2D_11,v1000,z1,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveL k2D_12,v1000,z1,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveL k2D_13,v1000,z1,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveL k2D_14,v1000,z5,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveC k2D_15,k2D_16,v150,z10,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveL k2D_17,v1000,fine,t_Varilna_Pistola\WObj:=wobj_2D_kontura;
    MoveJ RT1, v20, z50, t_Varilna_Pistola\WObj:=wobj_2D_kontura;

ENDPROC
```

2. Ko so podprogrami urejeni, jih zapišete v glavnega. Programe iz drugih vaj komentirajte, da jih program ne bo upošteval.

Slika 86: Klic podprogramov v glavnega za Vajo 3

```
! Klic Welding 2D
p_Drzalo_orodja;
p_Welding_2D;
p_drzalo_orodjak;
WaitTime 1;
```

3. Sinhronizirajte Rapid v robotsko celico.

Simulacija vaje

1. V robotski celici preverite delovanje simulacije s pomočjo ukaza Play. Postopek najprej zaženemo v koračnem načinu, nato še v avtomatskem načinu.
2. Ob pravilnem gibanju robotske roke je vaja končana.

3.5 VAJA 4

Sledenje 2D zahtevnejše konture z varilno pištolo.

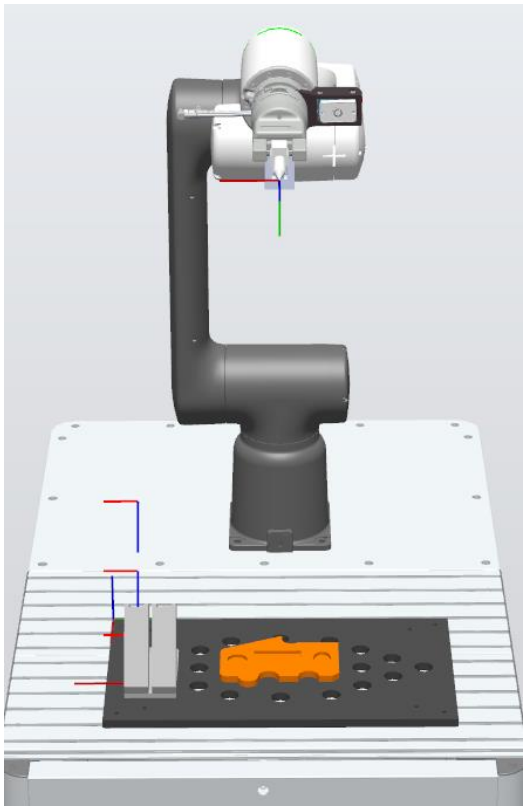
Robotska roka začne program v začetni poziciji. Pomakne se proti držalu orodja, kjer pobere varilno pištolo. Dvigne jo iz pobiralnega mesta in premakne do pozicije nad prvo točko sledenja. Sledenje začnete z enostavnejšimi oblikami in nadaljujete z zahtevnejšimi. Vsak lik rabi svoji točki približevanja obliki in vračanja. Orodje se lahko prične obračati, ko se vrača v točko vračanja. Ko konča z zunanjim robom konture, orodje vrne nazaj v držalo orodja. Kjer je vogal, se mora robotska roba ustaviti, v kolikor pa kota ni, se mora konstanto premikati.
Ustvarite podprogram, v katerem bo robotska roka sledila robu oblik. Vsaka oblika sledenja mora biti shranjena v svojem podprogramu.
Za pobiranje in odlaganje valja uporabite že narejena podprograma.

Upoštevajte, da mora biti hitrost premika ob sledenju konturi 20 mm/s.

Dodajanje komponent

1. S funkcijo Import Geometry dodajte 2D_Kontura_oblike.SAT. Postavite jo na sredino delovne plošče.
2. Ustvarite nov WorkObject v vogalu delovne plošče in ga poimenujte wobj_2D_kontura_oblike.
3. Preverite, da ima izbran pravilen WorkObject in Tool za izdelavo vaje.

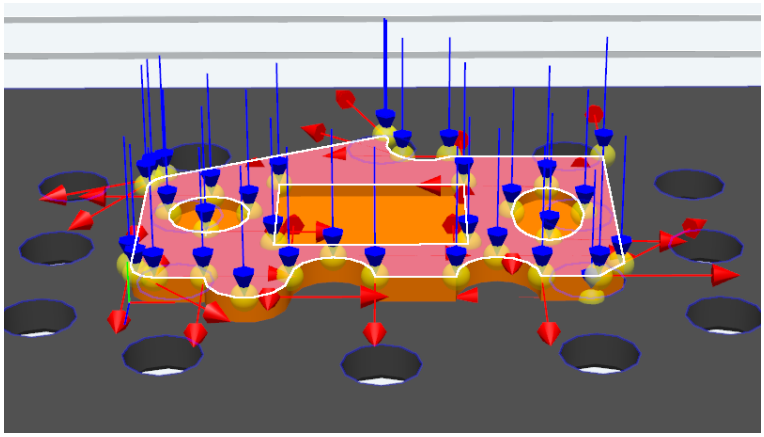
Slika 87: Postavitev v robotski celici za Vajo 4



Ustvarjanje točk

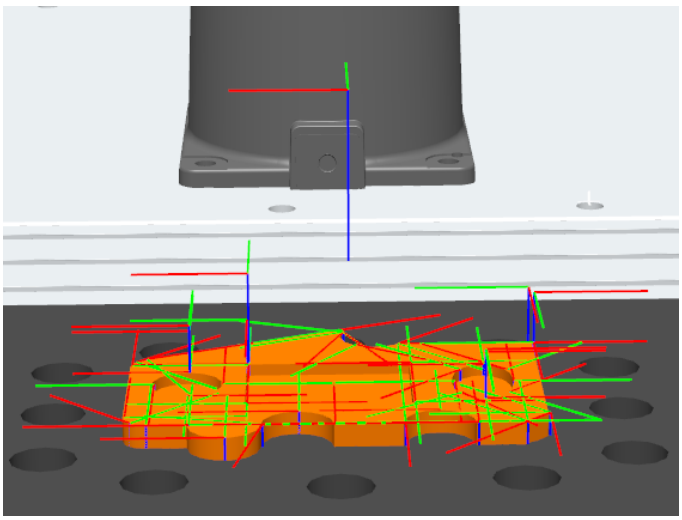
1. Ustvarite točke na konturi in jih poimenujte. Lahko tudi vsako obliko posebej. Točkam uredite orientacijo. Kjer je to potrebno, točko podvojite in ji spremenite rotacijo.

Slika 88: Ustvarjanje točk za Vajo 4



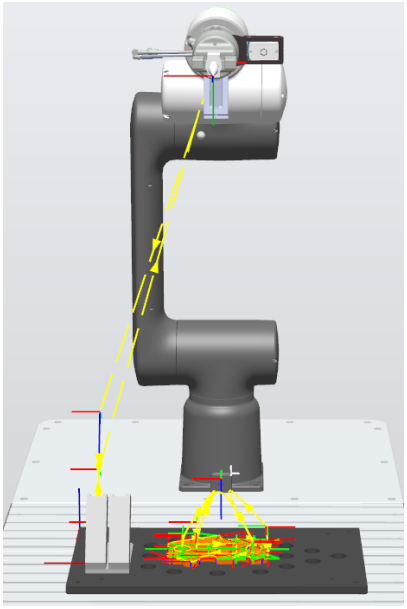
2. Vsaki obliki in robu dodajte točko približevanja konturi in vračanja. Da bo lažje sistemsko preverjati program, dodajte še za vsako obliko in robom eno točko nekje nad konturo.

Slika 89: Dodane točke za Vajo 4



3. Iz točk po vrsti ustvarite pot gibanja tako, da bo vsaka oblika shranjena v svojem podprogramu.

Slika 90: Ustvarjena pot Vaje 4



4. Sinhronizirajte robotsko celico v Rapid.

Programiranje v Rapidu

1. V podprograme dodajte komentarje in ukaze za odpiranje ter zapiranje prijemala. Nastavite hitrost sledenja konturi in kako se v katerem koraku točki približa. V podprogramu s komentarjem označite vsako obliko.

Slika 91: Vaje 4 – pravokotnik

```
! Pravokotnik
MoveJ k2Do_2e_20, v20, z5, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_3p_1, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_3p_2, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_3p_3, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_3p_4, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_3p_5, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_3p_6, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_3p_7, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_3p_8, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_3p_18, v20, z50, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_3p_9, v20, z100, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
```

Slika 92: Vaja 4 – krog

```
! Krog
MoveJ k2Do_3p_19, v20, z30, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_4k_1, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveC k2Do_4k_2, k2Do_4k_3, v20, z10, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveC k2Do_4k_4, k2Do_4k_5, v20, z10, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_4k_5, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_4k_15, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
```

Slika 93: Vaja 4 – elipsa

```
! Elipsa
MoveJ k2Do_1r_32, v20, z100, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_42, v20, z10, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_2e_1, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveC k2Do_2e_2, k2Do_2e_3, v20, z10, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveC k2Do_2e_4, k2Do_2e_5, v20, z10, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveC k2Do_2e_6, k2Do_2e_7, v20, z10, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveC k2Do_2e_8, k2Do_2e_9, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_2e_19, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_2e_10, v20, z100, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
```

Slika 94: Vaja 4 – rob

```
! Rob
MoveJ APP1, v500, z10, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_1, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_2, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_3, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_4, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_5, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_6, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_7, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveC k2Do_1r_8, k2Do_1r_9, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_10, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_11, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_12, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveC k2Do_1r_13, k2Do_1r_14, v20, z10, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveC k2Do_1r_15, k2Do_1r_16, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_17, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_18, v20, z10, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveC k2Do_1r_19, k2Do_1r_20, v20, z10, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_21, v20, z10, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveC k2Do_1r_22, k2Do_1r_23, v20, z10, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_24, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_25, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_26, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_27, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveC k2Do_1r_28, k2Do_1r_29, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_30, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL k2Do_1r_31, v20, fine, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
MoveL RTT1, v20, z10, t_Varilna_Pistola\WObj:=wobj_2D_kontura_oblike;
```

2. Ko so podprogrami urejeni, jih zapišete v glavnega.

Slika 95: Klic podprogramov v glavnega za Vajo 4

```
! Klic Welding 2D z oblikami
p_Drzalo_orodja;
p_Welding_2D_oblike;
p_Welding_2D_oblike_2;
p_Welding_2D_oblike_3;
p_Welding_2D_oblike_4;
p_drzalo_orodjak;
WaitTime 1;
```

Simulacija vaje

1. V robotski celice preverite delovanje simulacije. Postopek najprej zaženemo v koračnem načinu, nato še v avtomatskem načinu.
2. Ob pravilnem gibanju robotske je vaja končana.

3.6 VAJA 5

Nastavitev koordinatnega sistema orodja.

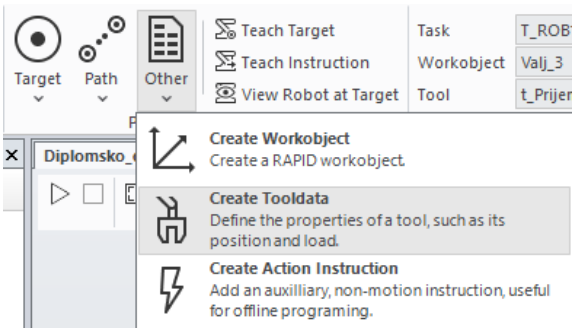
Nastavite nov Tooldata za prijemalo, konico brez zastavice in varilno pištolo.

Ustvarjanje Tooldata

Koordinatni sistem orodja lahko nastavimo na tri načine:

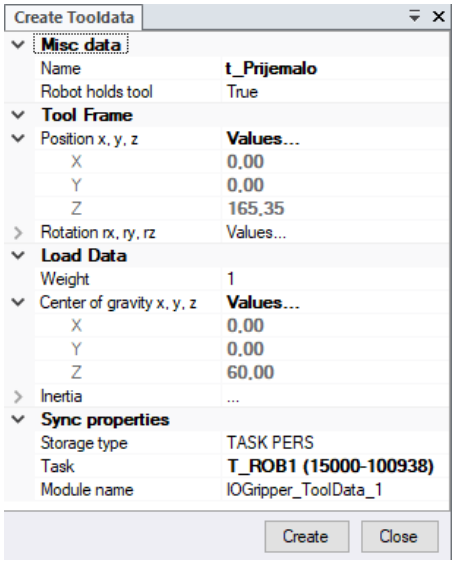
1. Z vpisovanjem pozicije:
    - V zavihku Home izberemo Other in nato Create Tooldata.

Slika 96: Ustvari Tooldata



- V oknu za kreiranje orodja, vnesemo ime orodja t_Prijemalo, pozicijo po osi Z 165,35 mm in maso po osi Z 60 kg.

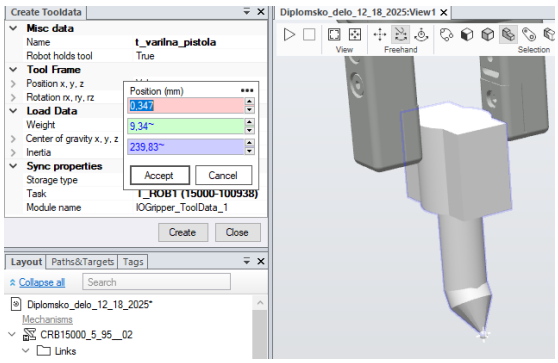
Slika 97: Podatki za Tooldata



- V zavihku Paths&Targets lahko vidite ustvarjen Tooldata.

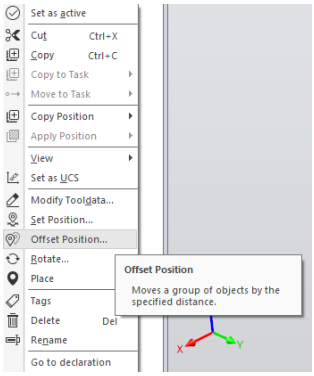
2. S klikom na točko:
   - V zavihku Home izberemo Other in nato Create Tooldata.
   - V oknu za kreiranje orodja se postavimo na rdeče polje in z ukazom Snap Object izberemo konico orodja.

Slika 98: Nastavitev Tooldata s klikom na točko



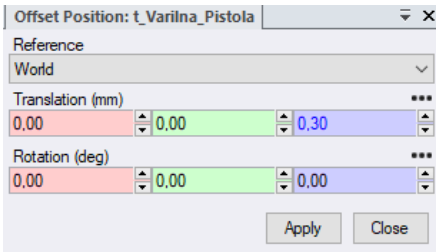
- Ko smo označili pozicijo, še vnesemo podatek za maso po osi Z 60 kg in ustvarimo Tooldata.
- Nato se izberemo ta ustvarjen Tooldata in izberemo Offset Position.

Slika 99: Offset Tooldata



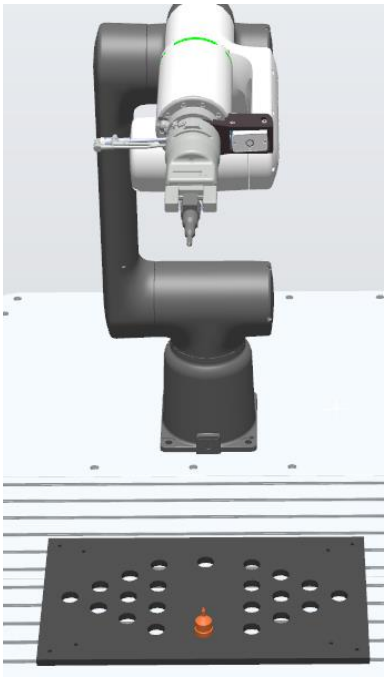
- Nastavimo Offset po osi Z za 0,3 mm, zato da se orodje ne bo dotikalo obdelovanca in da bo vmes nekaj prostora.

Slika 100: Zamik TCP



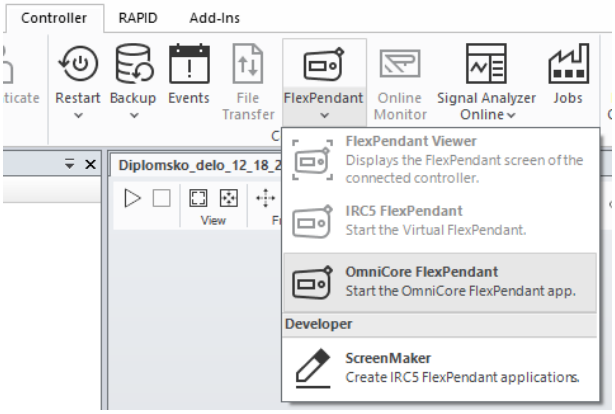
3. S pomočjo učne enote:
   - S funkcijo Import Geometry dodajte Konica_na_plosci. Postavite jo na sredino delovne plošče.
   - S funkcijo Import Geometry dodajte Konica_brez_zastavice. Postavite jo v prijemalo.

Slika 101: Postavitev za nastavitev TCP-ja



- V zavihku Controller odprite OmniCore FlexPendant.

Slika 102: OmniCore učna enota

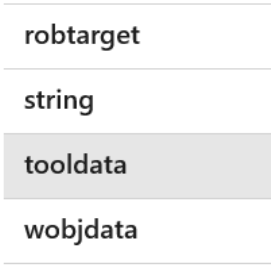


- Odpre se pojavno okno, v katerem je delujoča učna enota. Vsak premik robotske roke v njej bo viden tudi v robotski celici.
- Odprite okno s programskimi podatki.

Slika 103: Programski podatki

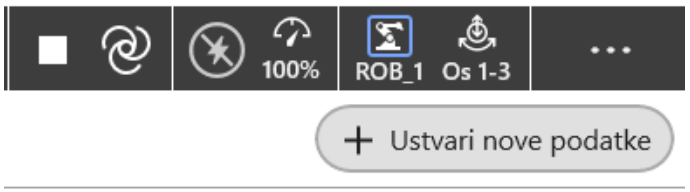
- Poiščite element Tooldata in ga izberite.

Slika 104: Element Tooldata



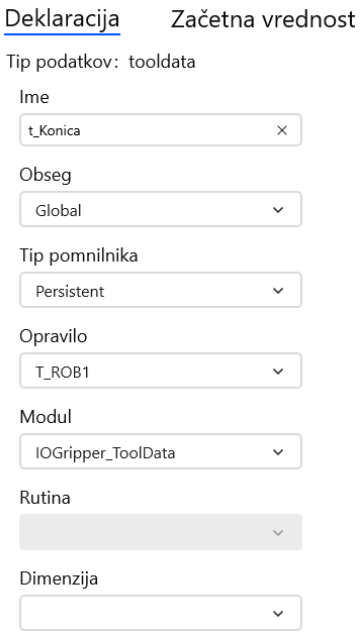
- Pojavi se ekran s shranjenimi podatki za Tooldata. V desnem kotu ekrana izberite možnost za ustvarjanje novih podatkov.

Slika 105: Ustvarjanje novih podatkov



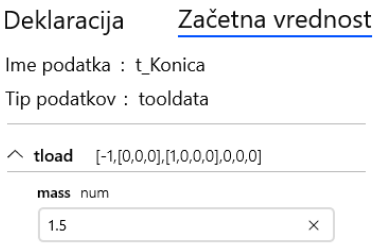
- V meniju Deklaracija določite ime za koordinatni sistem orodja in v kateri modul želite, da se shrani.

Slika 106: Deklaracija za Tooldata



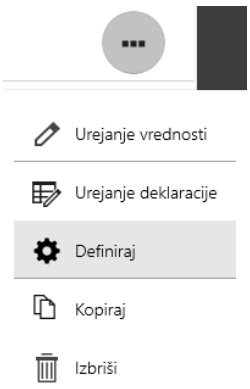
- V meniju Začetna vrednost spremenite samo podatek za tload mass, na 1,5.

Slika 107: Začetna vrednost za Tooldata



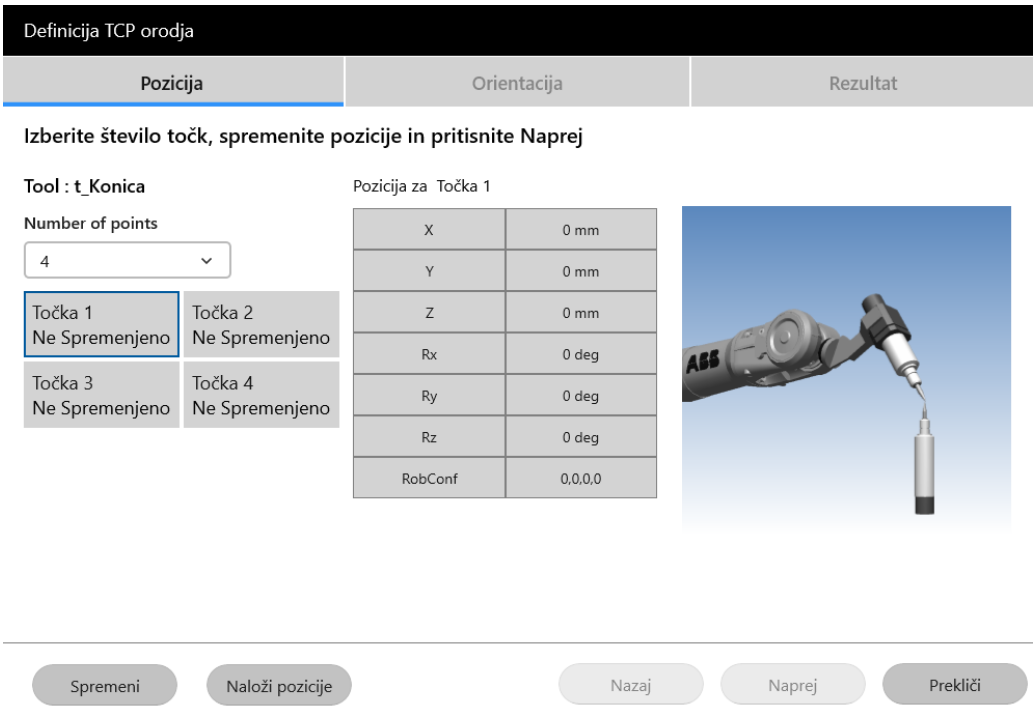
- V desnem kotu ekrana potrdite z gumbom Uporabi.
- Prikaže se novo narejen Tooldata.
- Na desni strani pri njem izberite tri pikice in nato ukaz Definiraj.

Slika 108: Definiranje Tooldata



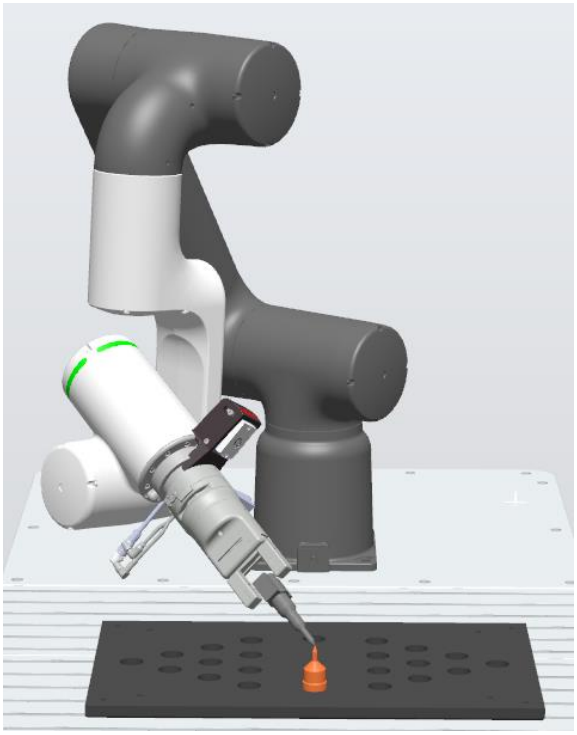
- Na učni enoti izberite metodo s 4 točkami.

Slika 109: Definicija TCP orodja



- Z ukazom Jog TCP premaknite robotsko roko v poljubni položaj tako, da se bo konica orodja dotikala konice na delovni plošči.

Slika 110: Pozicija 1



- Na učni enoti shranite pozicijo za točko 1 in z izbiro ukaza Spremeni jo prilagodite.

Slika 111: Sprememba točke

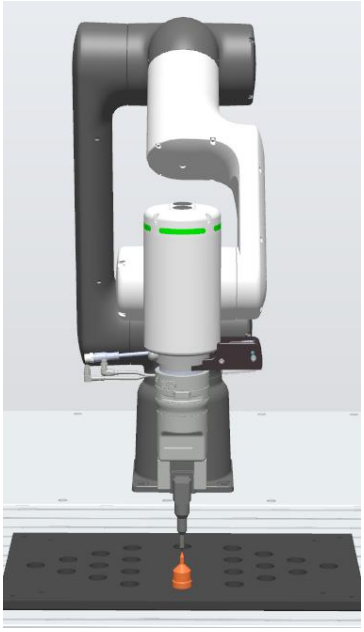
- Robotsko roko prestavite v drugo pozicijo, v kateri se bo konica orodja dotikala konice na delovni plošči.

Slika 112: Pozicija 2

- Na učni enoti izberite točko 2 in izberite ukaz Spremeni.

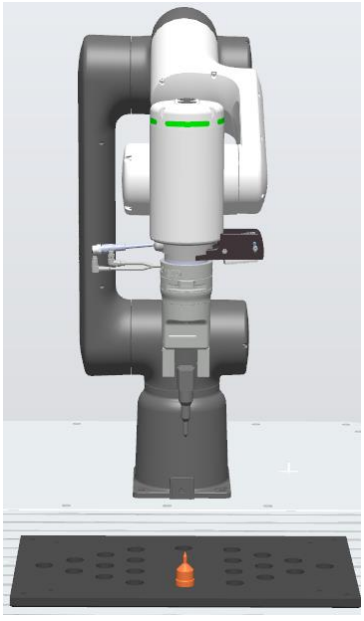
- Robotsko roko prestavite v tretjo pozicijo, v kateri se bo konica orodja dotikala konice na delovni plošči. Robotska roka naj bo postavljena pravokotno nad konico na delovni plošči.

Slika 113: Pozicija 3



- Na učni enoti izberite točko 3 in ukaz Spremeni.
- Pri točki 4 spremenite samo os Z točki 3. Pomembno je, da orodja ne preusmerjate med točko 3 in točko 4.

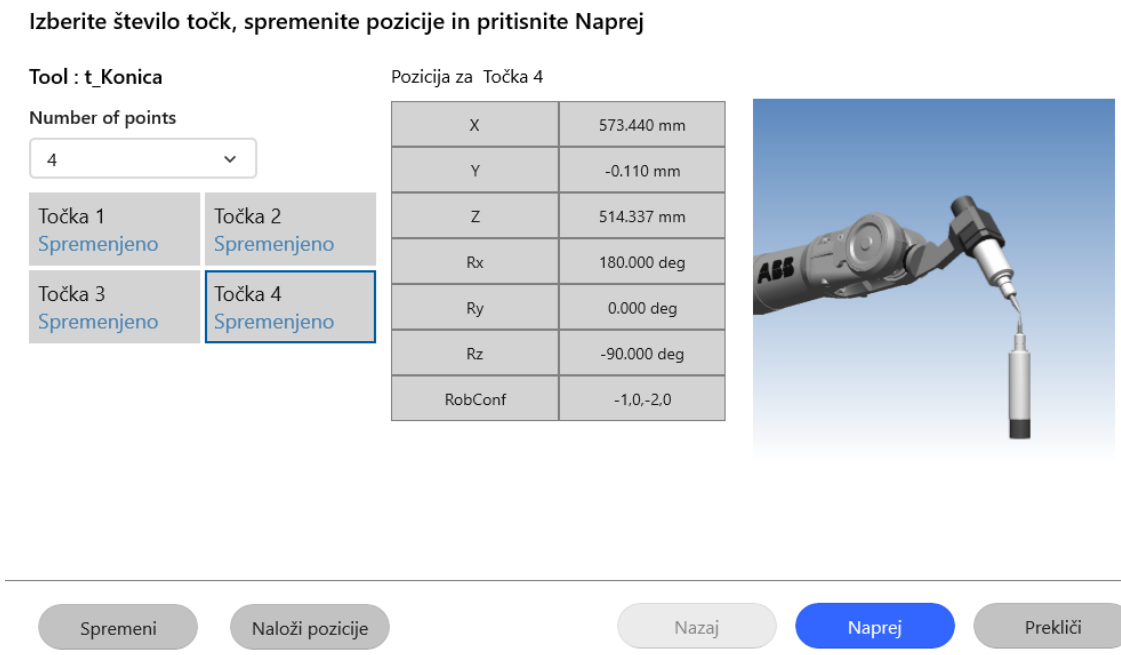
Slika 114: Pozicija 4



- Na učni enoti izberite točko 4 in izberite ukaz Spremeni.

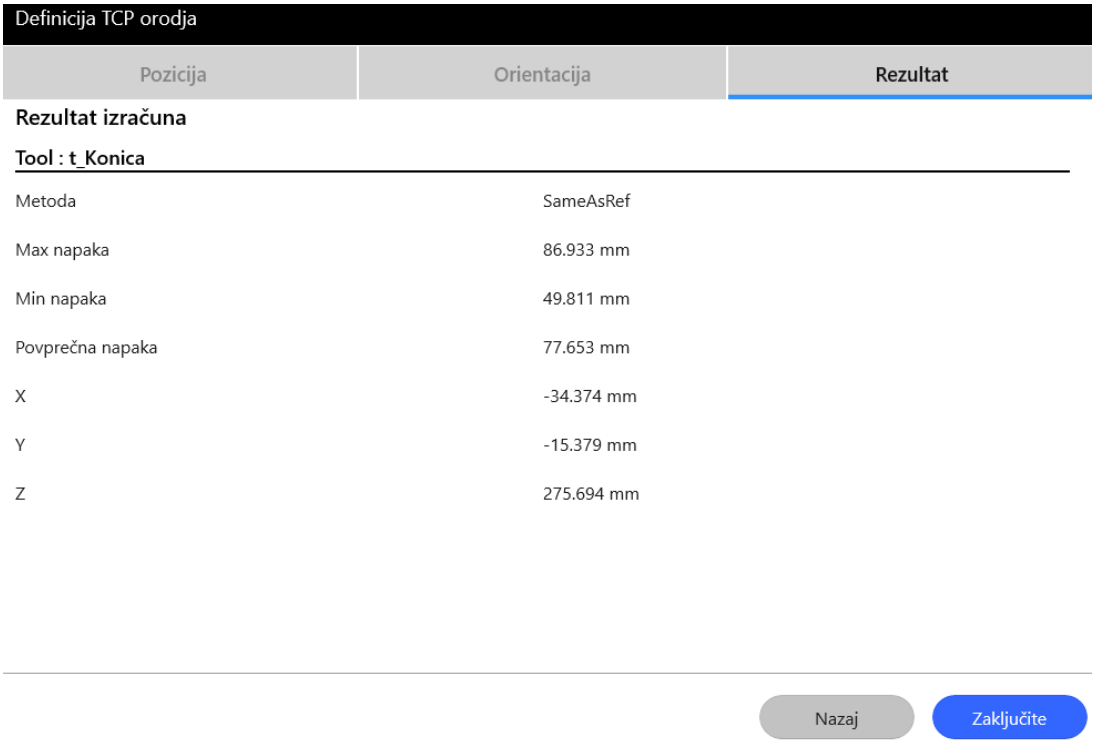
- Ko so vse točke spremenjene, izberite ukaz Naprej.

Slika 115: Spremenjene točke



- Odpre se zavihek Orientacija, v katerem potrdimo orientacijo z izbiro ukaza Naprej.
- Odpre se zavihek Rezultat, v katerem nam izpiše rezultat izračuna. Potrdite ga z izbiro ukaza Zaključi.

Slika 116: Rezultat izračuna



- Izpiše se element t_Konica z novimi podatki.

Slika 117: Nove vrednosti TCP-ja

t_Konica
[TRUE,[[-34.374,-15.379,275.694],[1,0,0,0]],[1.5,[0,0,0],[1,0,0,0],0,0,0]]

- Narejen TCP se shrani tudi v Rapid-u, v modolu, ki smo ga določili pri deklaraciji Tooldata.

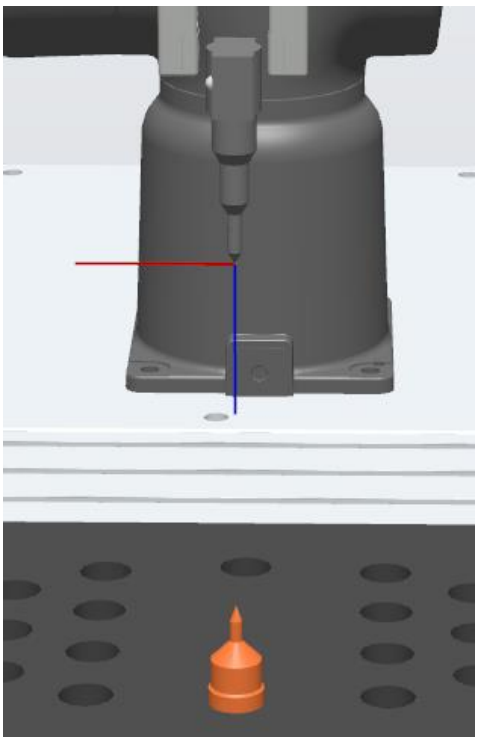
Slika 118: TCP v Rapidu

```
PERS tooldata t_Konica := [TRUE,[[-34.37399,-15.379,275.694],[1,0,0,0]],[1.5,[0,0,0],[1,0,0,0],0,0,0]];
```

- Da bo TCP viden tudi v robotski celici, moramo sinhronizirati podatke za orodje iz Rapida v robotsko celico.

Slika 119: Sinhronizacija orodja

Slika 120: Ustvarjen TCP v robotski celici



3.7 VAJA 6

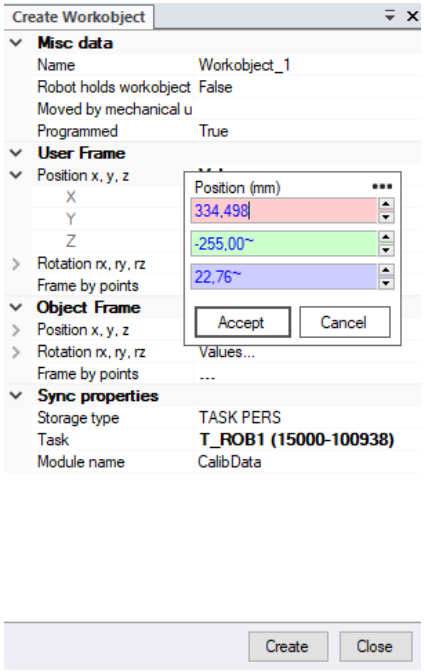
Nastavitev delovnega koordinatnega sistema.

Nastavite WorkObject za komponente v ravnem položaju in na poševnini.

Ustvarjanje WorkObject

1. V zavihku Home izberemo Other in nato Create WorkObject.
2. Določimo mu ime in izhodišče. To lahko storite na dva načina.
   - Z direktnim klikom na točko izhodišča s pomočjo orodja Snap Object.
   - V vrednosti za User Frame se postavimo na rdeče mesto za vrednost in stisnemo na točko. Točka se nam bo shranila po vseh treh oseh.

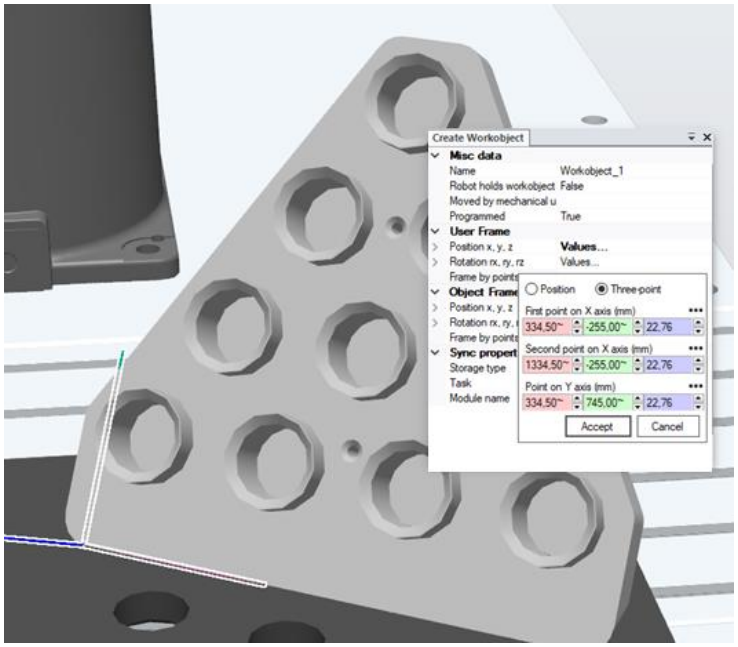
Slika 121: Ustvarjanje WorkObject s klikom na točko



- Z metodo treh točk. Ta metoda je enaka tisti na samem robotu. V vrednosti User Frame izberemo Frame by points in nato Three-point.

- V prvo rdeče polje postavimo točko, kjer bo izhodišče.
- V drugo se premaknemo samo po osi X in zabeležimo točko.
- Za vnos podatkov v tretjo polje se premaknemo po osi Y.

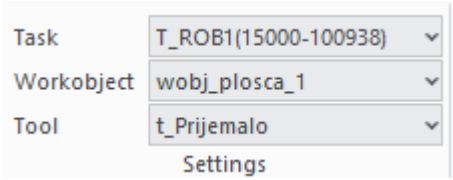
Slika 122: Ustvarjanje WorkObjet z metodo treh točk



3. Ko smo zadovoljni z vnešenimi podatki, jih zabeležite s Create.
4. V zavihku Paths&Targets lahko vidite ustvarjen WorkObject.
5. Metoda treh točk pride še posebej prav, ko je komponenta, katerega uporabljamo v programu, v drugi ravnini.
6. Vsak predmet naj ima svoj WorkObject, program pa svoj uporabniški koordinatni sistem, kar omogoča enostavnejše delo.

Med programiranjem in označevanjem točk je potrebno stalno imeti izbrani pravilni Tool, zato da program ve, kateri TCP upošteva in WorkObject, zato da program ve, kam si točke shranjuje.

Slika 123: Nastavitev izbir



3.8 VAJA 7

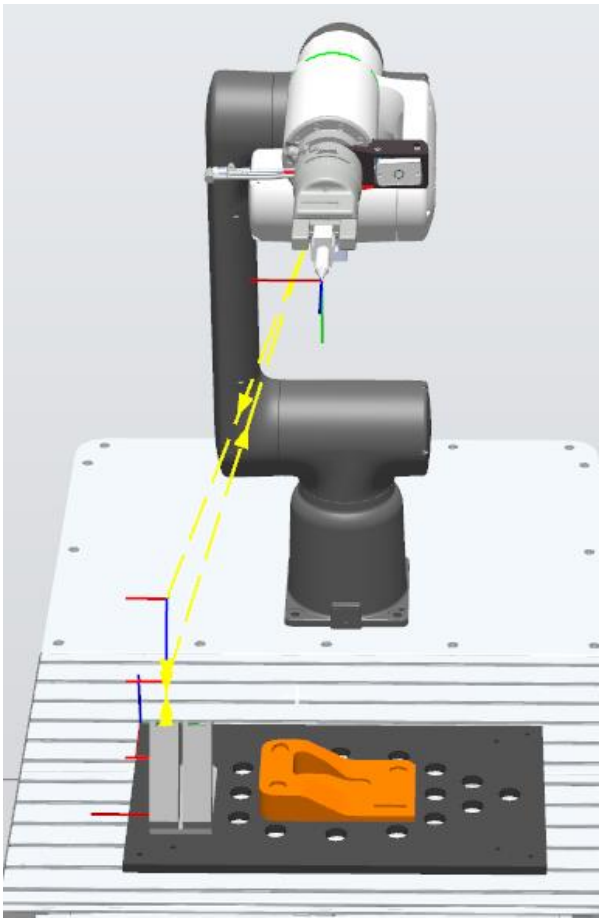
Sledenje 3D konturi

Robotska roka začne program v začetni poziciji. Pomakne se proti držalu orodja, kjer pobere varilno pištolo. Dvigne jo iz pobiralnega mesta in premakne do pozicije nad prvo točko sledenja oblike. Vrstni red sledenja naj bo izveden tako, da prvo sledi najenostavnejšim likom in napreduje proti zahtevnejšim (pravokotnik, šestkotnik, krog, elipsa, notranji rob in na koncu še zunanji rob). Ko konča s sledenjem, orodje vrne nazaj v držalo orodja. Kjer je vogal, se mora robotska roba ustaviti, v kolikor pa vogala ni, se mora konstanto premikati. Vsaki obliki ustvarite točko približevanja in vračanja. V njej se bo robotska roka odvrtela v začetno rotacijo in iz nje šla na naslednjo obliko. Za pobiranje in odlaganje valja uporabite že narejena podprograma.

Dodajanje komponent

1. S funkcijo Import Geometry dodajte 3D_kontura.SAT. Postavite jo na sredino delovne plošče.
2. Ustvarite nov WorkObject v kotu delovne plošče in ga poimenujte wobj_3D_kontura.
3. Preverite, da ima izbran pravilen WorkObject in Tool za izdelavo vaje.

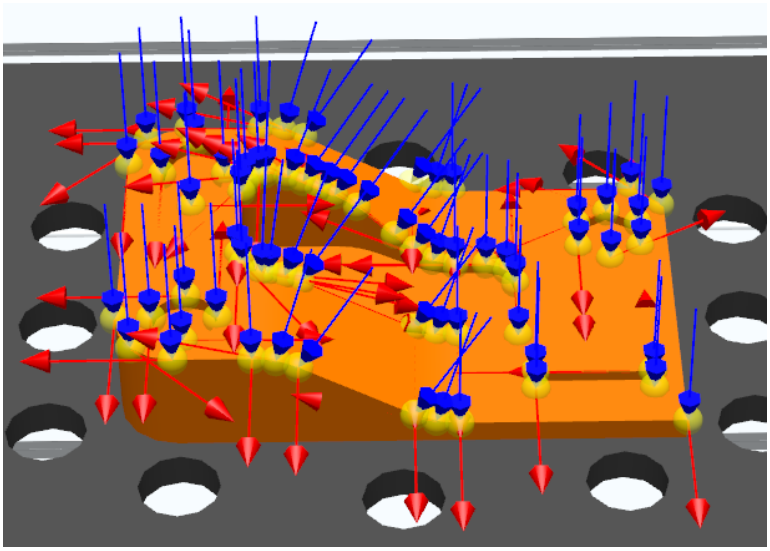
Slika 124: Postavitev v robotski celici za Vajo 7



Ustvarjanje točk

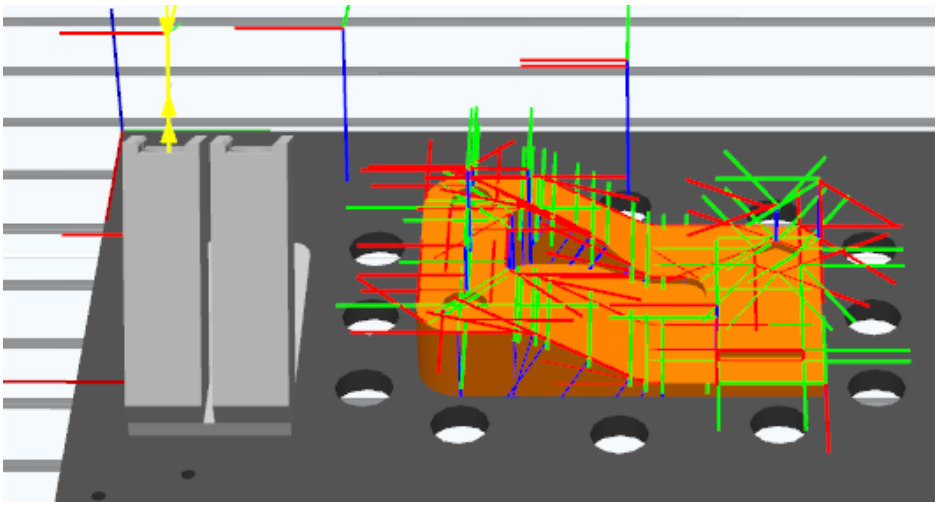
1. Ustvarite točke na konturi in jih poimenujte. Lahko tudi vsako obliko posebej. Točkam uredite orientacijo. Kjer je to potrebno, točko podvojite in ji spremenite rotacijo.

Slika 125: Ustvarjanje točk za Vajo 7



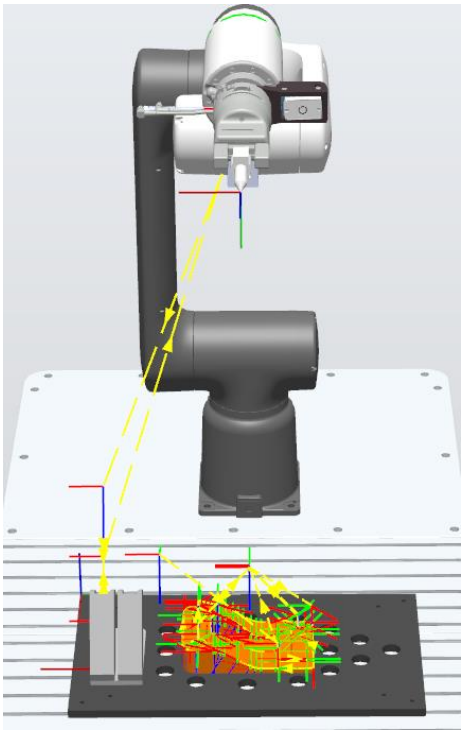
2. Vsaki obliki in robu dodajte točko približevanja konturi in umika. Da bo lažje sistemsko testirati program, dodajte še za vsako obliko in robom eno točko nekje nad konturo.

Slika 126: Ustvarjanje točk za Vajo 7



5. Iz točk po vrsti ustvarite pot gibanja tako, da bo vsaka oblika shranjena v svojem podprogramu.

Slika 127: Ustvarjena pot Vaje 7



Programiranje v Rapidu

1. V podprograme dodajte komentarje in ukaze za odpiranje ter zapiranje prijemala. Nastavite hitrost sledenja konturi in kako se v katerem koraku točki približa. V podprogramu s komentarjem označite vsako obliko.

Slika 128: Vaja 7 – pravokotnik

```
! Pravokotnik
MoveJ APPP1,v500,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveJ k3D_2s_32,v500,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_3p_1,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_3p_2,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_3p_3,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_3p_4,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_3p_5,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_3p_6,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_3p_7,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_3p_8,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_3p_18,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveJ k3D_3p_9,v500,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
```

Slika 129: Vaja 7 – šest kotnik

```
! Heksagon
MoveJ k3D_3p_19,v500,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_4h_1,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_4h_2,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_4h_3,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_4h_4,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_4h_5,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_4h_6,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_4h_7,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_4h_8,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_4h_9,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_4h_10,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_4h_11,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_4h_12,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_4h_22,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveJ k3D_4h_13,v500,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
```

Slika 130: Vaja 7- krog

```
! Krog
MoveJ k3D_4h_23,v500,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_5k_1,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_5k_2,k3D_5k_3,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_5k_4,k3D_5k_5,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_5k_5,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_5k_15,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveJ k3D_5k_6,v500,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
```

Slika 131: Vaja 7 - elipsa

```
! Elipsa
MoveJ k3D_5k_16,v500,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_6e_1,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_6e_2,k3D_6e_3,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_6e_4,k3D_6e_5,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_6e_5,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_6e_15,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveJ k3D_5k_6,v500,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
```

Slika 132: Vaja 7 - sredinski rob

```
! Sredinski rob
MoveJ k3D_1r_44, v500, z10, t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_1,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_2,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_2s_3,k3D_2s_4,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_5,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_2s_6,k3D_2s_7,v150,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_8,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_2s_9,k3D_2s_10,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_11,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_12,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_2s_13,k3D_2s_14,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_15,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_16,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_17,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_2s_18,k3D_2s_19,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_20,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_2s_21,k3D_2s_22,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_23,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_2s_24,k3D_2s_25,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_26,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_2s_27,k3D_2s_28,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_28,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_2s_42,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveJ k3D_2s_29,v500,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
```

Slika 133: Vaja 7 - zunanji rob

```
! Rob
MoveJ k3D_1r_31,v500,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_1,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_2,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_3,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_4,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_1r_5,k3D_1r_6,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_7,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_8,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_1r_9,k3D_1r_10,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_11,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_1r_12,k3D_1r_13,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_14,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_1r_15,k3D_1r_16,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_17,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_1r_18,k3D_1r_19,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_20,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_21,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveC k3D_1r_22,k3D_1r_23,v20,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_24,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_25,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveL k3D_1r_26,v20,fine,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
MoveJ k3D_1r_34,v500,z10,t_Varilna_Pistola\WObj:=wobj_3D_kontura;
```

2. Ko so podprogrami urejeni, jih zapišete v glavnega in sinhronizirajte Rapid z robotsko celico.

Slika 134: Klic podprogramov v glavnega za Vajo 7

```
! Klic Welding 3D
p_Drzalo_orodja;
p_Welding_3D;
p_Welding_3D_2;
p_Welding_3D_3;
p_Welding_3D_4;
p_Welding_3D_5;
p_Welding_3D_6,
p_drzalo_orodjak;
WaitTime 1;
```

Simulacija vaje

1. V robotski celice preverite delovanje simulacije. Postopek najprej zaženemo v koračnem načinu, nato še v avtomatskem načinu.
2. Ob pravilnem gibanju robotske roke je vaja končana.

3.9 VAJA 8A

Pobiranje in odlaganje treh valjev preko ovire.

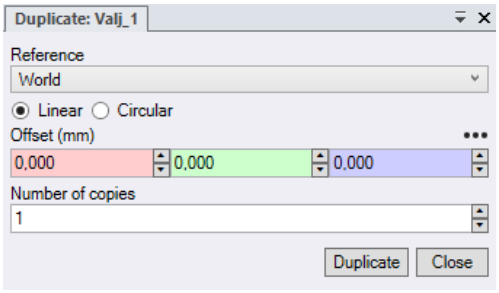
Ustvarite program, v katerem bo robotska roka valje iz ene strani ovire prestavila na drugo stran na isto mesto. Program se mora začeti in končati v začetni poziciji. Vsebovati mora točko približevanja valju, točko pobiranja, točko dviga, točko nad oviro, točko približanja odlagalnemu mestu in točko, v kateri se dvigne nad odlagalno mesto. Te točke ustvarite za vse tri valje. V Rapid-u dodajte še zapiranje in odpiranje prijemala ter uredite gibe.

Ustvarite nov WorkObject, točke programa, pot gibanja in program.

Dodajanje komponent

1. Skrijte vse komponente, ki jih ne potrebujete (3D kontura, držalo orodja, varilna pištola) in prikažite vse, ki jih (ovira, valj 1, 2, 3).
2. Dodatna valja ustvarite s kopiranjem prvega in ju postavite na ustrezni lokaciji.
    - Z desnim klikom na Valj 1 izberite funkcijo Duplicate in v pojavnem oknu potrdite z ukazom Duplicate.

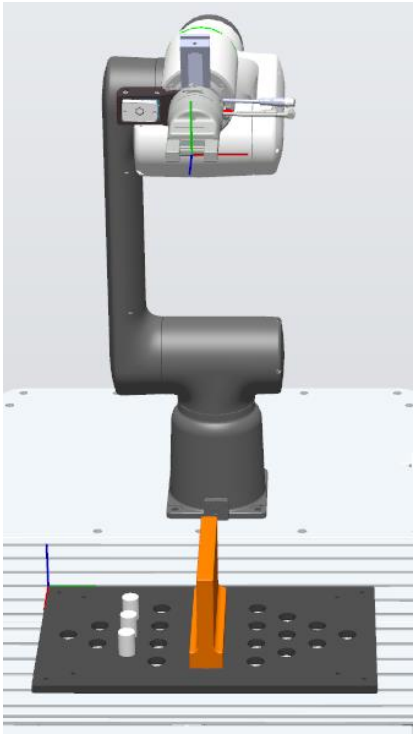
Slika 135: Dodajanje valjev



    - Nov valj je postavljen na enako mesto kot že obstoječi, zato ga s funkcijo Move and Rotate premaknemo na želeno pozicijo. V zavihku Layout ga poimenujte.
    - Enako ponovite za tretji valj.

3. Ustvarite nov WorkObject v vogalu delovne plošče.
4. Preverite, da ima izbran pravilen WorkObject in Tool za izdelavo vaje.
5. Robotsko roko postavite v položaj jt_Home.

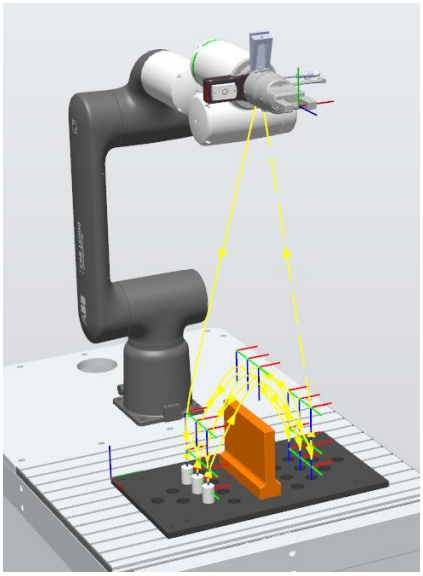
Slika 136: Postavitev robotske celice za Vajo 8a



Ustvarjanje točk pobiranja in odlaganja

1. Za vsak valj ustvarite potrebne točke in jih poimenujte. Vsak valj naj ima 7 točk in nato še točko, v kateri se premakne nad oviro pred pobiranjem naslednjega valja.
2. Ustvarite novo pot, jo poimenujte in v njo dodajte ustvarjene točke in Home pozicijo.
3. Nato robotsko celico sinhronizirajte v Rapid.

Slika 137: Točke in pot gibanja za Vajo 8a



Programiranje v Rapidu

1. V podprogramu dodajte komentarje in ukaze za odpiranje ter zapiranje prijemala. Nastavite tudi hitrost premikanja robotske roke.

Slika 138: Del podprograma za Vajo 8a

```
 ! Zacetek programa za 3 valje
PROC Pick_and_place_3()
    ! Zacetna pozicija
    MoveAbsJ jt_Home,v1000,z100,t_Prijemalo\WObj:=wobj_plosca_3;
    ! Premik nad valj 1
    MoveJ v1_1_Pick_Above,v300,z100,t_Prijemalo\WObj:=wobj_plosca_3;
    ! Spust do valja
    MoveL v1_2_Pick,v300,z100,t_Prijemalo\WObj:=wobj_plosca_3;
    ! Zapri prijemalo
    closeGripper;
    WaitTime 1;
    ! Dvig valja
    MoveL v1_3_Pick_Up,v200,z100,t_Prijemalo\WObj:=wobj_plosca_3;
    ! Pomik nad oviro
    MoveJ v1_4_Over_Obstacle,v200,z100,t_Prijemalo\WObj:=wobj_plosca_3;
    ! Pomik nad odlagalno mesto
    MoveJ v1_5_Place_Above,v200,z100,t_Prijemalo\WObj:=wobj_plosca_3;
    ! Spust valja
    MoveL v1_6_Place,v200,z100,t_Prijemalo\WObj:=wobj_plosca_3;
    ! Odpri prijemalo
    openGripper;
    WaitTime 1;
    ! Dvig prijemala
    MoveL v1_7_Place_Up,v300,z100,t_Prijemalo\WObj:=wobj_plosca_3;
    ! Pomik nad oviro
    MoveJ v1_8_Over_Obstacle,v300,z100,t_Prijemalo\WObj:=wobj_plosca_3;

    ! Premik nad valj 2
    MoveJ v2_1_Pick_Above,v300,z100,t_Prijemalo\WObj:=wobj_plosca_3;
    ! Spust do valja
    MoveL v2_2_Pick,v300,z100,t_Prijemalo\WObj:=wobj_plosca_3;
```

2. Po vzoru premika za valj 1 naredite še to za valj 2 in 3.
3. Ko je podprogram urejen, ga zapišite v glavnega in sinhronizirajte Rapid z robotsko celico.

Slika 139: Klic podprograma v glavnega za Vajo 8a

```
!Klic pick and place 3
Pick_and_place_3;
WaitTime 1;
```

Simulacija vaje

1. V robotski celice preverite delovanje simulacije. Postopek najprej zaženemo v koračnem načinu, nato še v avtomatskem načinu.
2. Ob pravilnem gibanju robotske roke je vaja končana.

3.10 VAJA 8B

Pobiranje in odlaganje treh valjev s pomočjo ukaza Offset.

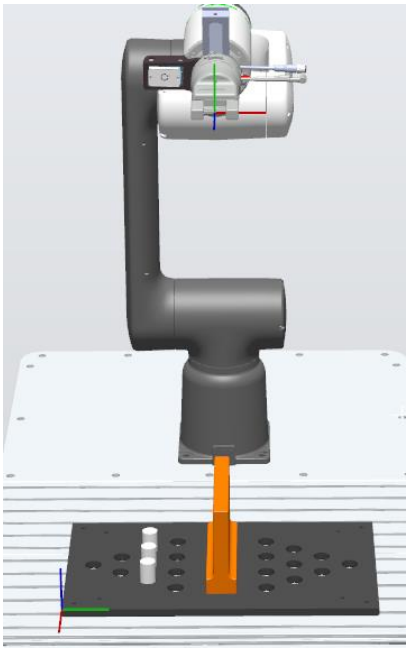
Ustvarite program, v katerem bo robotska roka valje iz ene strani ovire prestavila na drugo stran na isto mesto. Program se mora začeti in končati v začetni poziciji. Vsak valj mora vsebovati točke za približevanje valju, točko pobiranja, točko vračanja, točko nad oviro, točko približanja odlagalnemu mestu in točko, v kateri se vrne nad odlagalno mesto. V Rapid-u dodajte še zapiranje in odpiranje prijemala ter uredite gibe.

Ustvarite nov WorkObject in v njem shranite točko, v kateri prijemalo valj pobere. Vse ostale točke boste ustvarili z zamikom začetne točke.

Dodajanje komponent

1. Ustvarite nov WorkObject v kotu delovne plošče in ga poimenujte wobj_plosca_3.
2. V nastavitvah v zavihku Home preverite, da imate izbran pravilen Tool in WorkObject.
3. Robotsko roko postavite v položaj jt_Home.

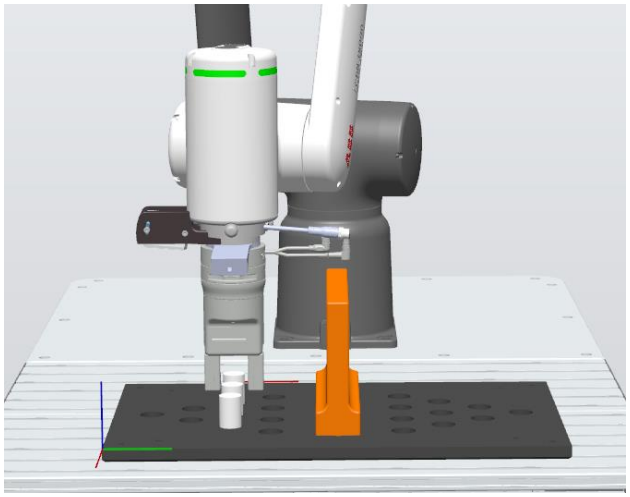
Slika 140: Postavitev robotske celice za Vajo 8b



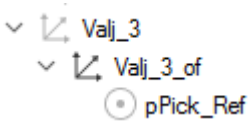
Ustvarjanje točke

1. TCP postavite v točko, v kateri prijemalo pobere valj pri Vaji 2. Izberite točko in nato View Robot at Target.
2. Z ukazom Create Target ustvarite novo točko, ki bo v trenutnem položaju TCP-ja in jo poimenujte.

Slika 141: Ustvarjena točka za Vajo 8b



Slika 142: Shranjena referenčna točka



3. Ustvarjeno točko sinhronizirajte v Rapid.

Programiranje v Rapidu

1. V Module1, med shranjenimi točkami, spremenite podatkovni tip kreirane točke iz CONST v PERS.
   Ta ukaz omogoči, da se referenčna točka shrani kot konstanta, glede na katero se ustvarjajo premiki.

Slika 143: Sprememba podatkovnega tipa točke

```
PERS robtarget pPick_Ref :=
```

2. Pod točko dodajte konstante in spremenljive, s katerimi boste nastavljali pozicije preostalih točk oziroma zamikov. Če kateri izmed premikov ni ustrezen, prilagodite vrednost tega premika v tem delu.

Slika 144: Konstante in spremenljivke za premike

```
PERS robtarget pPick_Ref := [[96.129,134.829271

CONST num yPlace       := 240.33;
CONST num zAppPick     := 130;
CONST num zAppPlace    := 130;
CONST num zObstacle    := 250;
CONST num yObstacle    := 120;
CONST num zPickUp      := 80;
CONST num zPlaceUp     := 80;

CONST num dxValji{3}   := [0, 53, 106];
CONST num xPlace       := -300;

VAR num i;
VAR num dx;
```

3. Na začetku podprograma vstavite ukaz za Home pozicijo. Ustvarite FOR zanko. Ta bo določila, nad katerim valjem se bo izvajal premik.

Slika 145: Zanka FOR za Vajo 8b

```
! Program za racunanje premika za 3 valje
PROC p_Valj_3()
        ! Zacetna pozicija
        MoveAbsJ jt_Home,v1000,z100,t_Prijemalo\WObj:=Valj_3;
   FOR i FROM 1 TO 3 DO
        dx := dxValji{i};
```

4. Obrazložitev ukaza Offset.

Slika 146: Ukaz Offset

```
MoveJ Offs(pPick_Ref, dx + xPlace, yObstacle, zObstacle),
```

- pPick_Ref – referenčna točka, zamiki premikov se računajo glede na pozicijo te točke.
- dx + xPlace – premik po osi X.
- yObstacle – premik po osi Y.
- zObstacle – premik po osi Z.

5. Ustvarite premike za točke približevanja valju, pobiranje valja, dvig valja, premik čez oviro, približevanju mesta odlaganja, spust valja, dvig prijemala in še enkrat čez oviro.
6. Podprogramu dodajte ukaza za zapiranje in odpiranje prijemala ter komentarje za vsak ukaz.
7. Ko se FOR zanka konča, se robotska roka vrne v Home pozicijo.

Slika 147: Podprogram za Vajo 8b

```
! Program za racunanje premika za 3 valje
PROC p_Valj_3()
        ! Zacetna pozicija
        MoveAbsJ jt_Home,v1000,z100,t_Prijemalo\WObj:=Valj_3;
   FOR i FROM 1 TO 3 DO
        dx := dxValji{i};
        !PickAbove
        MoveJ Offs(pPick_Ref, dx + xPlace, 0, zAppPick), v300, z50, t_Prijemalo \WObj:=Valj_3;
        !Pick
        MoveL Offs(pPick_Ref, dx + xPlace, 0, 0), v300, fine, t_Prijemalo \WObj:=Valj_3;
        !Zapre prijemalo
        closeGripper;
        WaitTime 0.5;
        !PickUp
        MoveL Offs(pPick_Ref, dx + xPlace, 0, zPickUp), v200, z50, t_Prijemalo \WObj:=Valj_3;
        !OverObsatcle
        MoveJ Offs(pPick_Ref, dx + xPlace, yObstacle, zObstacle), v200, z50, t_Prijemalo \WObj:=Valj_3;
        !PlaceAbove
        MoveJ Offs(pPick_Ref, dx + xPlace, yPlace, zAppPlace), v200, z50, t_Prijemalo \WObj:=Valj_3;
        !Place
        MoveL Offs(pPick_Ref, dx + xPlace, yPlace, 0), v200, fine, t_Prijemalo \WObj:=Valj_3;
        !Odpre prijemalo
        openGripper;
        WaitTime 0.5;
        !PlaceUp
        MoveL Offs(pPick_Ref, dx + xPlace, yPlace, zPlaceUp), v300, z50, t_Prijemalo \WObj:=Valj_3;
        !OverObsatcle
        MoveJ Offs(pPick_Ref, dx + xPlace, yObstacle, zObstacle), v300, z50, t_Prijemalo \WObj:=Valj_3;
    ENDFOR
        ! Zacetna pozicija
        MoveAbsJ jt_Home,v1000,z100,t_Prijemalo\WObj:=Valj_3;
ENDPROC
```

8. Ko je podprogram urejen, ga zapišite v glavnega in sinhronizirajte Rapid z robotsko celico.

Slika 148: Klic podprograma v glavnega za Vajo 8b

```
! Klic programa za 3 valje z racunanjem
p_Valj_3;
WaitTime 1;
```

Simulacija vaje

1. V robotski celice preverite delovanje simulacije. Postopek najprej zaženemo v koračnem načinu, nato še v avtomatskem načinu. Prikazana bo samo ena točka, ostale pozicije so izračunane.
2. Ob pravilnem gibanju robotske je vaja končana.

3.11 VAJA 9A

Pobiranje in odlaganje desetih valjev preko ovire.

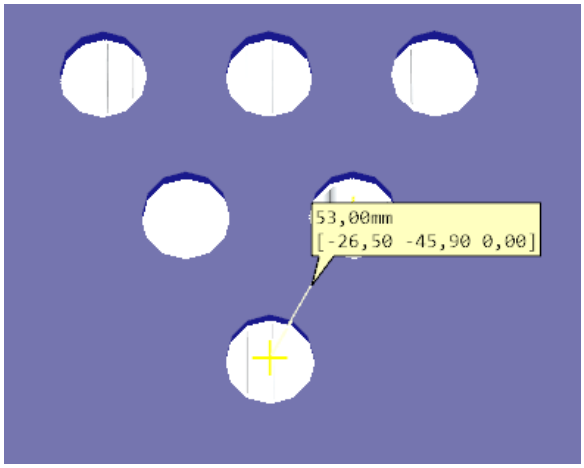
Ustvarite program, v katerem bo robotska roka valje iz ene strani ovire prestavila na drugo stran na isto mesto. Program se mora začeti in končati v začetni poziciji. Vsebovati mora točko približevanja valju, točko pobiranja, točko vračanja, točko nad oviro, točko približanja odlagalnemu mestu in točko vračanja. Te točke ustvarite za vseh deset valjev. V Rapid-u dodajte še zapiranje in odpiranje prijemala ter uredite gibe.

Ustvarite nov WorkObject, točke programa, pot gibanja in program.

Dodajanje komponent

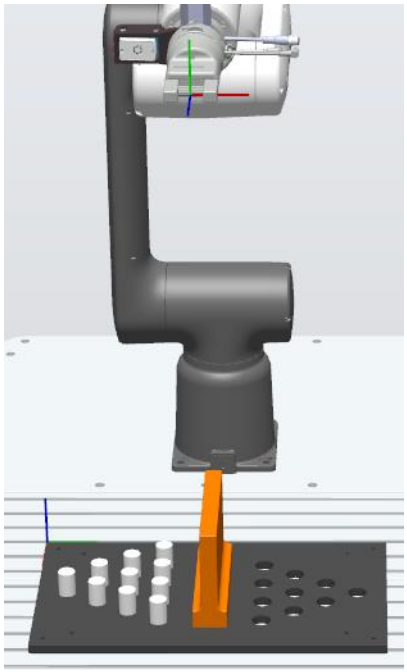
1. Ustvarite nov WorkObject v kotu delovne plošče in ga poimenujte wobj_plosca_10.
2. Dodatne valje, ustvarite s kopiranjem že obstoječih, jih poimenujte in postavite na ustrezne lokacije. Razdaljo med pobiralnimi mesti si lahko izmerite s pomožnim orodjem Measure.

Slika 149: Merjenje razdalje pobiralnih mest



3. V nastavitvah v zavihku Home preverite, če imate izbran pravilen Tool in WorkObject.
4. Robotsko roko postavite v položaj jt_Home.

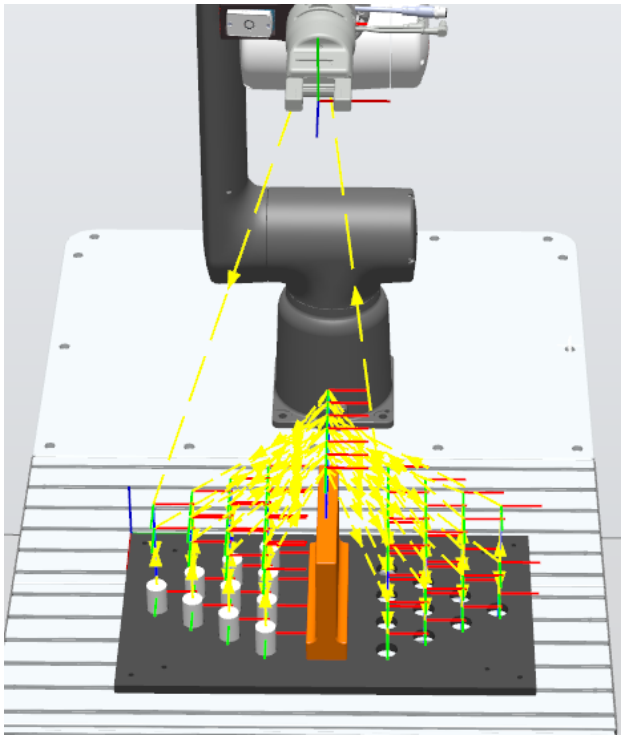
Slika 150: Postavitev robotske celice za Vajo 9a



Ustvarjanje točk pobiranja in odlaganja

1. Za vsak valj ustvarite potrebne točke in jih poimenujte. Vsak valj naj ima 7 točk in nato še točko, v kateri se premakne nad oviro pred pobiranjem naslednjega valja.
2. Ustvarite novo pot, jo poimenujte in v njo dodajte ustvarjene točke in Home pozicijo.
3. Nato robotsko celico sinhronizirajte v Rapid.

Slika 151: Točke in pot gibanja za Vajo 9a



Programiranje v Rapidu

1. V podprogramu dodajte komentarje in ukaze za odpiranje ter zapiranje prijemala. Nastavite tudi hitrost premikanja robotske roke.

Slika 152: Del podprograma za Vajo 9a

```
! Zacetek programa za 10 valjev
PROC Pick_and_place_10()
    ! Zacetna pozicija
    MoveAbsJ jt_Home,v1000,z100,t_Prijemalo\WObj:=wobj_plosca_10;
    ! Premik nad valj 1
    MoveJ d1_1_Pick_Above,v300,z100,t_Prijemalo\WObj:=wobj_plosca_10;
    ! Spust do valja
    MoveL d1_2_Pick,v300,fine,t_Prijemalo\WObj:=wobj_plosca_10;
    ! Zapri prijemalo
    closeGripper;
    WaitTime 1;
    ! Dvig valja
    MoveL d1_3_Pick_Up,v200,z100,t_Prijemalo\WObj:=wobj_plosca_10;
    ! Pomik nad oviro
    MoveJ d1_4_Over_Obstacle,v200,z100,t_Prijemalo\WObj:=wobj_plosca_10;
    ! Pomik nad odlagalno mesto
    MoveJ d1_5_Place_Above,v200,z100,t_Prijemalo\WObj:=wobj_plosca_10;
    ! Spust valja
    MoveL d1_6_Place,v200,fine,t_Prijemalo\WObj:=wobj_plosca_10;
    ! Odpri prijemalo
    openGripper;
    WaitTime 1;
    ! Dvig prijemala
    MoveL d1_7_Place_Up,v300,z100,t_Prijemalo\WObj:=wobj_plosca_10;
    ! Pomik nad oviro
    MoveJ d1_8_Over_Obstacle,v300,z100,t_Prijemalo\WObj:=wobj_plosca_10;
    ! Premik nad valj 2
    MoveJ d2_1_Pick_Above,v300,z100,t_Prijemalo\WObj:=wobj_plosca_10;
    ! Spust do valja
    MoveL d2_2_Pick,v300,fine,t_Prijemalo\WObj:=wobj_plosca_10;
```

2. Po vzoru premika za valj 1 naredite še to za preostalih 9 valjev.
3. Ko je podprogram urejen, ga zapišite v glavnega in sinhronizirajte Rapid z robotsko celico.

Slika 153: Klic podprograma v glavnega za Vajo 9a

```
!Klic pick and place 10
Pick_and_place_10;
WaitTime 1;
```

Simulacija vaje

1. V robotski celice preverite delovanje simulacije. Postopek najprej zaženemo v koračnem načinu, nato še v avtomatskem načinu.
2. Ob pravilnem gibanju robotske roke je vaja končana.

3.12 VAJA 9B

Prenos desetih valjev preko ovire in nato nazaj na prvotno mesto.

Ustvarite program, v katerem bo robotska roka valje iz ene strani ovire prestavila na drugo stran na isto mesto in nato te valje premaknila ponovno na začetno pozicijo. Program se mora začeti in končati v začetni poziciji. Ko zaključite prvi prenos, se postavite v Home pozicijo in nato nadaljujte s prenosom valjev na prvotno mesto.

Iz že narejenih točk, ki so shranjene v wobj_plosca_10, ustvarite novo pot gibanja in jo preimenujte. Uporabite že narejene točke za sestavo programa.

Ustvarjanje poti

1. V zavihku Home izberite ukaz Path in nato Empty Path.
2. Novo pot poimenujte.
3. Sinhronizirajte podatke iz robotske celice v Rapid.

Programiranje v Rapidu

1. V Module1, v zavihku Pick_and_place_10, kopirajte program in ga prilepite v novo pot.
2. Uredite vrstni red poteka programa tako, da bo ustrezal zahtevam vaje.
3. Popravite tudi hitrost giba in pot gibanja tam, kjer je to potrebno.
4. Vsak korak komentirajte.

Slika 154: Del programa za Vajo 9b

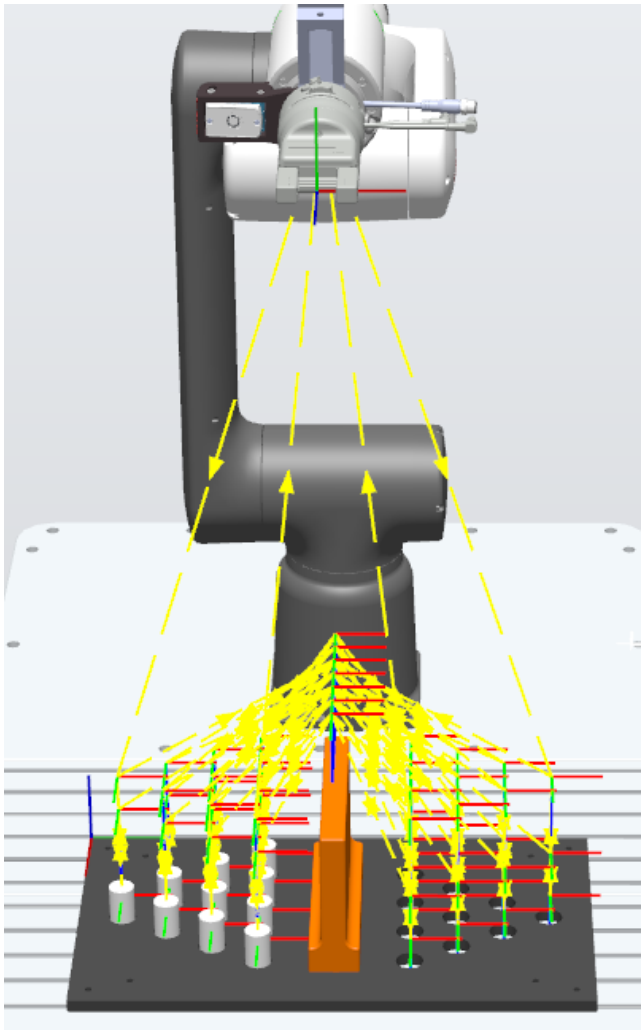
```
! Zacetek pobiranja na drugi strani ovire

! Premik nad valj 1
MoveJ d1_7_Place_Up,v300,z100,t_Prijemalo\WObj:=wobj_plosca_10;
! Spust do valja
MoveL d1_6_Place,v300,fine,t_Prijemalo\WObj:=wobj_plosca_10;
! Zapri prijemalo
closeGripper;
WaitTime 1;
! Dvig valja
MoveL d1_5_Place_Above,v200,z100,t_Prijemalo\WObj:=wobj_plosca_10;
! Pomik nad oviro
MoveJ d1_4_Over_Obstacle,v200,z100,t_Prijemalo\WObj:=wobj_plosca_10;
! Pomik nad odlagalno mesto
MoveJ d1_3_Pick_Up,v200,z100,t_Prijemalo\WObj:=wobj_plosca_10;
! Spust valja
MoveL d1_2_Pick,v200,fine,t_Prijemalo\WObj:=wobj_plosca_10;
! Odpri prijemalo
openGripper;
WaitTime 1;
! Dvig prijemala
MoveL d1_1_Pick_Above,v300,z100,t_Prijemalo\WObj:=wobj_plosca_10;
! Pomik nad oviro
MoveJ d1_8_Over_Obstacle,v300,z100,t_Prijemalo\WObj:=wobj_plosca_10;
! Premik nad valj 2
MoveJ d2_7_Place_Up,v300,z100,t_Prijemalo\WObj:=wobj_plosca_10;
! Spust do valja
MoveL d2_6_Place,v300,fine,t_Prijemalo\WObj:=wobj_plosca_10;
```

5. Po vzoru premika za valj 1 naredite še to za preostalih 9 valjev.
6. Ko je podprogram urejen, ga zapišite v glavnega in sinhronizirajte Rapid z robotsko celico.

Slika 155: Klic podprograma v glavnega za Vajo 9b

```
!Klic pick and place 10_1
Pick_and_place_10_1;
WaitTime 1;
```

Slika 156: Točke in pot gibanja za Vajo 9b



Simulacija vaje

1. V robotski celice preverite delovanje simulacije. Postopek najprej zaženemo v koračnem načinu, nato še v avtomatskem načinu.
2. Ob pravilnem gibanju robotske roke je vaja končana.

3.13 VAJA 10A

Pobiranje in odlaganje desetih valjev na poševnino.

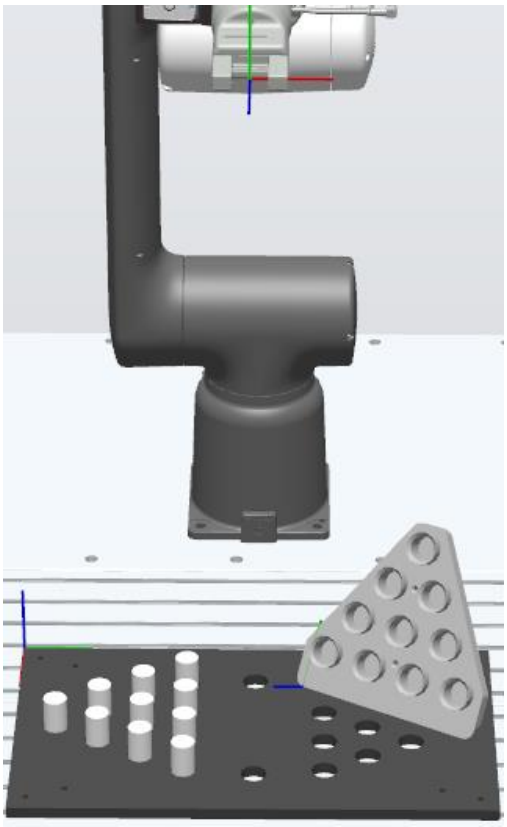
Ustvarite program, v katerem bo robotska roka valje iz ene strani prestavila na drugo stran na isto mesto na poševnini. Program se mora začeti in končati v začetni poziciji. Vsebovati mora točko približevanja valju, točko pobiranja, točko vračanja, točko nad sredino delovne plošče, točko približanja odlagalnemu mestu in točko vračanja. Te točke ustvarite za vseh deset valjev. V Rapid-u dodajte še zapiranje in odpiranje prijemala ter uredite gibe.

Ustvarite nov WorkObject za pobiranje valjev na delovni plošči.
Ustvarite nov WorkObject za odlaganje valjev na poševnini.
Ustvarite točke programa, pot gibanja in program.

Dodajanje komponent

1. S funkcijo Import Geometry dodajte poševnino. Postavite jo na delovno ploščo.
2. Ustvarite nov WorkObject v kotu delovne plošče in ga poimenujte wobj_plosca_posevnina.
3. Ustvarite nov WorkObject v kotu delovne plošče in ga poimenujte wobj_posevna_plosca.
4. V nastavitvah v zavihku Home preverite, da imate izbran pravilen Tool in WorkObject, ko dodajate točke na ploščo ali poševnino.
5. Robotsko roko postavite v položaj jt_Home.

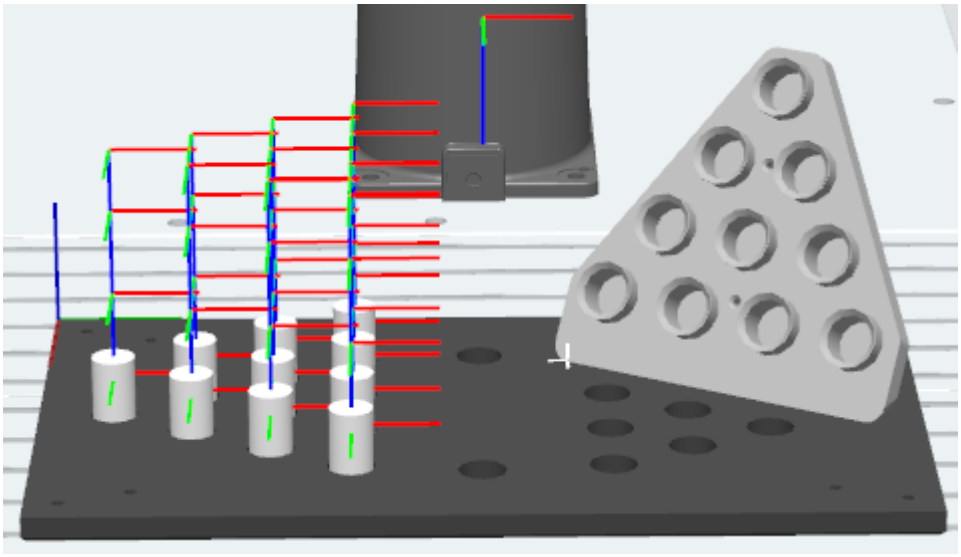
Slika 157: Postavitev robotske celice za Vajo 10a



Ustvarjanje točk pobiranja in odlaganja

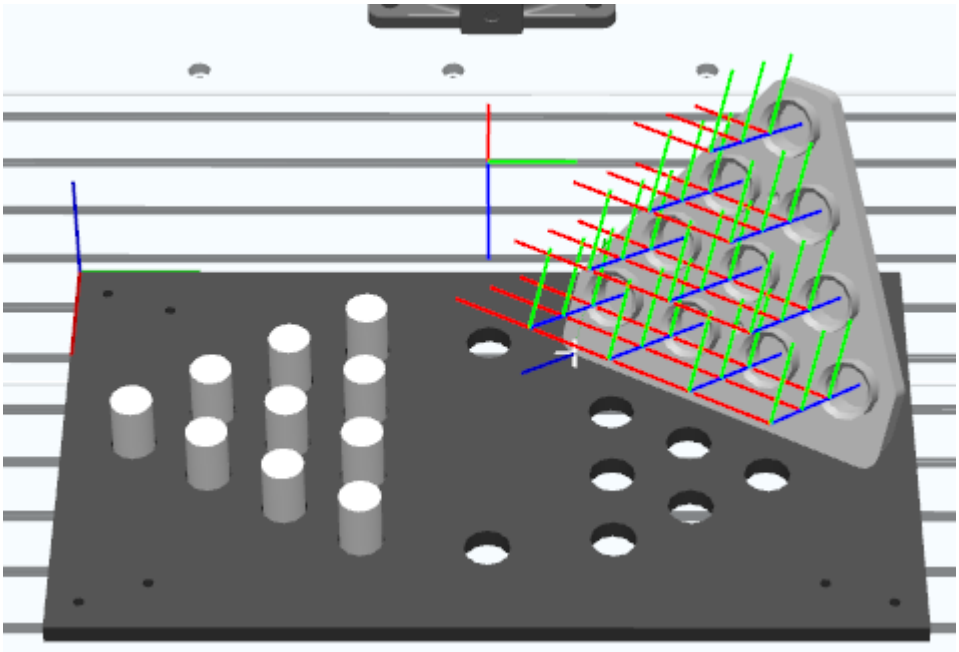
4. Za vsak valj ustvarite potrebne točke in jih poimenujte. Točke za pobiranje naj bodo v enem WorkObject-u, za odlaganje pa v drugem. Med obema WorkObjectom-a naj bo vmesna točka, ki je v isti poziciji, vendar v drugi orientaciji.

Slika 158: Ustvarjene točke v enem WorkObject-u



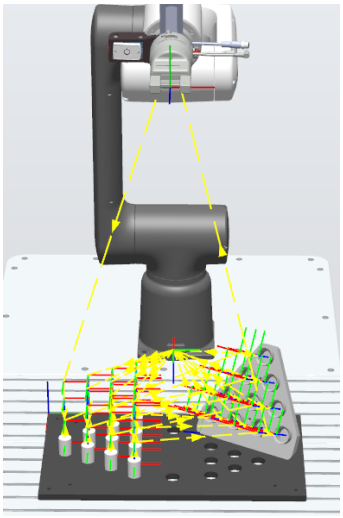
5. Pri ustvarjanju točk na poševnini si pomagajte z nastavitvijo reference na aktivni WorkObject. Pomagajte si s pomožnim orodjem Measure, da izmerite razdaljo med trenutno lokacijo TCP-ja in lokacijo, kjer želite ustvariti točko. Nato se z ukazom Jog TCP za izmerjeno vrednost premaknite. S tem boste lahko bolj natančni pri določitvi točk.

Slika 159: Ustvarjene točke v drugem WorkObject-u



6. Ustvarite eno pot, jo poimenujte in v njo dodajte ustvarjene točke iz obeh WorkObject-ov ter Home pozicijo. Točke dodajte v pot po vrsti za vsak valj, da ne bo prevelike zmede v Rapidu.

Slika 160: Ustvarjena pot za Vajo 10a



7. Nato robotsko celico sinhronizirajte v Rapid.

Programiranje v Rapidu

1. V podprogramu dodajte komentarje in ukaze za odpiranje ter zapiranje prijemala. Določite, kateri gib v programu bo linearen in kateri osni. Nastavite tudi hitrost premikanja robotske roke.

Slika 161: Del programa za Vajo 10a

```
! Zacetek programa za 10 valjev na posevnino
PROC Pick_and_place_10_posevnina()
    ! Zacetna pozicija
    MoveAbsJ jt_Home,v1000,z100,t_Prijemalo\WObj:=wobj_posevna_plosca;
    ! Premik nad valj 1
    MoveJ k1_1_Pick_Above,v300,z100,t_Prijemalo\WObj:=wobj_plosca_posevnina;
    ! Spust do valja
    MoveL k1_2_Pick,v300,fine,t_Prijemalo\WObj:=wobj_plosca_posevnina;
    ! Zapri prijemalo
    closeGripper;
    WaitTime 1;
    ! Dvig valja
    MoveL k1_3_Pick_Up,v200,z100,t_Prijemalo\WObj:=wobj_plosca_posevnina;
    ! Position
    MoveJ k1_4_Position,v200,z100,t_Prijemalo\WObj:=wobj_plosca_posevnina;
    ! Pomik nad odlagalno mesto
    MoveJ k1_5_Place_Above,v200,z100,t_Prijemalo\WObj:=wobj_posevna_plosca;
    ! Spust valja
    MoveL k1_6_Place,v200,fine,t_Prijemalo\WObj:=wobj_posevna_plosca;
    ! Odpri prijemalo
    openGripper;
    WaitTime 1;
    ! Dvig prijemala
    MoveL k1_7_Place_Up,v300,z100,t_Prijemalo\WObj:=wobj_posevna_plosca;
    ! Pomik na position
    MoveJ k1_8_Position,v300,z100,t_Prijemalo\WObj:=wobj_posevna_plosca;
    ! Premik nad valj 2
    MoveJ k2_1_Pick_Above,v300,z100,t_Prijemalo\WObj:=wobj_plosca_posevnina;
    ! Spust do valja
    MoveL k2_2_Pick,v300,fine,t_Prijemalo\WObj:=wobj_plosca_posevnina;
```

2. Po vzoru premika za valj 1 naredite še to za preostalih 9 valjev.
3. Ko je podprogram urejen, ga zapišite v glavnega in sinhronizirajte Rapid z robotsko celico.

Slika 162: Klic podprograma v glavnega za Vajo 10a

```
!Klic pick and place 10 posevnina
Pick_and_place_10_posevnina;
WaitTime 1;
```

Simulacija vaje

1. V robotski celice preverite delovanje simulacije. Postopek najprej zaženemo v koračnem načinu, nato še v avtomatskem načinu.
2. Ob pravilnem gibanju robotske roke je vaja končana.

3.14 VAJA 10B

Pobiranje in odlaganje desetih valjev na poševnino in nato nazaj na prvotno mesto.

Ustvarite program, v katerem bo robotska roka valje iz ene strani prestavila na drugo stran na isto mesto na poševnini in nato nazaj na prvotno mesto. Program se mora začeti in končati v začetni poziciji. Vsebovati mora točko približevanja valju, točko pobiranja, točko dviga, točko nad sredino delovne plošče, točko približanja odlagalnemu mestu in točko, v kateri se prijemalo dvigne nad odlagalno mesto. Te točke ustvarite za vseh deset valjev. V Rapid-u dodajte še zapiranje in odpiranje prijemala ter uredite gibe.

Ustvarite novo pot in v njej uporabite že narejene točke.

Ustvarjanje poti

1. V zavihku Home izberite ukaz Path in nato Empty Path.
2. Novo pot poimenujte.
3. Sinhronizirajte podatke iz robotske celice v Rapid.

Programiranje v Rapidu

1. V Module1, v zavihku Pick_and_place_10_posevnina, kopirajte program in ga prilepite v novo pot.
2. Uredite vrstni red poteka programa tako, da bo ustrezal zahtevam vaje.
3. Popravite tudi hitrost giba in pot gibanja tam, kjer je to potrebno.
4. Vsak korak komentirajte.

Slika 163: Del programa za Vajo 10b

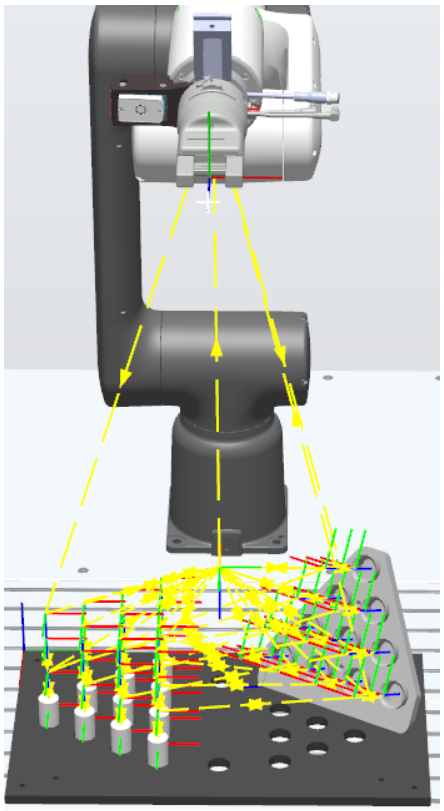
```
! Premik nad valj 10
MoveJ k1_7_Place_Up,v300,z100,t_Prijemalo\WObj:=wobj_posevna_plosca;
! Spust do valja
MoveL k1_6_Place,v200,fine,t_Prijemalo\WObj:=wobj_posevna_plosca;
! Zapri prijemalo
closeGripper;
WaitTime 1;
! Dvig valja
MoveL k1_5_Place_Above,v200,z100,t_Prijemalo\WObj:=wobj_posevna_plosca;
! Position
MoveJ k1_4_Position,v200,z100,t_Prijemalo\WObj:=wobj_plosca_posevnina;
! Pomik nad odlagalno mesto
MoveJ k1_3_Pick_Up,v200,z100,t_Prijemalo\WObj:=wobj_plosca_posevnina;
! Spust valja
MoveL k1_2_Pick,v300,fine,t_Prijemalo\WObj:=wobj_plosca_posevnina;
! Odpri prijemalo
openGripper;
WaitTime 1;
! Dvig prijemala
MoveL k1_1_Pick_Above,v300,z100,t_Prijemalo\WObj:=wobj_plosca_posevnina;
! Pomik na position
MoveJ k1_8_Position,v300,z100,t_Prijemalo\WObj:=wobj_posevna_plosca;
! Premik nad valj 9
MoveJ k2_7_Place_Up,v300,z100,t_Prijemalo\WObj:=wobj_posevna_plosca;
! Spust do valja
MoveL k2_6_Place,v200,fine,t_Prijemalo\WObj:=wobj_posevna_plosca;
```

5. Po vzoru premika za valj 1 naredite še to za preostalih 9 valjev.
6. Ko je podprogram urejen ga zapišite v glavnega in sinhronizirajte Rapid z robotsko celico.

Slika 164: Klic podprograma v glavnega za Vajo 10b

```
!Klic pick and place 10 posevnina obe smeri
Pick_and_place_10_posevnina_2;
WaitTime 1;
```

Slika 165: Točke in pot gibanja za Vajo 10b



Simulacija vaje

1. V robotski celice preverite delovanje simulacije. Postopek najprej zaženemo v koračnem načinu, nato še v avtomatskem načinu.
2. Ob pravilnem gibanju robotske roke je vaja končana.

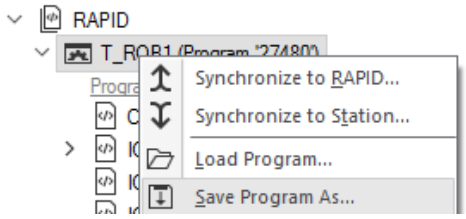
3.15 NALAGANJE PROGRAMA NA ROBOTA

Po uspešno izvedeni simulaciji in preverjanju pravilnosti gibov robota, orientacije orodja v okolju ABB RobotStudio, je sledil prenos programa na dejanskega šolskega robota. Namen prenosa je bil preveriti, ali se program, pripravljen v simulaciji, pravilno in varno izvaja tudi v realnem okolju. Za prenos programa sem uporabil dva različna načina, in sicer prenos s pomočjo USB ključka ter prenos z neposredno povezavo prenosnega računalnika z robotom, kar omogoča večjo prilagodljivost.

3.15.1 Prenos s pomočjo USB ključka

Program, pripravljen v simulacijskem okolju ABB RobotStudio na prenosnem računalniku, sem na dejanskega šolskega robota prenesel s pomočjo USB ključka. Najprej sem v RobotStudio shranil program kot RAPID modul (.mod) in ga skopiral na USB ključek. USB ključek sem nato priključil v USB vhod na učni enoti robotske roke. V Home oknu sem izbral ikono za kodo in v desnem zgornjem kotu pritisnil na tri pikice. To mi je ponudilo izbiro za naložitev modula. Nato sem izbral pogon, iz katerega želim modul shraniti v krmilnik robota ter izbral pravilen modul. Po nalaganju sem preveril pravilnost prenosa in program je bil prenesen.

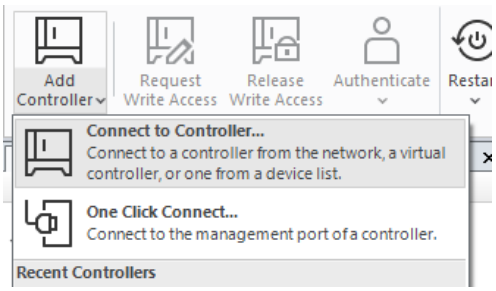
Slika 166: Postopek shranjevanja programa na USB ključek



3.15.2 Prenos s pomočjo omrežnega kabla

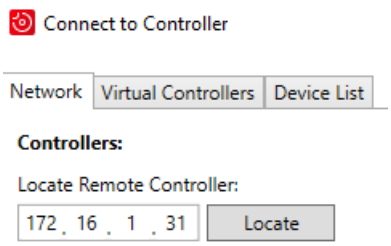
Drugi način prenosa programa na robota je bil izveden z neposredno povezavo prenosnega računalnika s krmilnikom robota preko omrežnega (LAN) kabla. Po vzpostavitvi fizične povezave je bilo potrebno v programski opremi ABB RobotStudio dodati dejanski krmilnik. V oknu Controller je bila izbrana možnost Add Controller in nato Real Controller.

Slika 167: Povezovanje na krmilnik



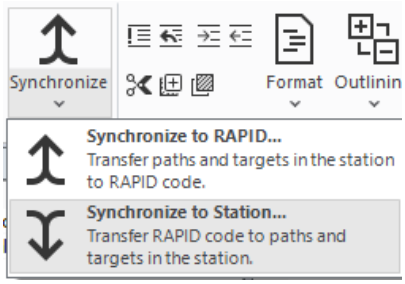
V naslednjem koraku je bil vnesen IP naslov robota, ki je bil znan iz predhodnega postopka aktivacije licence. Po uspešni vzpostavitvi povezave se je dejanski robot pravilno prikazal v simulacijski celici, kar je potrdilo pravilno komunikacijo med računalnikom in krmilnikom robota.

Slika 168: Povezava na IP naslov



Prenos programa na robota je bil izveden s pomočjo funkcije Synchronize. V postopku sinhronizacije je bila izbrana smer prenosa iz okolja RobotStudio v krmilnik robota. Pri tem je bil označen izključno programski modul, ki je vseboval razvite uporabniške programe, medtem ko sistemski podatki in nastavitve krmilnika niso bili vključeni v prenos. Na ta način je bil program varno in neposredno prenesen v pomnilnik robota, brez poseganja v obstoječo konfiguracijo sistema.

Slika 169: Prenos programa na krmilnik



4 ZAKLJUČEK

4.1 SKLEPI

V diplomskem delu so bili uspešno doseženi vsi zastavljeni cilji. Pripravljene so bile učne situacije za šolsko robotsko celico ABB, izvedene v simulacijskem okolju ABB RobotStudio, ki omogočajo spoznavanje osnov programiranja industrijskega robota brez uporabe dejanske opreme.
Razvite vaje vključujejo celoten postopek priprave simulacije, nastavitev virtualnega robota ter programiranje gibanja, kar omogoča postopno in razumljivo pridobivanje znanja. Pripravljena navodila omogočajo samostojno izvedbo vaj, zato so učne situacije primerne za uporabo pri pouku in nadaljnjem izobraževanju naslednjih generacij dijakov ali študentov.

4.2 DISKUSIJA

Uporaba simulacijskega okolja ABB RobotStudio se je izkazala kot zelo primerna za izobraževalne namene, saj omogoča varno in ponovljivo učenje programiranja industrijskega robota. Virtualni robot omogoča preizkušanje različnih rešitev brez tveganja za poškodbe opreme, kar je še posebej pomembno v šolskem okolju.
Med pripravo učnih situacij so se pojavile tudi določene težave, saj je šlo za prvo delo v programskem okolju ABB RobotStudio. Pri tem je prišlo do težav pri shranjevanju in ponovnem odpiranju simulacijske celice. Zaradi tega je bilo potrebno celico ponovno izdelati, pri čemer so bile referenčne točke uspešno prenesene iz varnostne kopije krmilnika. Reševanje te težave je prispevalo k boljšemu razumevanju strukture simulacijskega okolja ter postopkov za obnovo podatkov.
Pripravljene učne situacije so bile poleg simulacijskega okolja uspešno prenesene tudi na dejanski industrijski robot, kar potrjuje pravilnost in uporabnost razvite programske zasnove. S tem je bilo dokazano, da je mogoče vaje, pripravljene v simulatorju, neposredno uporabiti tudi v realnem okolju brez večjih prilagoditev.
Z vidika izobraževanja pripravljene učne situacije omogočajo postopno in sistematično pridobivanje znanja ter so primerne za uporabo pri pouku ali nadaljnjem usposabljanju. Njihova zasnova omogoča prilagajanje zahtevnosti glede na predznanje uporabnikov ter nadaljnje dopolnjevanje vsebin. Tako predstavljajo ustrezno osnovo za uporabo in razvoj učnih gradiv na področju industrijske robotike tudi za prihodnje generacije dijakov ali študentov.

5 VIRI

ABB. 2025. ABB CRB 15000. *CRB 15000.* [Elektronski] 2025. [Navedeno: 19. 11. 2025.] https://new.abb.com/products/3HAC074301-001/crb-15000.

—. 2025. ABB IRB. *IRB.* [Elektronski] 2025. [Navedeno: 11. 19. 2025.] https://www.abb.com/global/en/areas/robotics/products/robots/articulated-robots/medium-robots/irb-2600.

—. 2025. ABB IRB 2600. *IRB 2600.* [Elektronski] 2025. [Navedeno: 19. 11. 2025.] https://www.abb.com/global/en/areas/robotics/products/robots/articulated-robots/medium-robots/irb-2600.

—. 2025. ABB logo. [Elektronski] 2025. [Navedeno: 19. 11. 2025.] https://www.abb.com/global/en.

—. 2025. ABB Manual activation. [Elektronski] 2025. [Navedeno: 19. 11. 2025.] http://manualactivation.e.abb.com/.

—. 2025. ABB Robotics. *ABB Robotics.* [Elektronski] 2025. [Navedeno: 19. 11. 2025.] https://www.abb.com/global/en/areas/robotics.

—. 2025. ABB Robotics GoFa. *GoFa.* [Elektronski] 2025. [Navedeno: 19. 11. 2025.] https://www.abb.com/global/en/areas/robotics/products/robots/collaborative-robots/gofa.

—. 2025. ABB RobotStudio. *RobotStudio.* [Elektronski] 2025. [Navedeno: 19. 11. 2025.] https://www.abb.com/global/en/areas/robotics/products/software/robotstudio-suite/robotstudio-desktop.

—. 2025. ABB Šolanje navodila. *RobotStudio tečaj.* 2025.

—. 2025. History of ABB. *History of ABB.* [Elektronski] 2025. [Navedeno: 19. 11. 2025.] https://global.abb/group/en/about/history?utm_source=chatgpt.com.

Jerman, Karl. 2024. *ABB aktivacija licence.* 2024.

Sigh, Aman. 2025. eLearning Industry. *Scenario-Based Learning.* [Elektronski] 6. 7. 2025. [Navedeno: 3. 3. 2026.] https://elearningindustry.com/scenario-based-learning-transforming-corporate-ld-with-real-world-context.