Uporabniški vmesnik za robotski sistem pobiranja kosov v razsutem stanju iz zabojnika

Urban Kolman[1,2]
Mentor: izr. prof. dr. Aleš Hace[3], dr. Rok Pahič[2,3]

[1]Univerza v Mariboru, Fakulteta za strojništvo, Smetanova ulica 17, 2000 Maribor
[2]Moderne tehnologije d.o.o. , Kolodvorska ulica 35c, 2310 Slovenska Bistrica

[3]Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, Koroška cesta 46, 2000 Maribor
urban.kolman1@student.um.si

User Interface for a robotic bin-picking system

The article discusses the development and implementation of a modular web graphical interface for managing and controlling a robotic bin-picking system through Robotic Operating System 2 (ROS2). Web user interface, developed in the JavaScript programming language, is based on communication between a website and the ROS2 Humble, using the roslibjs library and the Rosbridge communication protocol. Key functionalities include executing ROS2 service calls and publishing and subscribing data. Enables visualizing a 3D model of the robot inside of robotic cell during operation and a point cloud of detected objects. The web interface allows intuitive modification of system parameters, deletion or creation of new projects stored in the MongoDB database. Integration with the noVNC client provides direct access to the PyQt application, which enables parametric modeling of CAD models of parts and definition of key picking points using the pyOCC library. The developed system, which combines a web graphical interface with a robotic cell, has been successfully implemented in the automotive industry, where it demonstrated improved efficiency and flexibility in the workflow.

Kratek pregled prispevka

Članek obravnava razvoj in implementacijo modularnega spletnega grafičnega vmesnika za upravljanje in nadzor robotskega sistema za pobiranje kosov v razsutem stanju iz zabojnika, preko robotskega operacijskega sistema (ROS2). Vmesnik, razvit v programskem jeziku JavaScript, temelji na komunikaciji med spletno stranjo in robotskim operacijskim sistemom ROS2 Humble, pri čemer uporablja knjižnico roslibjs in komunikacijski protokol Rosbridge. Ključne funkcionalnosti vključujejo izvajanje storitvenih klicev ter objavljanje in branje podatkov Omogoča vizualizacijo 3D modela robota v robotski celici med delovanjem in oblaka točk zaznanih objektov. Spletni vmesnik omogoča intuitivno spreminjanje sistemskih parametrov, brisanje ali ustvarjanje novih projektov, shranjenih v podatkovni bazi MongoDB. Integracija z odjemalcem noVNC zagotavlja neposreden dostop do PyQt aplikacije, ki omogoča parametrično modeliranje CAD modelov kosov in definiranje ključnih pobiralnih točk s pomočjo knjižnice pyOCC. Razviti sistem, ki združuje spletni grafični vmesnik z robotsko celico, je bil uspešno implementiran v avtomobilski industriji, kjer je demonstriral izboljšano učinkovitost in fleksibilnost v delovnem procesu.

1 Uvod

V sodobni industriji, kjer je avtomatizacija ključnega pomena za povečanje produktivnosti, zmanjšanje stroškov in izboljšanje kakovosti proizvodnih procesov, se pojavljajo novi izzivi pri robotizaciji vedno manj strukturiranih procesov, ki jih je prej lahko upravljal samo človek. Eden izmed sistemov, ki naslavljajva takšne izzive, je robotski sistem pobiranja kosov v razsutem stanju iz zabojnika (ang. bin-picking). Ta sistem omogoča, da robot prepoznava in pobira posamezne kose, ki so neurejeno zloženi v zabojnik, ter jih nato pravilno razporeja za nadaljnjo obdelavo. Tak sistem omogoča obdelavo različnih vrst kosov brez potrebe po predhodnem urejanju ali ločevanju, kar povečuje prilagodljivost proizvodnje [1] .

Učinkovitost in uspešnost delovanje takšnega sistema, ter še posebej hitrost in enostavnost njegove rekonfiguracije je močno odvisna od kakovosti uporabniškega vmesnika, ki ga uporablja operater za nadzor in konfiguracijo robotske celice. Grafični vmesnik zasnovan za sistem pobiranja kosov iz zabojnika mora jasno in pregledno predstaviti podatke o položaju kosov, uspešnosti pobiranja ter diagnostične informacije o delovanju sistema, kar je ključnega pomena za hitro identifikacijo in odpravljanje težav. Klasični HMI vmesniki pogosto ne omogočajo dovolj kompleksne vizualizacije in interakcije, zato smo se odločili za razvoj modularnega spletnega grafičnega vmesnika, ki omogoča napredno vizualizacijo, nadzor in prilagoditev parametrov na intuitiven.

2 Arhitektura robotskega sistema

V našem sistemu ROS2 deluje kot centralna povezovalna platforma, ki usklajuje vse komponente sistema. Omogoča komunikacijo in integracijo različnih naprav, kot so robot, programirljiv logični krmilnik (PLK), 3D kamera, in spletni grafični vmesnik. ROS2 združuje podatkovna sporočila, ki jih sistem potrebuje za delovanje med časom izvajanja, ter skrbi za usklajevanje procesov. Tako ROS2 deluje kot osrednji nadzorni sloj, ki omogoča nemoteno povezavo in usklajevanje vseh komponent. V našem sitemu smo uporabili distribucijo ROS2 Humble Hawksbill [2].

2.1 Komponente sistem

Robotski sistem za pobiranje kosov v razsutem stanju iz zabojnika je sestavljen iz več medsebojno povezanih komponent (slika 1), ki omogočajo celovit in avtomatiziran proces pobiranja. Glavna komponenta je računalnik z ROS2 sistemom, ki centralno povezuje vse ostale komponente. Za zajem tridimenzionalnih podatkov o položaju objektov je uporabljena globinska kamera, ki zagotavlja visoko natančnost pri prepoznavanju kosov. Interakcijo uporabnika s sistemom omogoča zaslon na dotik, ki služi kot intuitiven grafični vmesnik. Ključno vlogo pri manipulaciji kosov ima industrijski robot, opremljen z ustreznim vakuumskim prijemalom, ki omogoča varno in učinkovito prijemanje objektov. Za krmiljenje in sinhronizacijo posameznih strojnih komponent sistema je uporabljen PLK, ki zagotavlja stabilno in zanesljivo izvajanje procesov.

2.2 Komunikacija

Neprekinjena komunikacija med naštetimi komponentami sistema je ključnega pomena za zanesljivo delovanje celotnega sistema. Jedro komunikacije predstavlja računalnik z ROS2, ki omogoča integracijo vseh komponent. Za izmenjavo podatkov med napravami se uporabljajo različni komunikacijski protokoli, kot je prikazano na sliki 1.

Slika 1: Komunikacijski protokoli med komponentami sistema

3D kamera je s sistemom ROS2 povezana preko omrežnega protokola TCP/IP, ki omogoča stabilno komunikacijo v času izvajanja. Industrijski robot je z ROS2 povezan preko UDP protokola, kar omogoča izmenjavo podatkov o željenih pobiralnih točkah. Za integracijo Siemens PLK z ROS2 sistemom je uporabljen Siemensov komunikacijski protokol SNAP7, ki omogoča dvosmerno komunikacijo med računalnikom in PLK-jem. SNAP7 je odprtokodni protokol, ki omogoča branje in pisanje podatkovnih blokov (DB), pridobivanje vhodno-izhodnih (I/O) podatkov ter manipulacijo s časovniki (ang. timers) in označevalniki (ang. markers) [9]. Robot in PLK sta med seboj povezana preko industrijskega komunikacijskega protokola ProfiNET.

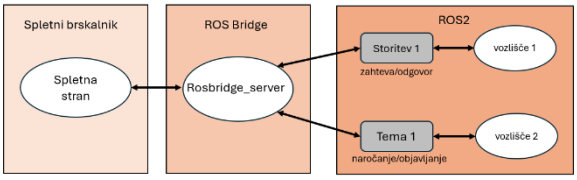
Za interakcijo uporabnika s sistemom je implementiran spletni grafični vmesnik, ki se prikazuje na zaslonu na dotik. Vmesnik komunicira z ROS2 prek protokola ROS Bridge, ki temelji na WebSocket in HTTP protokolih ter omogoča asinhrono izmenjavo podatkov v formatu JSON [3]. Vmesnik je zasnovan kot univerzalna komunikacijska platforma, kar omogoča povezovanje z različnimi napravami, ne glede na uporabljene komunikacijske protokole. Z modularno arhitekturo lahko vmesnik podpira različne industrijske sisteme, kar omogoča široko prilagodljivost in uporabo v različnih avtomatiziranih okoljih. S tem zagotavlja univerzalno komunikacijo s sistemom ROS2, hkrati pa omogoča enostavno razširitev za specifične aplikacije in potrebe drugih naprav.

3 Tehnologije

Grafični vmesnik je zasnovan kot sodobna spletna aplikacija, ki omogoča učinkovito komunikacijo s sistemom ROS2 in uporabniku prijazno upravljanje robotske celice. Za delovanje vmesnika so uporabljene različne tehnologije in knjižnice, ki omogočajo stabilno in odzivno interakcijo med uporabnikom in sistemom. Spletni vmesnik je razvit s pomočjo standardnih spletnih tehnologij, vključno z JavaScript, HTML in CSS. HTML je uporabljen za strukturo strani, CSS za oblikovanje in prilagodljivost uporabniškega vmesnika, medtem ko JavaScript omogoča dinamično interakcijo in povezavo s sistemom ROS2. Za gostovanje spletnega grafičnega vmesnika je uporabljen statični npm HTTP strežnik, ki omogoča distribucijo spletnih vsebin in zagotavlja stabilno povezavo med uporabnikom in sistemom. Strežnik omogoča hitro nalaganje in odzivnost vmesnika, kar je ključno za nemoteno delovanje sistema [4]. V trenutni konfiguraciji dostopamo do spletne strani lokalno prek računalnika, na katerem teče sistem. V primeru povezave računalnika v omrežje pa bi bilo mogoče dostopati do spletne strani tudi prek tablice ali pametnega telefona, kar omogoča večjo fleksibilnost pri upravljanju sistema.

3.1 ROS Bridge

Za vzpostavitev komunikacije med spletnim grafičnim vmesnikom in sistemom ROS2 smo uporabili paket rosbridge_suite [5]. Ta paket omogoča komunikacijo med spletnimi aplikacijami in ROS2 preko protokola WebSocket, kar omogoča pošiljanje in prejemanje sporočil v JSON formatu S tem smo dosegli izvajanje storitvenih klicev, naročanje in objavljanje na teme, izvajanje akcij ter branje in nastavljanje parametrov v sistemu ROS2. Slika 2 prikazuje komunikacijski most med spletno stranjo in ROS2 preko ROS Bridge strežnika.



Slika 2: Komunikacijski most med spletno stranjo in ROS2

3.2 Knjižnici roslibjs in ros3djs

Za izvedbo komunikacije med spletnim grafičnim vmesnikom in ROS2 smo izkoristili dve ključni odprtokodni knjižnici iz zbirke Robot Web Tools: roslibjs in ros3djs. Obe knjižnici sta napisana v programskem jeziku JavaScript, omogočata enostavno integracijo ROS-a v spletne aplikacije, ter podpirata širok spekter funkcionalnosti. Knjižnica "roslibjs" nam omogoča izvajanje klicev storitev, naročanje in objavljanje na podatke, izvajanje akcije, branje in

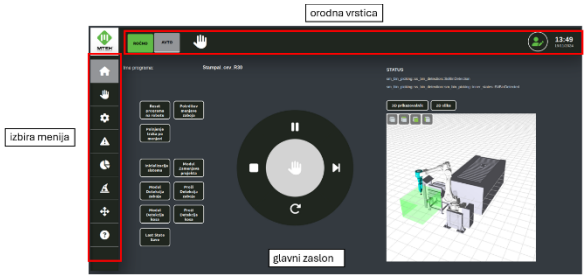
nastavljanje parametrov ter vzpostavitev komunikacije med spletno stranjo in ROS Bridge strežnikom.

Za izvedbo klica storitve je najprej potrebno definirati instanco objekta "ROSLIB.Service", kjer določimo ime storitve in njen ustrezni tip, ki ga zahteva sistem ROS2. Nato pripravimo podatkovno strukturo zahteve v skladu s predpisanim tipom storitve. Klic storitve izvedemo z metodo "callService", ki pošlje zahtevo strežniku ROS in pridobi povratni odgovor z rezultatom izvedene operacije. Podobno lahko s pomočjo metode "publish" objavljamo podatke na določeno temo, medtem ko metoda "subscribe" omogoča naročanje na določeno temo in sprejemanje podatkov, ki jih objavljajo druga vozlišča v sistemu ROS2. Ta mehanizem zagotavlja asinhrono in učinkovito izmenjavo podatkov med spletnim grafičnim vmesnikom in robotskim sistemom [6].

Knjižnico "ros3djs" smo uporabili za napredno 3D vizualizacijo vmesnika. Zgrajena je na osnovi "roslibjs" in izkorišča zmogljivosti knjižnice "three.js" za prikaz kompleksnih tridimenzionalnih scen [7]. Omogoča prikaz URDF modela robota, oblaka točk, vizualizacijo STL modela celice, markerjev in drugih grafičnih elementov, ter tako ponuja enako vizualno zmogljivost kot ROS2 vizualizator Rviz. S tem je omogočena jasna in natančna upodobitev procesa, kar izboljša uporabniško izkušnjo pri upravljanju robotskega sistema.

4 Razvoj vmesnika

Spletni grafični vmesnik je razvit kot sodobna spletna stran. Strukturo vmesnika sestavlja glavni prikazovalni zaslon, ki omogoča uporabniku spremljanje vsebine posameznih menijev in interakcijo s sistemom. Navigacija med meniji poteka prek stranske izbire, ki omogoča hitro preklapljanje med različnimi funkcionalnostmi. Vmesnik vključuje tudi orodno vrstico, kjer lahko uporabnik izbira režim delovanja sistema, izvaja prijavo uporabnika ter spremlja trenutno stanje robotske celice (slika 3).



Slika 3: Spletni grafični vmesnik

4.1 Upravljanje sistema

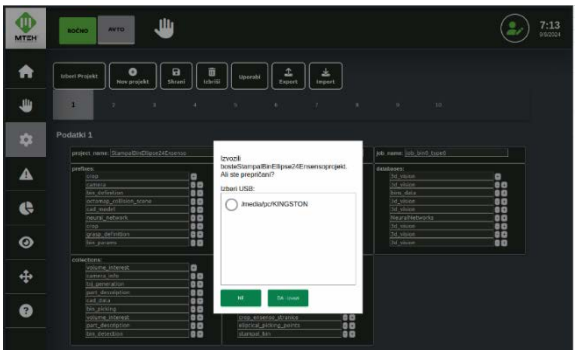
Vmesnik omogoča uporabniku enostavno upravljanje celotnega sistema. Preko objavljanja podatkov na določene teme lahko zapisujemo vrednosti v podatkovne bloke PLK-ja, kar omogoča izvedbo ključnih operacij, kot sta zagon in zaustavitev sistema. Poleg osnovnega nadzora lahko uporabnik spreminja parametre, povezane z delovanjem robotske celice, kar omogoča prilagoditev sistema glede na specifične zahteve proizvodnega procesa. Z naročanjem na podatke, objavljene na ROS temah, lahko v času izvajanj spremlja stanje stroja, spremlja ključne operativne parametre ter diagnosticira napake in odzive sistema.

4.2 Dostop do projektnih parametrov

Kot dodatno funkcionalnost spletni grafični vmesnik omogoča dostop do parametrov, povezanih z definiranimi projekti za specifične kose pobiranja. Uporabnik lahko za vsak projekt določi tip obdelovanca in mu dodeli ustrezne parametre, ki so ključni za učinkovito pobiranje. Ti parametri, kot so dimenzije, pobiralne točke, dimenzije zaboja so shranjeni v podatkovni bazi MongoDB.

Vmesnik je zasnovan dinamično, kar pomeni, da se vsi parametri na spletni strani samodejno posodabljajo glede na spremembe v podatkovni bazi. Če se dodajo novi parametri, se ti samodejno prikažejo v uporabniškem vmesniku, kar omogoča prilagodljivo in razširljivo upravljanje projektov brez potrebe po ročnih posodobitvah spletne aplikacije. Preko spletnega vmesnika ima uporabnik možnost pregleda, urejanja in shranjevanja projektov, s čimer lahko prilagaja sistem glede na zahteve posameznih tipov kosov. Poleg tega lahko projekte izbriše ali

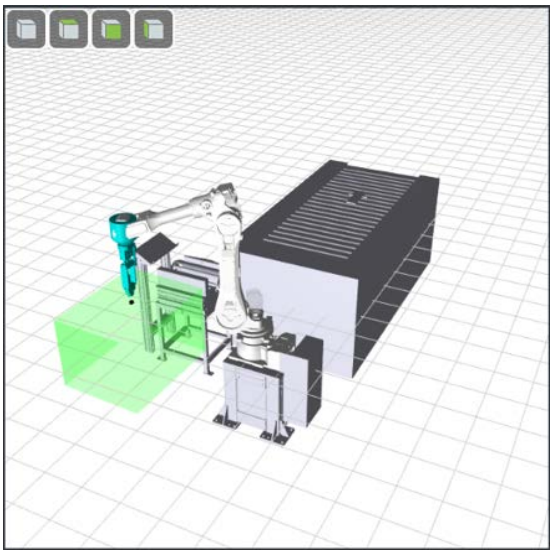
jih aktivira za neposredno uporabo v delovnem procesu. Ta funkcionalnost omogoča hitro menjavo med različnimi kosi, saj uporabnik ob ponovni obdelavi že obstoječega obdelovanca preprosto izbere ustrezen projekt iz baze, sistem pa takoj začne s pobiranjem brez dodatne konfiguracije. Dodatno vmesnik omogoča izvoz izbranega projekta na zunanji disk, kar omogoča prenos projektov med različnimi robotskimi sistemi. S tem lahko uporabnik hitro in enostavno prenese obstoječe nastavitve na drugo napravo, kar omogoča večjo fleksibilnost in hitrejšo integracijo robotskih sistemov v proizvodnem procesu. Slika 4 prikazuje pojavno okno za izvoz projekta, kjer izberemo željen zunanji disk.



Slika 4: Izvoz projektov

4.3 Vizualizacija stanja celice

Na začetni meni spletnega grafičnega vmesnika smo vključili 3D prikazovalnik, ki omogoča podroben vpogled v stanje robotskega sistema med delovanjem. Znotraj 3D vizualizacije je prikazan STL model robotske celice, kar uporabniku omogoča boljše razumevanje delovnega okolja. Poleg tega se dinamično prikazujejo položaj in dimenzije zaboja, kot jih zazna 3D kamera, kar omogoča spremljanje sprememb v okolju v času izvajanja. Dodatno je v prikazovalnik vključen oblak točk zajetih kosov, ki ga generira kamera, kar omogoča vizualizacijo zaznanih objektov v zaboju. Vmesnik prav tako omogoča prikaz URDF modela robota, s katerim uporabnik lahko spremlja premike robota v času izvajanja (slika 5). S pomočjo naročanja na ROS temo, kjer se objavljajo dejanske vrednosti sklepov robota, lahko sistem sinhronizirano prikazuje premikanje robota med delovanjem. To omogoča natančno spremljanje gibanja robotske roke in prilagajanje procesnih parametrov glede na realno stanje v proizvodnem okolju. Poleg tega omogoča, da lahko že ob hitrem pregledu vmesnika uporabnik hitro ugotovi morebitne nepravilnosti v procesu ter pravočasno reagira na napake.



Slika 5: 3D vizualizacija

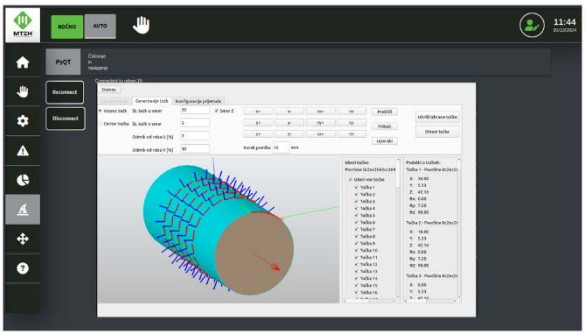
3D prikazovalnik lahko zamenjamo s prikazom 2D slike zajetih kosov. Slika se dinamično posodabljala v času izvajanja, glede na uspešnost lokalizacije kosov, kar omogoča hitro zaznavanje morebitnih napak v procesu detekcije in njihovo takojšnjo odpravo (slika 6).

Slika 6: 2D slika zajetih kosov

4.4 Parametrično modeliranja in izbira pobiralnih točk

Za uspešno izvajanje detekcije in pobiranja kosov z robotskim sistemom je ključnega pomena natančna definiranje CAD modela kosa ter določitev pobiralnih točk na modelu. Za to smo razvili namensko aplikacijo, ki omogoča uporabniku, da interaktivno nastavi parametre CAD modela in izbere pobiralne točke. Aplikacija je razvita v programskem jeziku Python in temelji na knjižnici PyOCC, ki omogoča manipulacijo CAD geometrije, 3D modeliranje in vizualizacijo, kar je ključno pri določanju optimalnih pobiralnih točk [8]. Za razvoj grafičnega vmesnika smo uporabili PyQt, kar omogoča enostavno integracijo z 3D prikazom in zagotavlja intuitivno uporabniško izkušnjo. Za vključitev aplikacije v spletno stran smo uporabili JavaScript knjižnico noVNC, ki omogoča dostop do VNC strežnika preko brskalnika. Z uporabo noVNC smo omogočili, da se uporabniki lahko povežejo z VNC strežnikom, kjer se izvaja Python aplikacija. Na ta način smo omogočili interaktivno upravljanje aplikacije na oddaljenem strežniku s spletnega grafičnega vmesnika.

Postopek modeliranja se začne z izbiro tipa modela (cevni ali eliptični tip), pri čemer se model na podlagi izbranih geometrijskih parametrov samodejno generira in prikaže v 3D prikazovalniku. V drugem koraku uporabnik določi pobiralne točke, pri čemer izbere ravnino pobiranja, nastavi parametre porazdelitve točk ter prilagodi število točk v U- in V-smeri. Ob pritisku na gumb "Prikaži" se točke dinamično generirajo in prikažejo v 3D prostoru, pri čemer so koordinate in rotacije točk prikazane v desnem delu vmesnika (slika 7). Uporabnik lahko točke premika, briše ali generira nove, kar omogoča natančno prilagoditev za specifične naloge pobiranja. Aplikacija prav tako omogoča uvoz modelov v formatu STEP, kar povečuje njeno fleksibilnost za različne vrste kosov. Na koncu uporabnik določi ustrezno konfiguracijo prijemala, ki je odvisna od dimenzij obdelovanca, s čimer zagotovi stabilnost in učinkovit oprijem med manipulacijo. Ko je uporabnik zadovoljen z nastavitvami in generiranim projektom, le-tega shrani v podatkovno bazo, kar omogoča enostavno shranjevanje, urejanje in kasnejši dostop do projektov za nadaljnjo uporabo ali analizo.



Slika 7: Program za modeliranje kosov in izbiro pobiralnih točk

5 Rezultati

Spletni grafični vmesnik smo uspešno implementirali v industrijsko okolje, kjer je pokazal odlično zmogljivost pri upravljanju kompleksnih robotskih operacij (slika 8). Spletni vmesnik omogoča uporabnikom enostavno konfiguracijo parametrov, spremljanje stanja sistema ter hitro odzivanje na morebitne napake. Dodatne funkcionalnosti, kot so parametrično modeliranje kosov, izbira pobiralnih točk in shranjevanje projektov v podatkovno bazo, so še povečale prilagodljivost in fleksibilnost sistema. Zasnova je intuitivno, kar se pokaže, da uporabniku zgolj v parih minutah omogoča prilagoditve sistema za nove oblike kosov.

Slika 8: Integracija sistema v industriji

Pomembna lastnost vmesnika je njegova modularna in univerzalna zasnova, ki omogoča hitro in enostavno prilagoditev različnim industrijskim sistemom. Njegova prilagodljivost se kaže v tem, da smo ga uspešno uporabili tudi v procesu paletizacije, kjer je bil integriran drugačen tip robota, vendar smo kljub temu z istim vmesnikom dosegli enake rezultate (slika 9). Ta prilagodljivost potrjuje, da vmesnik ni vezan na specifično strojno opremo, temveč omogoča široko interoperabilnost in uporabo v različnih avtomatiziranih procesih, kar zmanjšuje potrebo po razvoju novih vmesnikov za vsako specifično aplikacijo.

Slika 9: Vmesnik v procesu paletizacije

6 Zaključek

V okviru tega projekta smo uspešno zasnovali in implementirali univerzalen in modularen spletni grafični vmesnik za nadzor in upravljanje robotskega sistema za pobiranje kosov v razsutem stanju iz zabojnika. Vmesnik je bil razvit s poudarkom na intuitivnosti, prilagodljivosti in hitrosti integracije v različne proizvodne procese. Njegova modularna zasnova omogoča hitro prilagajanje na različne zahteve avtomatizacije, kar ga naredi primerno rešitev za širok spekter industrijskih aplikacij (predstavljeno v poglavju 5)

Ključna prednost našega vmesnika je njegova sposobnost celovite integracije z ROS2, kar omogoča nemoteno komunikacijo med vsemi komponentami sistema, vključno z robotom, 3D kamero, PLK in drugimi napravami. Pomembno je poudariti, da nam univerzalna zasnova omogoča komunikacijo z različnimi tipi naprav, ne glede na njihov komunikacijski protokol. S pomočjo naprednih tehnologij, kot so ROS Bridge, WebSocket in knjižnice roslibjs ter ros3djs, smo zagotovili stabilno in odzivno interakcijo med uporabnikom in sistemom. To je omogočilo natančno vizualizacijo delovanja sistema v času izvajanja, kar izboljšuje preglednost in učinkovitost nadzora.

7 Literatura

[1] C. Martinez, R. Boca, B. Zhang, H. Chen and S. Nidamarthi, Automated bin picking system for randomly located industrial parts, 2015.

[2] ROS2 documentation: Humble. Dosegljivo: https://docs.ros.org/en/humble/index.html. [Dostopano:24.2.2025].

[3] Biobotus. Rosbridge suite. Dosegljivo: https://github.com/biobotus/rosbridge_suite. [Dostopano:24.2.2025].

[4] NPM. http-server. Dosegljivo: https://www.npmjs.com/package/http-server. [Dostopano:24.2.2025].

[5] RobotWebTools. Rosbridge_suite. Dosegljivo: https://github.com/RobotWebTools/rosbridge_suite. [Dostopano:25.2.2025].

[6] RobotWebTools. roslibjs. Dosegljivo: https://github.com/RobotWebTools/roslibjs. [Dostopano:25.2.2025].

[7] RobotWebTools. Ros3djs. Dosegljivo: https://github.com/RobotWebTools/ros3djs. [Dostopano:25.2.2025].

[8] Opencascade. Pythonocc. Dosegljivo: https://dev.opencascade.org/project/pythonocc. [Dostopano:25.2.2025].

[9] SNAP7. Dosegljivo: https://snap7.sourceforge.net. [Dostopano:25.2.2025].