Programiranje robota ABB IRB120 sa integriranim strojni vidom

Šarić, Karlo

Sveučilište
Sjever

Završni rad br. 004/MEH/2022

Programiranje robota ABB IRB120 sa integriranim strojnim vidom

Karlo Šarić, 033 6027 935

Varaždin, srpanj 2022. godine

Sveučilište Sjever

Odjel za Mehatroniku

Završni rad br. 004/MEH/2022

Programiranje robota ABB IRB120 sa integriranim strojnim vidom

Student

Karlo Šarić, 033 6027 935

Mentor

Zoran Busija, dipl. ing. strojarstva

Varaždin, srpanj 2022. godine

Sažetak

Tema ovog rada je postupak implementacije vizijskog sustava u radnu okolinu industrijskog robota. Cilj rada je upoznavanje s načinom rada ABB integriranog vida industrijske primjene te pokretanje i puštanje u pogon istog na konkretnom primjeru s robotom. Na kraju rada napisali smo program koji demonstrira rad robota u suradnji sa stacionarnom kamerom na problemu sortiranja domino pločica.

Ključne riječi: IRB 120, IRC5 Compact, ABB Integrated Vision, Machine Vision, Cognex 7402.

Summary

Topic of this thesis is the implementation of a machine vision system into the work environment of an industrial robotic arm. The goal is to describe the working principles of ABB's integrated vision and its commissioning on a real-life example. At the end of the thesis, we attached the program code written in RAPID programming language that was used to demonstrate how the robot and a stationary camera work together to solve a sorting problem.

Key words : IRB 120, IRC5 Compact, ABB Integrated Vision, Machine Vision, Cognex 7402.

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR–42000 Varaždin

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL: Odjel za mehatroniku

STUDIJ: preddiplomski stručni studij Mehatronika

PRISTUPNIK: Karlo Šarić | JMBAG: 0336027935

DATUM: 01.06.2022. | KOLEGIJ: Robotika

NASLOV RADA: Programiranje robota ABB IRB120 sa integriranim strojni vidom

NASLOV RADA NA ENGL. JEZIKU: Programming of a robot ABB IRB120 with Integrated Vision

MENTOR: Zoran Busija, dipl. ing. stroj. | ZVANJE: predavač

ČLANOVI POVJERENSTVA:

1. Siniša Švoger, dipl.ing.stroj, predavač - predsjednik povjerenstva
2. Zoran Busija, dipl.ing.stroj, predavač
3. prof.dr.sc. Ante Čikić
4. Josip Srpak, dipl.ing.el., predavač - rezervni član
5.

Zadatak završnog rada

BROJ: 004/MEH/2022

OPIS:

U završnom radu potrebno je:
Opisati dijelove robotskog sustava u koji je ugrađena kamera.
Prikazati rad sa programskim alatima koji omogućavaju korištenje kamere u sprezi s robotom.
Objasniti strukturu i potrebne naredbe programskog jezika RAPID.
Napisati program koji će koristiti strojni vid i izvršavati se na robotu ABB IRB 120.

ZADATAK URUČEN: 20.06.2022. | POTPIS MENTORA: Busija Zoran

REPUBLIKA HRVATSKA
SVEUČILIŠTE SJEVER

Popis korištenih kratica

ABB	ASEA Brown Boveri
AIA	Automated Imaging Association
IRB	Industrial Robot
IRC	Industrial Robot Controller
LED	Light Emmiting Diode
PU	Polyurethane
CCD	Charge Coupled Device
CMOS	Complementary Metal-Oxide semiconductor
TCP	Tool Centar Point

Sadržaj

1. Uvod

Sustav kojim ćemo se koristiti sastoji se od sljedećih elemenata: industrijski robot ABB IRB 120, upravljački jedinica ABB IRC 5 Compact, SCHUNK pneumatska prihvatnica, industrijska kamera Cognex 7402 serija i jedno stolno upravljačko računalo koje nam služi za nadzor i programiranje robota i kamere. Kroz rad ćemo detaljnije opisati spomenute elemente sustava te objasniti kako su međusobno povezani. Bitno je da se pravovremeno upoznamo sa mogućnostima i ograničenjima našeg sustava kako kasnije ne bi došlo do nepredviđenih problema koje bi lako izbjegli da smo poznavali alate s kojima radimo.

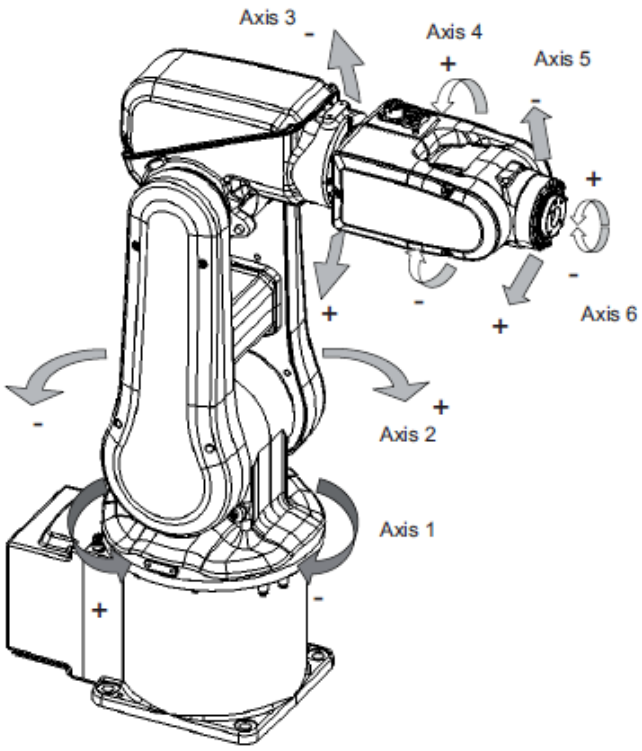
Rad je strukturiran tako da se na početku opisuju i upoznaju pojedini elementi sustava, nakon toga detaljno se objašnjava postupak njihovog umrežavanja, postupak slaganja u radnu cjelinu te na posljetku upoznajemo kako koristiti sustav koji smo stvorili za rješavanje zadanih problema. Oprema koju smo koristili u posjedu je našeg sveučilišta, isto tako tema i način njenog prezentiranja u ovom radu izvedeni su tako da rad može poslužiti kao dodatni izvor za učenje. Buduće generacije studenata na odjelu mehatronike našeg sveučilišta moći će lakše u svojim projektima koristiti kameru kao izvor informacija u radu s industrijskim robotom. Važno je napomenuti da se ovaj rad kao takav ne koristi kao zamjena za priručnike i podatkovne liste korištenih elemenata dostupne od strane proizvođača. Većina informacija o elementima sustava izvučena je iz njihovih priručnika te su isti navedeni kao izvori na kraju rada uz web adrese gdje ih se može preuzeti. U sljedećem poglavlju krenuti ćemo s upoznavanjem opreme.

2. Upoznavanje s elementima sustava

Sustav koji koristimo sastoji se od mnogo pojedinih elemenata. Da bi pojednostavili priču podijeliti ćemo ih na dvije osnovne grupe. Prva grupa sadrži opis samog robota i svih elemenata potrebnih za njegovo pravilno funkcioniranje, druga grupa sadrži hardverske dodatke koje ćemo koristiti, a koji nisu nužni za funkcioniranje samog robota.

2.1. Robot i kontroler

Kao što je već ranije spomenuto robot kojeg koristimo za ovaj projekt je proizvod Švedsko-Švicarskog proizvođača robotske i industrijske opreme - ABB (ASEA Brown Boveri). Puni naziv robota je IRB 120 3-58. Proizvođači industrijskih robota se često vode pravilom kojim specifikacije samog proizvoda grade njegovo ime, pa tako i u ovom slučaju; IRB 120 stoji za „Industrial Robot 120“, sljedeća znamenka „3“ govori nam koja je propisana nosivost robota što je u našem slučaju 3 kilograma te na posljetku broj 58 predstavlja najveću horizontalnu udaljenost koju robot može pravilno dosegnuti, u našem slučaju to je 0.58 metara. IRB 120 je robot sa 6 stupnjeva slobode označenih na slici ispod. Svi stupnjevi slobode IRB-a 120 izvedeni su kao rotacijski zglobovi, stoga IRB 120 spada u kategoriju artikuliranih robota. Svaka os pogonjena je sa jednim motorom na izmjeničnu struju. Motori i njihovi pripadajući reduktori koji pokreću osi robota izabrani su tako da ne zahtijevaju praktički nikakvo održavanje. [1][2]

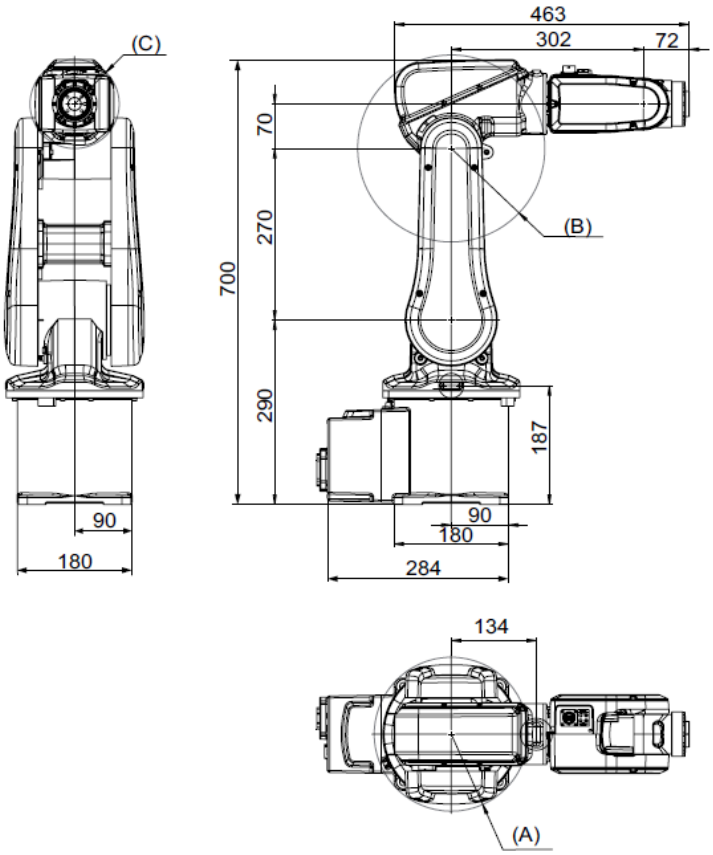


Slika 1 – Prikaz robota s označenim osima [2]

Ovaj robot kao takav je među prvim industrijskim robotima vrlo male nosivosti koje je ABB stavio na tržište. Namjena mu nije manevriranje glomazim dijelovima u proizvodnji već vrlo precizna montaža sitnih komponenti ili pick-and-place laganih predmeta na primjer u farmaceutskoj industriji pri manipuliranju ljekovima. Još jedna zanimljiva činjenica za IRB 120 je ta što zadovoljava standard rada u ekstremno čistom okruženju. Za rad u tako zvanim čistim sobama roboti moraju zadovoljavati standard DIN EN ISO 14644-1 koji nalaže da robot ne smije u svojem radu ili tokom procesa održavanja za sobom ostavljati čestice koje proizvode nečistoću. Da bi zadovoljio ovaj standard robot je obojen s četiri sloja PU (Polyurethane) boje s dodatnim premazom preko naljepnica kako bi se olakšalo čišćenje. [2]

IRB 120 specifikacija	**Podatak**
Nosivost	3 kg
Doseg	0.58 m
Masa robota	25 kg
Energetska potrošnja (ISO cube, max load)	0.24 kW
Energetska potrošnja-kočnice ON	0.095 kW
Energetska potrošnja-kočnice OFF	0.173 kW

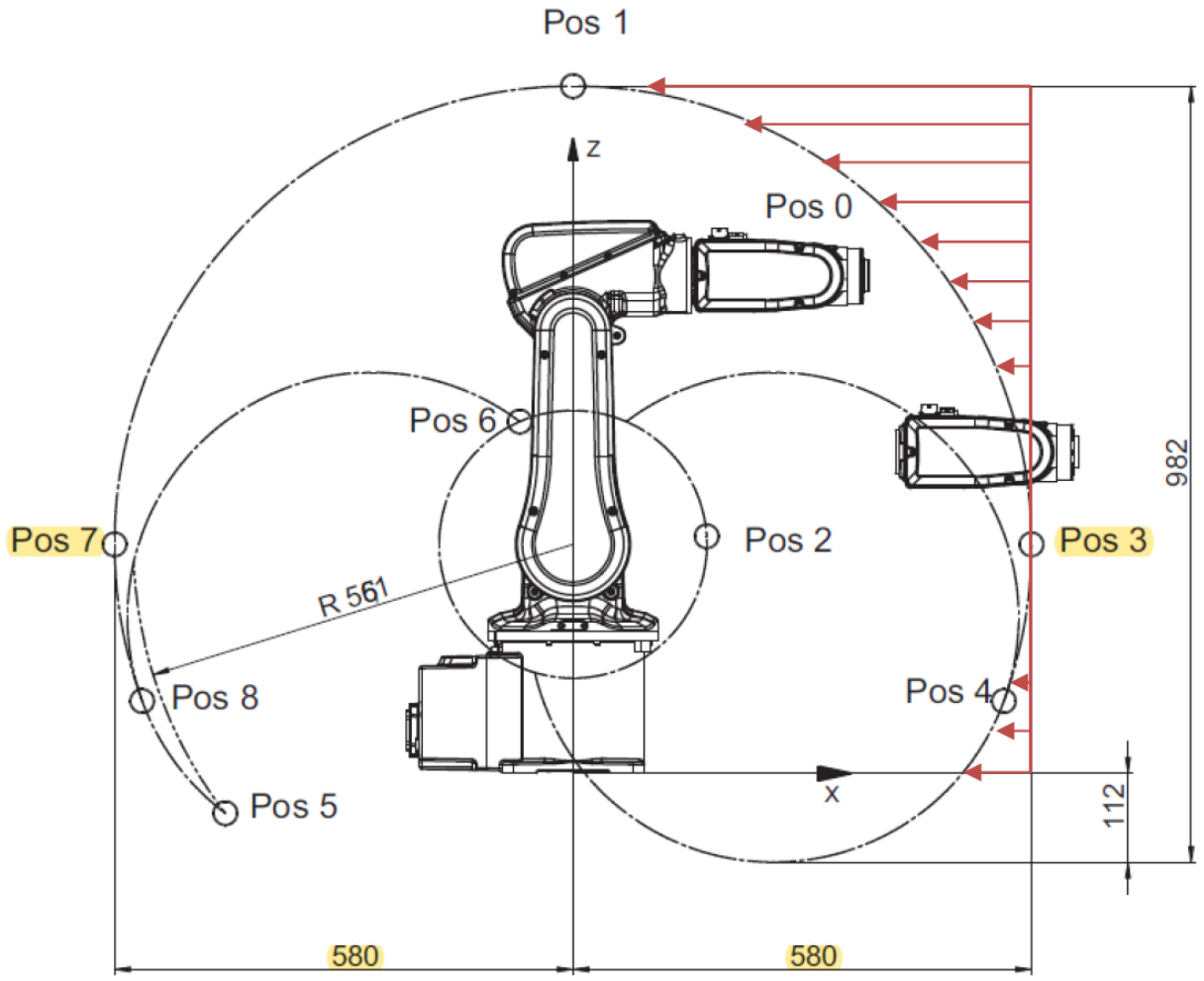
Tablica 1 – Osnovni tehnički podatci[2]



Pozicija	Min. R rotacije
A	R=121 mm
B	R=147 mm
C	R=70 mm

Slika 2 – Kotirane projekcije robota[2]

Većina artikuliranih robota sa šest stupnjeva slobode imaju sličan shematski izgled radnog područja. Nekada se lako zbuniti kada projektiramo okruženje robota misleći da robot može pravilno dohvatiti objekt bilo gdje u radijusu od 0.58 metara kao što je naznačeno u specifikacijama, no to nije slučaj. Slikom ispod prikazan je izgled radnog područja bez montirane prihvatnice. Bitno je da primijetimo gdje se nalazi točka maksimalnog dosega i kako se taj maksimalni doseg mijenja na različitim visinama u odnosu na bazu robota.



Slika 3 – radno područje IRB 120 [2]

Na slici možemo primijetiti da radno područje robota nije simetrično s prednje i stražnje strane. Robot ima jako specifičan oblik područja kojeg može doseći na stražnjoj strani i na to moramo pripaziti ili iskoristiti najbolje što možemo. Sljedeće, primijetimo točku (Pos 3) na udaljenosti od 580 milimetara od baze koja je dana kao maksimalni doseg robota. Vidimo kako točke koje spadaju unutar prostora kojeg zatvara vertikalni pravac kroz točku Pos 3 od robota nisu uvijek u radnom području, problem je prikazan crvenim strelicama na slici. Također vrijedi spomenuti da IRB 120 može doseći i mali dio prostora ispod razine svoje baze kako je prikazano i na slici.

Stupanj slobode	**Raspon zakreta**
Zglob 1	Od +165º do -165º
Zglob 2	Od +110º do -110º
Zglob 3	Od +70º do -110º
Zglob 4	Od +160º do -160º

Zglob 5	Od +120º do -120º
Zglob 6	Od +400º do -400º (tvornički) +242 okretaja do -242 okretaja maksimalno *

Tablica 2 – Raspon pokreta [2]

Tablicom 2 su dane vrijednosti raspona zakreta pripadajućih zglobova što nam može biti korisno pri pisanju programa za pokrete robota. Zvjezdicom(*) označena je druga verzija načina funkcioniranja šestog zgloba, točnije vrha robota ili nekada nazivanom glavom ili šakom. Tvornički zadan raspon pokreta ovog zgloba je ukupno 800 stupnjeva, no često ćemo ovaj zglob rotirati za maksimalno 360 stupnjeva u jednu ili drugu stranu zato što će nam prihvatinca biti montirana na ovaj zglob. Pneumatski ili električno pogonjene prihvatnice uvijek su povezane nekom vrstom kabela ili crijeva koje nam sprječava okret za više od 360 stupnjeva, u nekim slučajevima ovisno o načinu montaže – i manje. Ovaj problem moguće je izbjeći korištenjem dodatnog postolja za alat kroz kojeg možemo provući električne kablove ili crijeva za zrak i time omogućiti alatu nesmetanu rotaciju preko 360 stupnjeva[10].

Tip robota	**Zglob 1**	**Zglob 2**	**Zglob 3**	**Zglob 4**	**Zglob 5**	**Zglob 6**
IRB 120 – 3/0.58	250 º/s	250 º/s	250 º/s	320 º/s	320 º/s	420 º/s

Tablica 3 – Maksimalne brzine zglobova[2]

Sada kada smo upoznali strukturu mehaničkog dijela robota možemo krenuti sa pričom o upravljačkoj jedinici robota (u daljnjem tekstu ćemo zbog kraćeg zapisa često koristiti naziv kontroler). IRB 120 dolazi u kompletu s IRC5 C (Indrustrial Robot Controller 5 Compact) upravljačkom jedinicom. U osnovi IRC5 C je računalo sa generičkim procesorom s popratnim komponentama poput radne memorije i prostora za pohranu podataka te odvojenim napajanjima za računalo i motore robota.

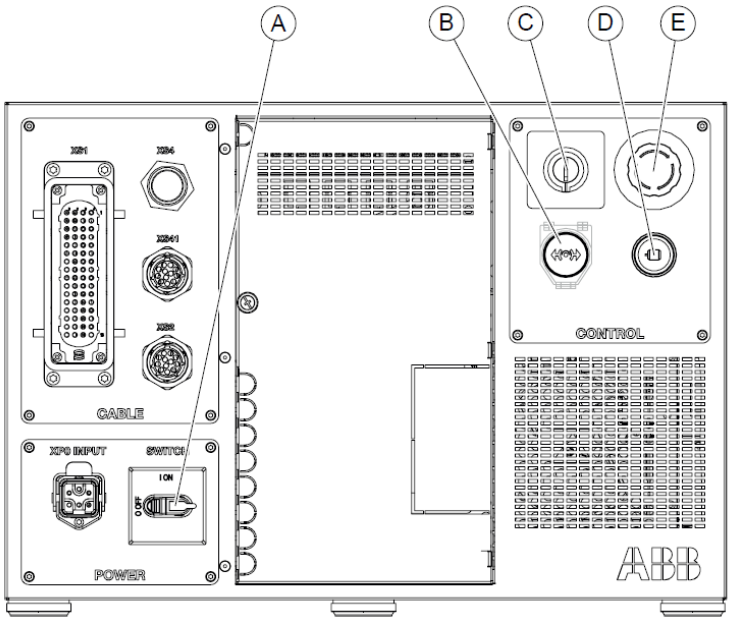
Dodatne komponente koje IRC 5 C posjeduje, a koje ga razlikuju od klasičnog stolnog računala su[3]:

- Linijski filteri – služe za filtriranje elektromagnetskih smetnji na signalnim kanalima
- Razvodne pločice za motore – služe za napajanje i nadzor motora
- Sigurnosna pločica – pre-naponska zaštita
- Pločica Ulazno/izlaznih signala – organizacija ulaznih i izlaznih kontakata
- Sigurnosno napajanje – služi za sigurno gašenje u slučaju nestanka struje

Slika 4 – IRC 5 C shematski i fotografija

U pravilu kada kupujemo industrijskog robota ovlašteni tehničar od strane proizvođača dužan je osposobiti robotski sustav što znači povezati sve kablove koji su nužni za pravilan rad robota te provesti postupak umjeravanja. To znači da se ovaj postupak napravi jednom i nakon toga krajnji korisnik nema veze sa njime. No kako je u cilju rada upoznati se s uobičajenim radom robota proći ćemo elemente kućišta s kojima se susrećemo u normalnim okolnostima rada.

Na slijedećoj slici prikazana je prednja projekcija kućišta IRC5 C s opisanim funkcijama sklopki i tipkala.

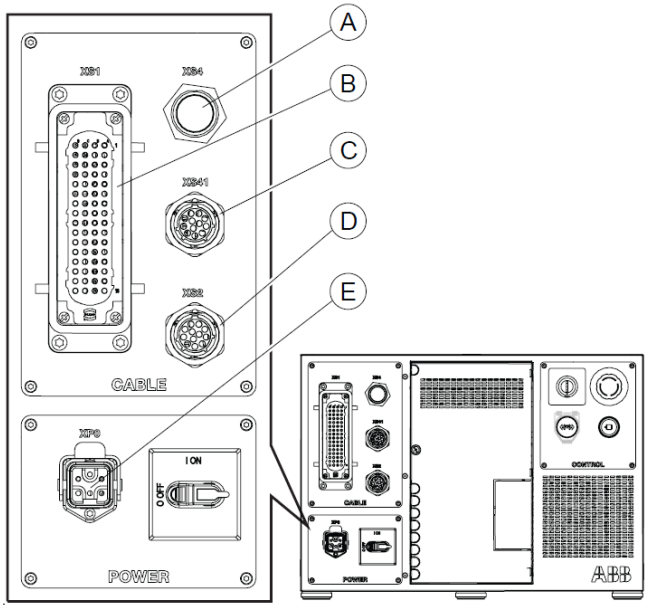


Slika 5 – Prednja projekcija IRC 5 C [3]

Oznaka	Opis
A	Glavna sklopka napajanja
B	Tipkalo za paljenje i gašenje aktivnih kočnica
C	Grebenasta sklopka za odabir načina rada (Auto/Manual/Manual+maks. brzina zglobova)
D	Tipkalo za omogućavanje motora na zglobovima
E	Sigurnosna gljiva

Tablica 4 – Opis elemenata sa slike 5

Ovo su dijelovi kućišta koji se koriste u uobičajenom radu s robotom. IRB 120, a s time i IRC5 C nema tvornički instaliranu proceduru gašenja sistema. Robota isključujemo tako da ga dovedemo u željenu poziciju te jednostavno okrenemo glavnu sklopku napajanja u OFF poziciju. Sustav uključujemo tako da sklopku okrenemo u ON poziciju. Posebnu važnost ovdje ima tipkalo za paljenje i gašenje aktivnih kočnica na motorima. Iako ćemo rijetko koristiti ovu funkciju važno je znati da dok je robot upaljen i spojen na kontroler pritiskom ove tipke možemo ugasiti kočnice na motorima kako bismo ručno doveli zglobove u željeni položaj. Ovaj postupak može biti opasan zato što se u trenutku gašenja kočnica zglobovi „ruše“ pod silom svoje težine te može doći do oštećenja ako ne koristimo adekvatne potpore. [3]



Slika 6 – Prednja projekcija IRC5 C sa detaljem [3]

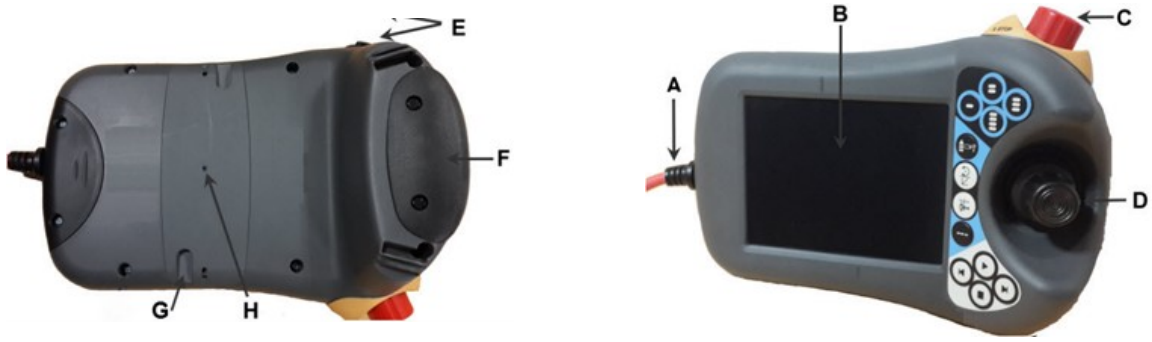
Oznaka	Opis
A	Priključak za FlexPendant

B	Izlazi napajanja za motore pojedinih zglobova i dodatni podatkovni kanali
C	Dodatni SMB konektor
D	SMD konektor za robota
E	Glavni priključak za napajanje sa mreže

Tablica 5 – Opis elemenata sa slike 6

Zadnja stvar koja nam je potrebna kako bi mogli raditi s robotom je neka vrsta sučelja sa kontrolerom. ABB svoju napravu za ručno upravljanje robotom naziva FlexPendant, radi se o industrijskom tabletu sa zaslonom na dodir kojim šaljemo komande na IRC5 C kontroler.

Općenito ime ovakvog uređaja je privjesak za učenje, svi proizvođači industrijskih robota dijele ovu ideju no razlikuju se po načinu izvedbe i nazivu. [3]



Slika 7 – FlexPendant [3]

Oznaka	**Opis**
A	Konektor
B	Zaslon na dodir
C	Sklopka u slučaju nužde
D	Gljivica za upravljanje
E	USB utor
F	Tro-kontaktna tipka
G	Olovka za zaslon
H	Tipka za resetiranje

Tablica 6 – Opis elemenata sa slike 7

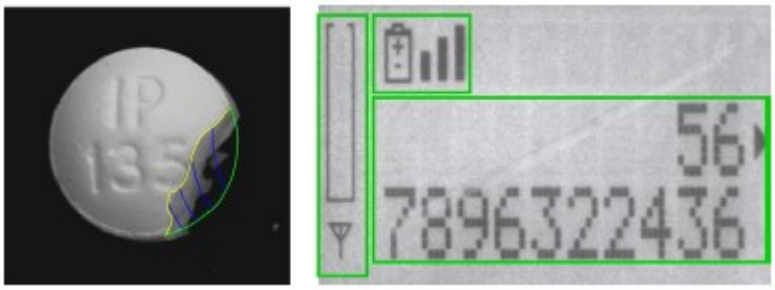
2.2. Industrijska kamera

Kamera koju koristimo u okolišu robota je prva generacija industrijskih kamera proizvođača Cognex. Radi se o Cognex 7402 Gen1 kameri koja uzima slike rezolucije 1280 x 1020 piksela. U tablici ispod dane su tehničke informacije ove kamere:

Podatak	**Vrijednost za kameru**
Programska memorija	512 MB
Radna memorija	256 MB
Rezolucija slike	1280 x 1024
Brzina zatvarača	Od 16 µs do 950ms
Potrošnja energije	24V, 2A
Masa	220g
Kućište	Aluminij
Radna temperatura	Od 0° do 45°
Spremišna temperatura	Od -30° do 80°
Maksimalna sila udarca	80G
Maksimalna sila vibracija	10G od 10Hz do 500Hz

Tablica 7 – Tehnički podatci kamere[4]

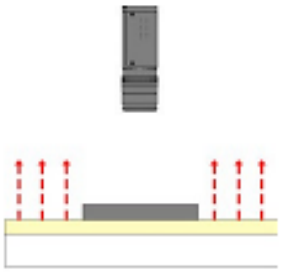
Prema AIA-i, strojni vid obuhvaća sve industrijske i neindustrijske primjene u kojima kombinacija hardvera i softvera pruža operativno vodstvo uređajima u izvršavanju njihovih funkcija na temelju snimanja i obrade slika.[4] U našem projektu, kamera služi kao glavni i jedini izvor informacija s okolinom u radu robota. Industrijski vizijski sustavi, za razliku od vojnih i akademskih, moraju zadovoljavati kriterij niske cijene, izdržljivosti, otpornosti na vanjske utjecaje, stabilnosti i preciznosti. Postavljaju se na proizvodnim linijama gdje se pokazuje potreba za brzim i preciznim mjerenjem ili prepoznavanjem uzoraka. Često zamjenjuju ljudske radnike na pozicijama gdje je posao koji se treba obaviti monoton ili previše zamarajući za čovjeka. Velika prednost industrijskog strojnog vida nad ljudskim okom je to što uz pravilan odabir kamere i konfiguraciju može prepoznati na tisuće uzoraka u sekundi.



Slika 8 – Primjer rada industrijske kamere [4]

Osvjetljenje predmeta koji nas zanima je jedna od najvažnijih stavki dobro izgrađenog vizijskog sustava. Za mnoge primjene strojnog vida nije potrebno pridavati mnogo pažnje posebnom načinu osvjetljavanja, dovoljno je samo osigurati jednoličnо osvjetljenje kroz vrijeme rada sustava, dok u nekim situacijama može biti jako teško doći do kvalitetne slike u industrijskom okruženju, u takvim slučajevima želimo iskoristiti okoliš što bolje možemo. Nekada se čak može pokazati korisno analiziranje sjene koju baca predmet koji nas zanima umjesto analiziranja samog predmeta.

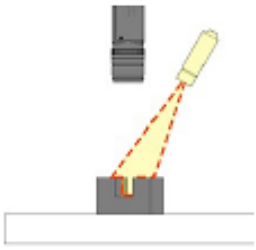
Neki načini osvjetljenja[1]:



Slika 9 - Pozadinsko osvjetljenje

Pozadinsko osvjetljenje

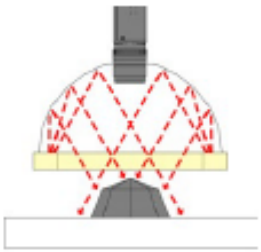
Korisno u slučajevima gdje želimo analizirati siluetu predmeta. Njime se postiže preciznije mjerenje dimenzija rubova predmeta.



Slika 10 - Strukturirano osvjetljenje

Strukturirano osvjetljenje

U nekim situacijama možemo projektirati mrežu ili neki oblik pod poznatim kutem na predmet koji nas zanima pa u odnosu na projekciju uzimati mjere preko kamere.



Slika 11 - Difuzirano osvjetljenje

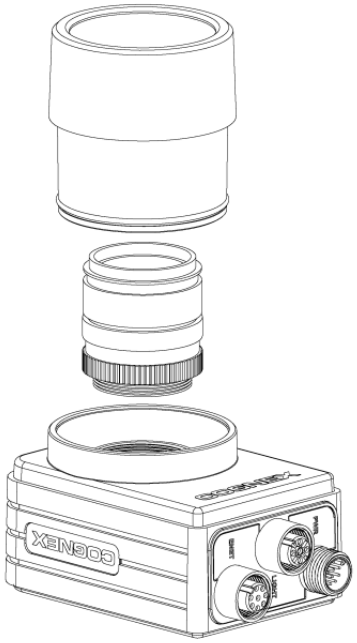
Difuzirano osvjetljenje

Najkvalitetnije osvjetljenje se postiže difuziranim izvorom svjetlosti okolo predmeta koji nas zanima. Time dobivamo čistu sliku na kojoj su zamaskirane eventualne nepravilnosti.

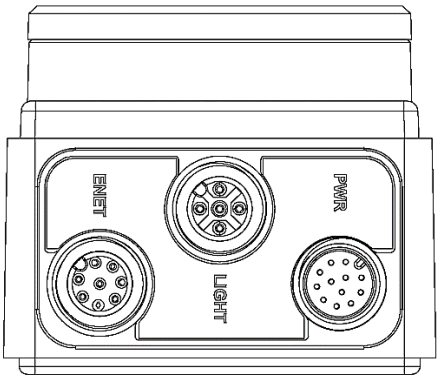
Objektiv na kameri uzima sliku i projektira ju na senzor u kameri. Objektivi se razlikuju u kvaliteti izrade i cijeni te određuje kvalitetu i rezoluciju snimljene slike.

Sposobnost kamere da snimi ispravno osvijetljenu sliku ne ovisi samo o objektivu, već i o senzoru unutar kamere. Senzori obično koriste fotoosjetljivi sloj (CCD) ili komplementarni

metal–oksidni poluvodički (CMOS) senzor, kao tehnologiju za pretvaranje svjetlosti u električne signale. Zadatak senzora je uhvatiti svjetlost i pretvoriti ju u digitalnu sliku, balansirajući šum, osjetljivost i dinamički raspon. Slika se preko senzora formira kao matrica piksela, slabo svjetlo stvara tamne piksele, dok jako svjetlo stvara svjetlije piksele. Uz pravilan odabir objektiva i dobro postavljeno osvjetljenje važno je osigurati da kamera ima odgovarajuću rezoluciju senzora za željenu primjenu. Što je rezolucija veća, slika će imati više detalja i točnija mjerenja.



Slika 12 – Prikaz Cognex 7402 kamere[4]



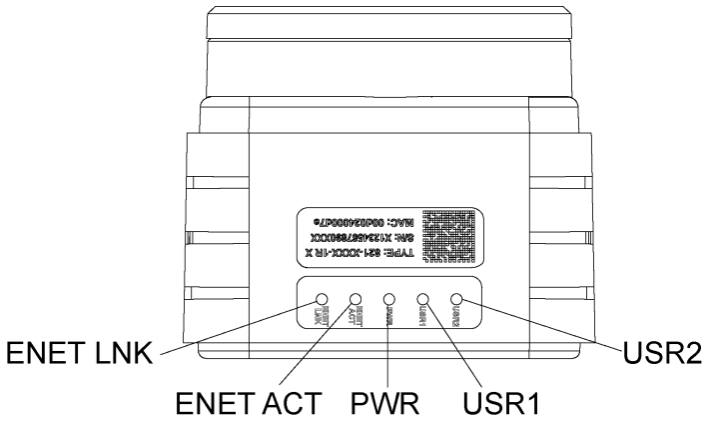
Slika 13 – Bočna strana kamere[4]

Na bočnoj strani kamere prikazanoj na slici (sl.13) vidimo tri priključka sa njihovim pripadajućim oznakama.

ENET – priključak za spajanje vizijskog sustava na mrežu. Preko ovog priključka kamera šalje informacije na upravljačku jedinicu ili neko drugo mjesto na koje želimo pohraniti slike i video.

LIGHT – priključak za spajanje dodatnog svjetla koje želimo kontrolirati preko kamere. Preko ovog priključka kamera može samostalno upaliti svjetlo u trenutku uzimanja fotografije za pravilno osvjetljenu sliku.

PWR – priključak kojim se kamera spaja na vanjski izvor napajanja. Uz napajanje, preko ovog priključka kamera dobiva naredbe za okidanje fotografije, prima ulazne signale te šalje izlazne signale i izlazne signale brzog prijenosa.



Slika 14 – Bočna strana kamere[4]

Na drugoj bočnoj strani kamere postavljene su signalne LED lampice s njihovim pripadajućim oznakama koje možemo vidjeti na slici (sl.14).

ENET LNK – lampica svijetli zeleno ako je kamera povezana na mrežu.

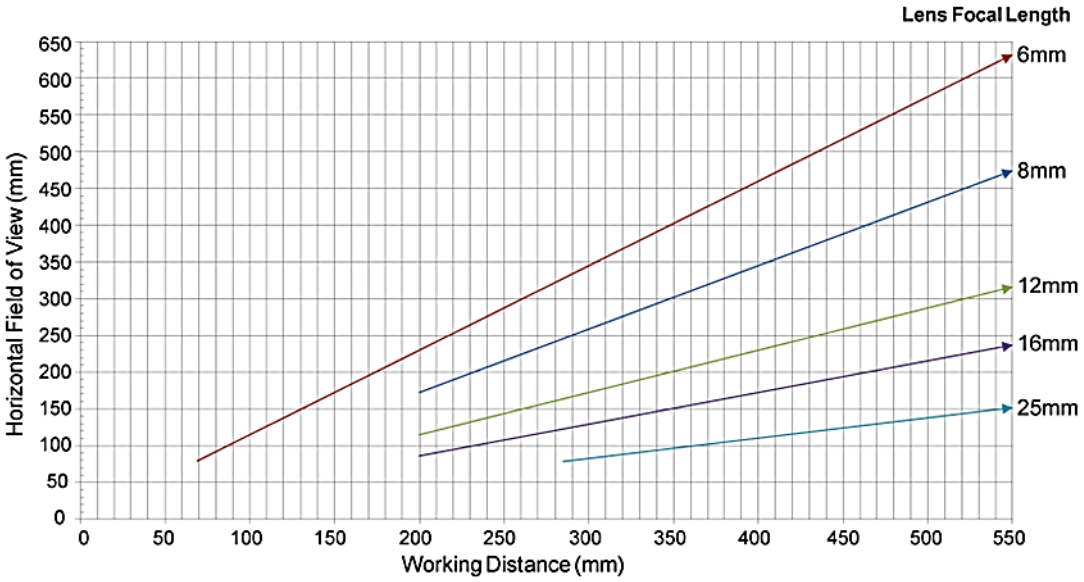
ENET ACT - lampica treperi kad se odvija prijenos podataka na mrežu.

PWR – lampica svijetli zeleno kada je kamera pravilno spojena na izvor napajanja.

USR1 – lampica svijetli crveno kad je aktivna. Ovu lampicu moguće je programirati kao izlaz – L13.

USR2 – lampica svijetli zeleno kad je aktivna. Ovu lampicu moguće je programirati kao izlaz – L12.

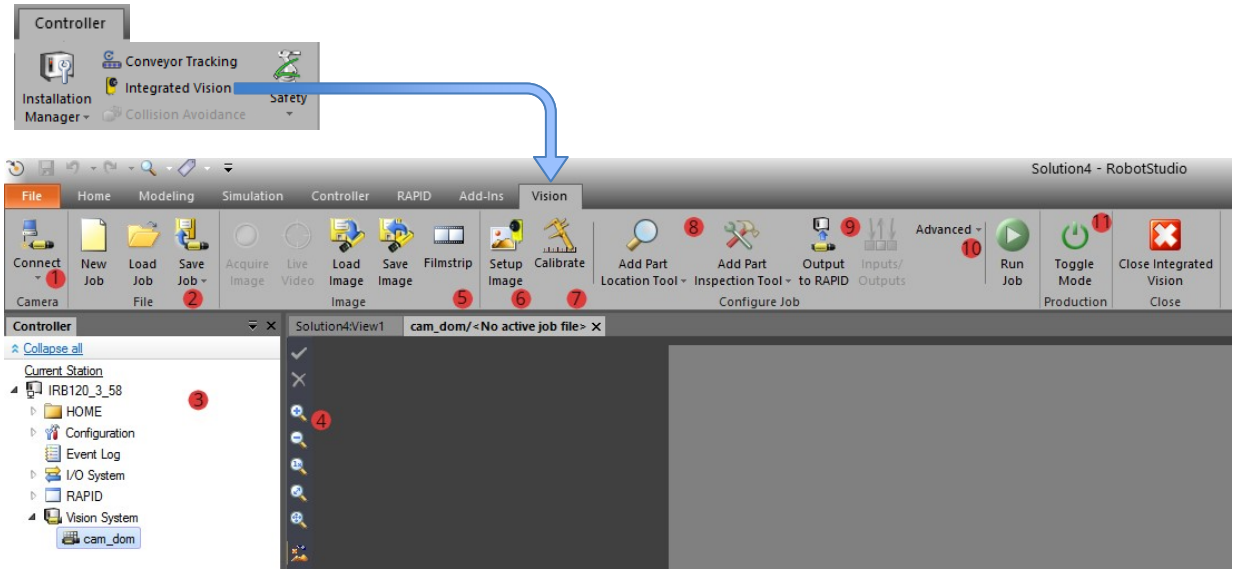
Slijedećim grafičkim prikazom prikazano je kako se mijenja veličina vidnog polje kamere u odnosu na visinu montiranja iznad predmeta koji nas zanima. Što je veća visina na kojoj je kamera montirana, to je veće vidno polje kamere.



Slika 15 – Grafički prikaz odnosa visine montaže i vidnog polja kamere[4]

3. Praktični dio – ROBOTSTUDIO

Programiranje robota s kamerom podijeljeno je na dva glavna dijela. U prvi dio spada treniranje kamere za pronalaženje predmeta koji nas zanima. Taj postupak se odrađuje korištenjem alata za analizu slike integriranih u RobotStudio. U drugom dijelu moramo se služiti Rapid programskim jezikom i napisati par linija koda kako bi inicijalizirali kameru i informacije s kamere smjestili u našu programsku rutinu. U ovom dijelu upoznat ćemo se s alatima za analizu slike i kako ih koristiti. Moramo uzeti u obzir da ABB-ov vizijski sustav može obrađivati samo jednu fotografiju po ciklusu stoga nam je bitno i pozicioniranje kamere u trenutku kada se uzima fotografiju za analizu. Kamera se može postaviti na fiksno mjesto sa kojega će uvijek uzimati slike, ali može se i montirati na zglob robota pa tako uzimati fotografije sa raznih položaja. U ovom radu ograničiti ćemo se na fiksiranu kameru, ali vrijedi znati da postoji još mogućnosti. Prije početka bitno nam je znati što točno tražimo na fotografiji koju dobijemo s kamere da na temelju oblika tog predmeta ili uzorka možemo odabrati alate koji će nam to omogućiti.



Slika 16 – Vision kartica

Oznaka	Opis	Oznaka	Opis
1	Padajući izbornik za spajanje s kamerom	**7**	Čarobnjak za umjeravanje kamere
2	Skup alata za spremanje i pozivanje „Job“ datoteke	**8**	Izbornici za dodavanja alata za određivanje položaja i analizu slike
3	Stablo podatkovnog sustava kontrolera	**9**	Izbornik za uređivanje „Cameratarget“ datoteke
4	Alati za uređivanje prikaza slike	**10**	Dodatne opcije

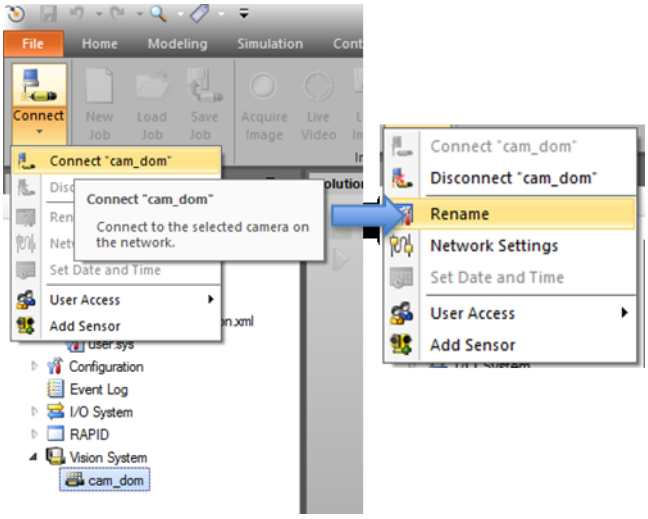
5	„Filmstrip“ alat za treniranje kamere slikama	**11**	Sklopka za manualno prebacivanje načina rada kamere
6	Alat za namještanje postavki kamere		

Tablica 8 - Opis vision kartice

3.1. Postavljanje parametara kamere

„Vision“ kartica u RobotStudio-u je mjesto na kojemu se nalaze svi alati koji su nam potrebni za dobivanje informacije s kamere kako bi ih mogli koristiti u programu kojeg će robot izvoditi. Na slici (Sl.16) je prikazano osnovno stanje Vision kartice i na njoj su označeni svi izbornici i alati koje možemo koristiti.

Prvi korak je povezivanje s kamerom, to činimo pomoću tipke „Connect“ u gornjem lijevom kutu. Početno ime kamere će vjerojatno biti njena IP adresa stoga odmah nakon povezivanja dodjeljujemo kameri ime koje će biti lako za pisati i pamtiti.[5]

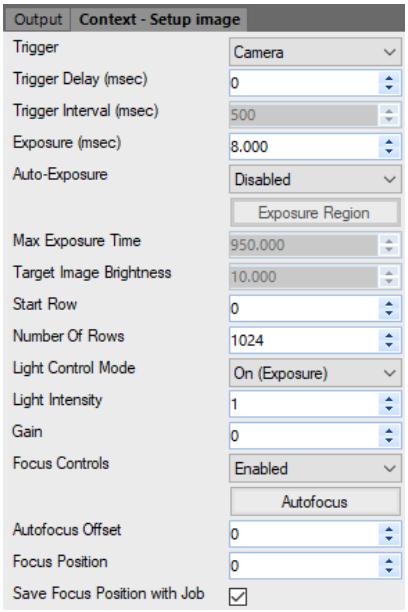


Slika 17 – Izbornik za povezivanje s kamerom

Kada smo se uspješno povezali s kamerom i dodijelili joj ime sljedeći korak je vidjeti i optimizirati kvalitetu slike koju dobivamo s kamere. To možemo učiniti pomoću alata „Setup image“ (Sl.16/6) koji će otvoriti svoj izbornik na akcijskoj traci.

Ovaj alat omogućava nam uređivanje svih parametara vezanih za kvalitetu slike koju dobivamo s kamere. Od svih ovdje ponuđenih opcija za nas su jedino bitne – Trigger (okidač) i Exposure (vrijeme ekspozicije).

Pomoću opcije Trigger biramo način okidanja fotografije. U većini slučajeva to će biti „Camera“. Kada naša kamera primi naredbu `CamReqImage,` iz programa kojeg izvršava robot, okinuti će novu fotografiju. [5].



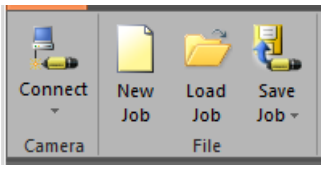
Slika 18 – Akcijska traka za Setup image alat

Vrijeme ekspozicije je jedan od osnovnih parametara koji određuje kvalitetu slike dobivene s fotoaparata ili kamere. Ekspozicija je proces osvjetljavanja senzora unutar kamere na kojem nastaje fotografija. Kvaliteta stvorene slike određena je pomoću tri glavna parametra koji zajedno čine ekspozicijski trokut: duljina ekspozicije, ISO i otvor blende. Duljina ekspozicije određuje koliko dugo će blenda biti otvorena i propuštati svjetlost na senzor, izražena u sekundama. Što je duljina ekspozicije veća, odnosno dulja to će količina svjetlosti na senzoru biti veća, a dobivena fotografija svjetlija. U mračnijoj okolini želimo povećati duljinu ekspozicije kako bi jasnije vidjeli predmet interesa na fotografiji, ali posljedica duljeg vremena ekspozicije je zamućenje pokreta na slici.

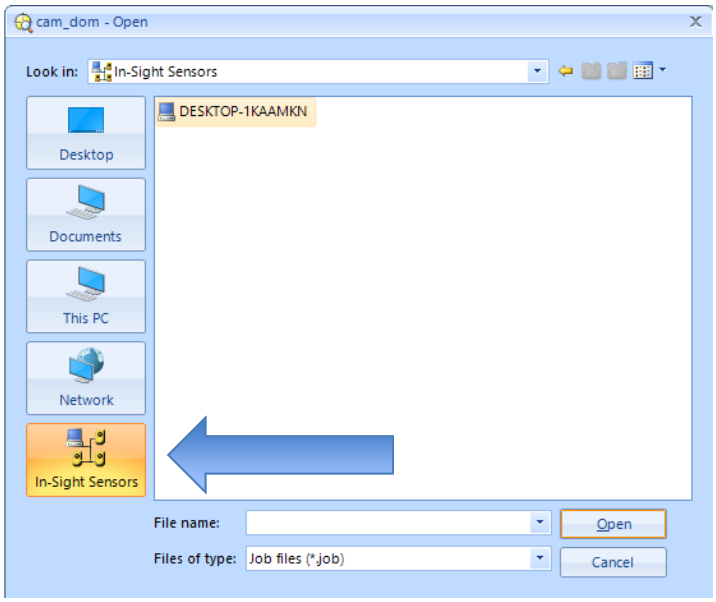
U našoj situaciji jedini parametar koji možemo kontrolirati je duljina ekspozicije koji se ovdje u milisekundama. Određivanje ovog parametra je iterativan proces. Možemo postići odličnu kvalitetu slike s jednom vrijednosti duljine ekspozicije samo da bi kasnije shvatili kako nam neko sporedno svijetlo u radnom okruženju narušava njenu kvalitetu. Stoga se vrijedi vraćati na ovaj korak kroz rad s kamerom kako bi stvorili sliku koja najviše odgovara našim trenutnim uvjetima. Za početak odredimo vrijednost za koju jasnu vidimo uzorak za umjeravanje pod kamerom. Nakon umjeravanja slobodno se vratimo na ovaj korak kako bi postavili vrijednost ekspozicije za koju će predmet koji nas zanima biti najbolje vidljiv. Promjena vrijednosti ekspozicije neće poremetiti kvalitetu odrađenog postupka umjeravanja kamere.[5]

3.2. Stvaranje „Job“ datoteke

Konfiguracija kamere i svi njeni parametri zajedno u RobotStudio-u nazivaju se „Job“ (eng. posao) datoteka. U njoj se pohranjuju parametri za kvalitetu slike, postavke umjeravanja i svi alati koje ćemo kasnije dodati u „posao“ kamere[5]. Možemo imati više spremljenih job datoteka, ali u danom trenutku samo jedna job datoteka može biti učitana u kameru. Prije početka stvaranja novog programa preporuča se stvaranje nove job datoteke kako bi krenuli s „čistog papira“. Job datotekom se upravlja sa „File“ grupom naredbi na vision kartici u RobotStudio-u.



Slika 19 – Grupa naredbi za spremanje job datoteke

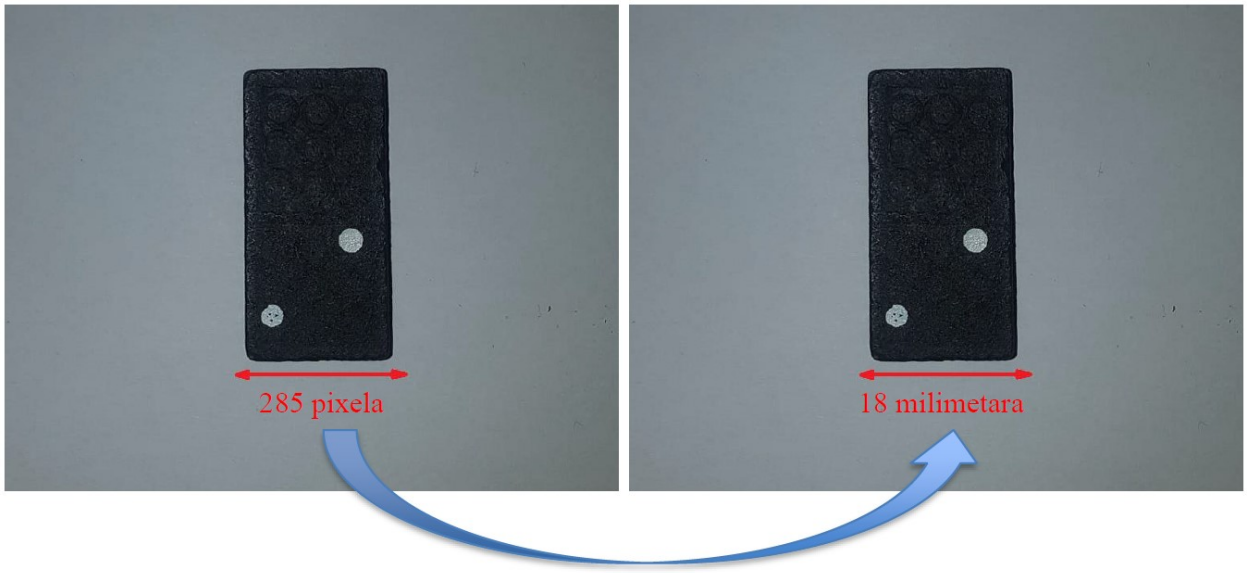


Slika 20 – Navigiranje do memorije kamere

Kasnije u radu, tijekom pisanja programa za kameru spominjat će se učitavanje job datoteke, da bi ju učitali job datoteka treba biti spremljena u memoriju kamere. Memoriji kamere možemo pristupiti na „In-Sight Sensors“ kategoriji u prozoru za stvaranje ili spremanje job datoteka. Pri imenovanju job datoteke preporuča se kratko i jasno ime bez dijakritički znakova npr. „**domino_job“.** Prije nastavka rada kameru prebacujemo u „Program mode“ (Sl.16/11).

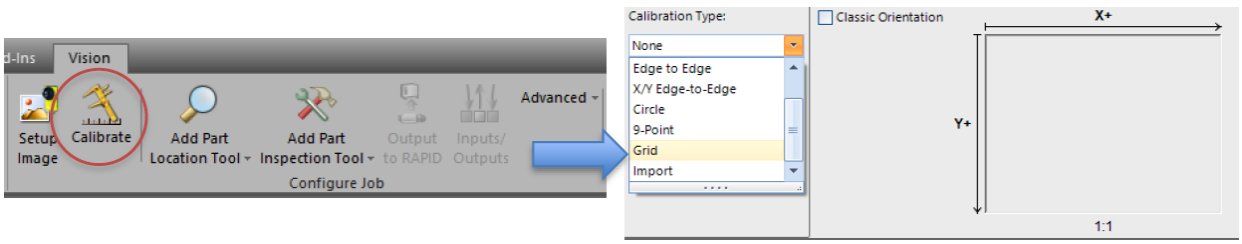
3.3. Umjeravanje kamere

Nakon što smo povezali upravljačko računalo s kamerom provesti ćemo postupak umjeravanja kamere. Umjeravanjem kamere omogućavamo programu da nauči odnos između slike i stvarnosti. Umjeravanje se odvija u dva glavna koraka – umjeravanje kamere i postavljanje zajedničkog koordinatnog sustava.



Slika 21 – Razlika informacija sa slike prije i poslije umjeravanja

Kada bi koristili kameru koja nije umjerena za traženje nekog objekta, informacije koje bi dobili od kamere odgovarale bi koordinatnom sustavu u kojem su mjerne jedinice pikseli na slici. Da bi takve koordinate pretvorili u korisne podatke moramo naći poveznicu između piksela na slici i milimetara u stvarnom prostoru. To je prvi korak procesa umjeravanja, za našu kameru ovaj postupak je integriran u RobotStudio, a možemo ga započeti s „Vision“ kartice pritiskom na ikonu „Calibrate“.

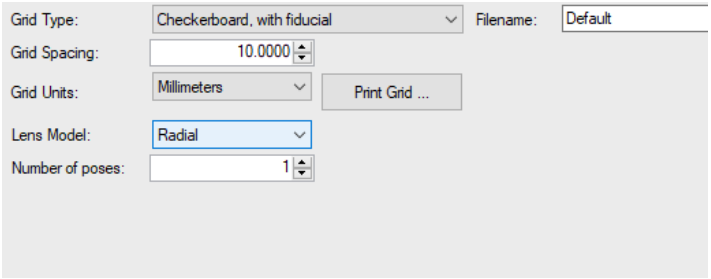


Slika 22 – Početak umjeravanja

Odabirom opcije za umjeravanje pokrećemo čarobnjak koji nas vodi kroz postupak. Koraci se trebaju napraviti poštujući redoslijed. Prvi izbor koji moramo napraviti je vrsta postupka za umjeravanje. Postoje razni načini kojima možemo doći do dobrog rezultata, ali za naš slučaj koristiti ćemo opciju „GRID“ iz padajućeg izbornika. Ovdje također vidimo da možemo učitati i od prije spremljenu datoteku koja sadrži informacije o postupku umjeravanja.

U padajućem izborniku „Grid type“ odabiremo koju vrstu uzorka za umjeravanje ćemo koristiti. U osnovi sve opcije ovdje nude slične rezultate i način kako do njih dolaze. Za kameru je bitno da vidi rešetku jednakih kvadrata, u slučaju da nemamo pristup printeru može se koristiti i obična šahovska ploča samo se mora izmjeriti veličina kvadrata. Mi ćemo koristiti opciju „Checkerboard, with fiducial“. Ova opcija za umjeravanje koristi šahovsku ploču s veličinama kvadrata od 10 milimetara i pomaže nam odrediti ishodište koordinatnog sustava pomoću oblika na sredini. Za svaku opciju koju izaberemo možemo odmah isprintati uzorak za umjeravanje koji nam je potreban odabirom na tipku „Print Grid…“ Ostale opcije služe nam za ispravljanje distorzije slike i njih u ovom koraku nećemo mijenjati [5].

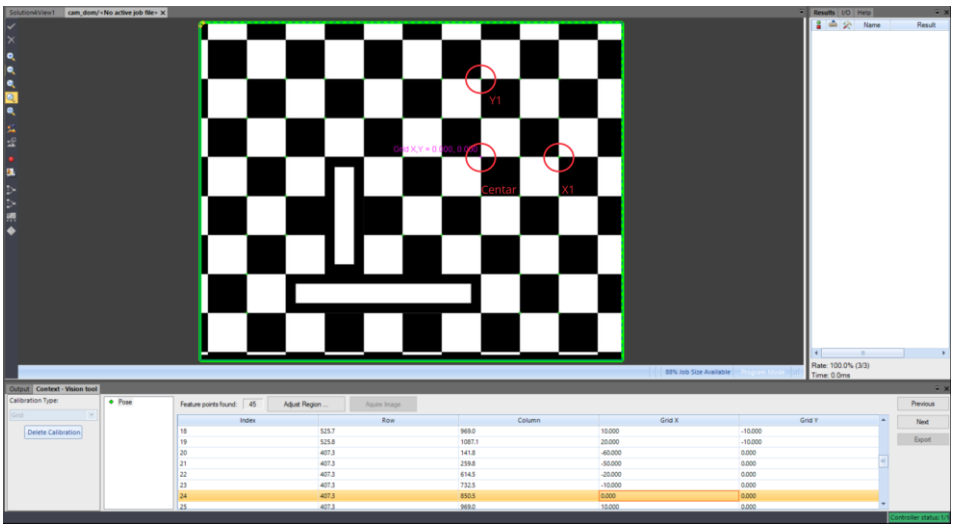
Postupak počinjemo s postavljanjem uzorka za umjeravanje na ravnu površinu direktno ispod kamere, da bi si olakšali posao možemo otići u izbornik „Setup Image“ (Sl.16/5) i pokrenuti prikaz slike u realnom vremenu kako bi lakše postavili rešetku u pravilan položaj. Rešetku ne pomičemo iz ovog položaja do kraja cijelog postupka umjeravanja.



Slika 23 – Postavke umjeravanja

Kada smo postavili rešetku prelazimo na sljedeći korak čarobnjaka.

Čarobnjak je sada prepoznao našu rešetku i označio je sve točke na njoj. U tablici točaka sada je potrebno pronaći točku ishodišta, te po jednu točku na x i y osi, pa zabilježiti ili zapamtiti njihove položaje u stvarnom svijetu. Te će nam točke biti potrebne u sljedećem koraku u kojem ćemo upariti koordinatni sustav slike i koordinatni sustav robota.



Slika 24 – tablica točaka umjeravanja

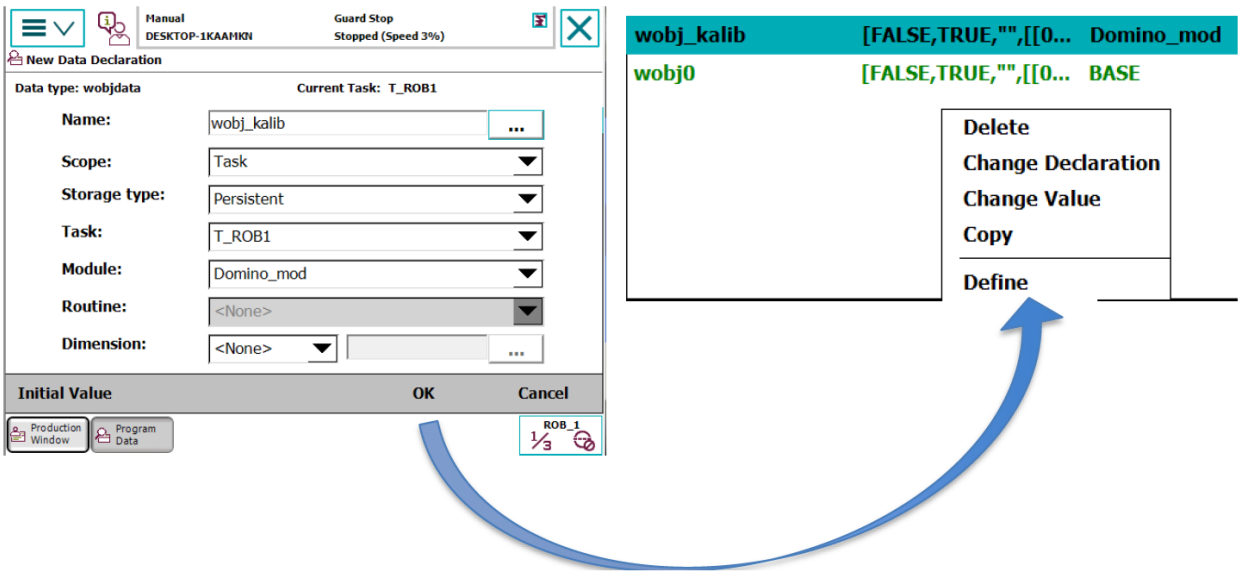
Slijedeći korak čarobnjaka pokazuje koliko je dobro odrađeno umjeravanje, također imamo opciju spremiti datoteku s podatcima o umjeravanju za buduće korištenje. Ako je ocjena umjeravanja zadovoljavajuća možemo prihvatiti promjene i nastaviti na drugi korak.

3.4. Usklađivanje koordinatnih sustava s robotom

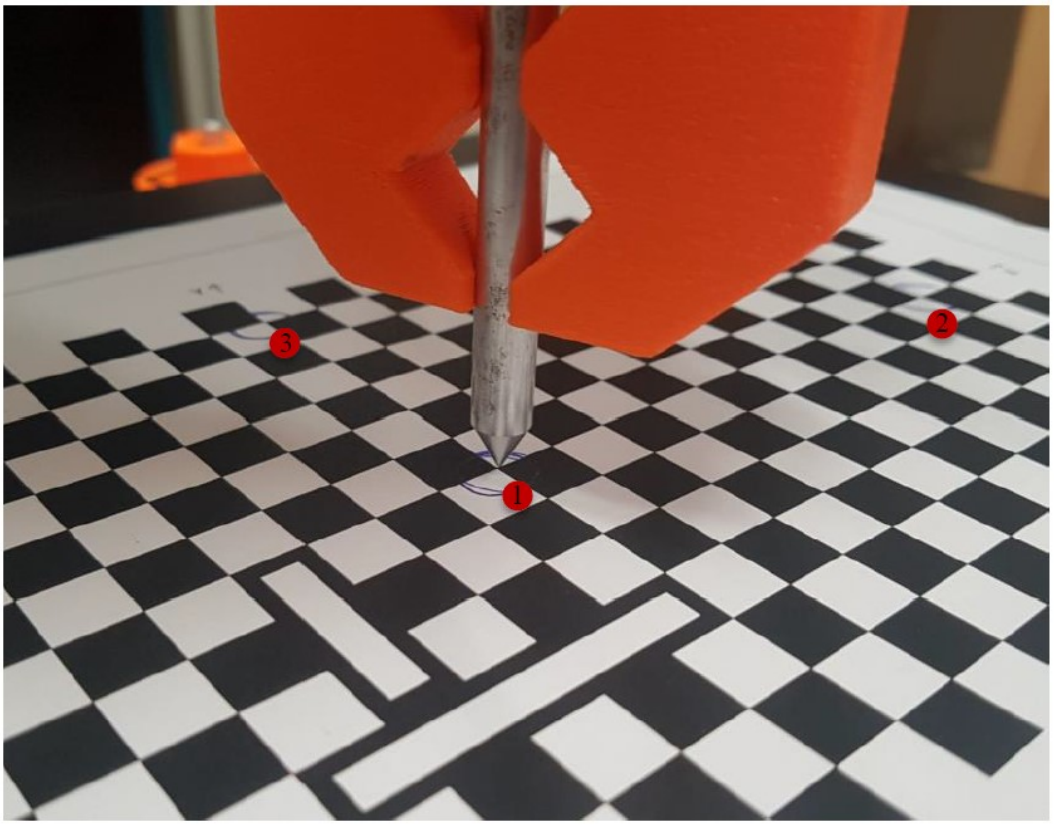
Umjeravanjem kamere odredili smo okvir slike iz kojega kamera može analizirati informacije, odredili smo položaj koordinatnog sustava fotografije tako da kamera sada zna prevoditi udaljenost izraženu u pikselima u udaljenost izraženu u milimetrima. Drugi korak postupka umjeravanja je povezivanje radnog prostora kamere s radnim prostorom robota. Da bi robot znao gdje u prostoru se nalazi točka čije koordinate šalje kamera moramo stvoriti zajednički koordinatni sustav kako bi kamera i robot mogli pravilno komunicirati. Proces stvaranja novog WorkObject-a se izvodi pomoću privjeska za učenje.

Kreiramo novi WorkObject te ga definiramo tako da se poklapa sa koordinatnim sustavom koji smo odredili na kameri pomoću tri točke čije smo pozicije zapamtili iz postupka umjeravanja.
Novi WorkObject na FlexPendantu kreiramo kroz izbornik: ProgramData/wobjdata/New…

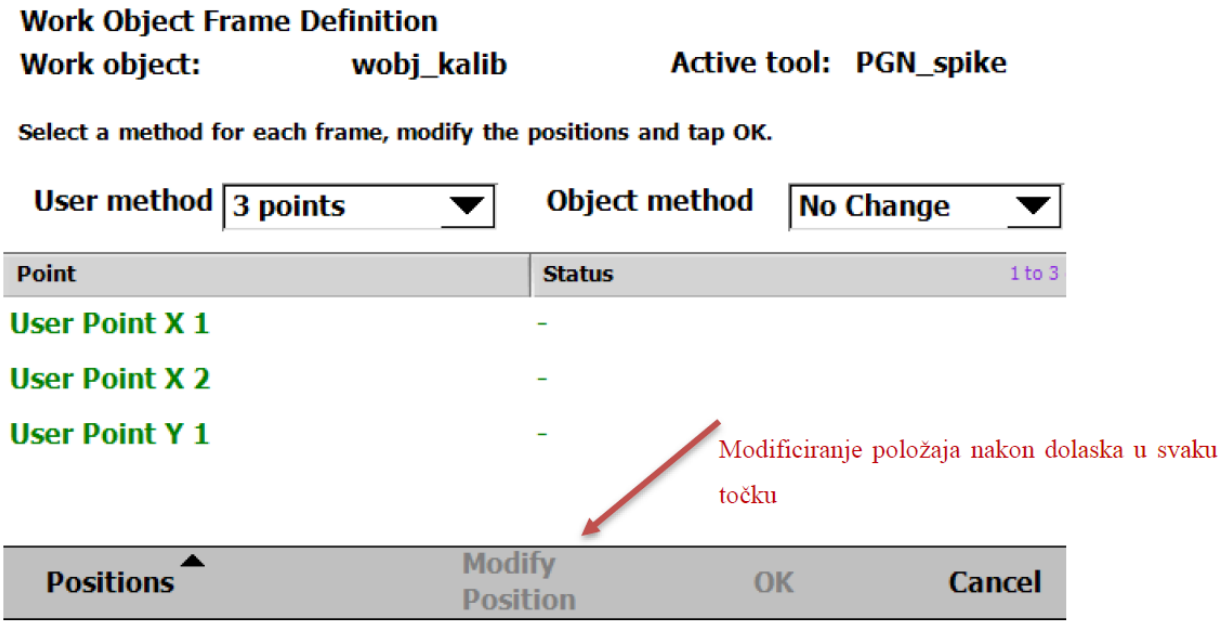
Nastavak procesa prikazan je slikama.



Slika 26 – prikaz postupka za izradu novog WorkObject-a



Slika 27 – Vođenje robota u točke za umjeravanje



Slika 28 – Modificiranje položaja točaka za umjeravanje

Nakon ovoga proces umjeravanja je gotov i možemo maknuti uzorak za umjeravanje sa pogleda kamere.

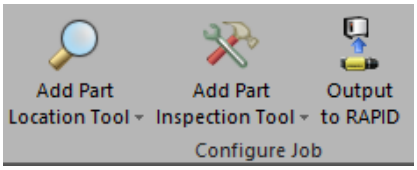
Napomena, umjeravanje kamere vrijedi za udaljenost na kojoj su bili kamera i uzorak za umjeravanje u trenutku umjeravanja, ako mijenjamo njihov položaj morat ćemo ponovno umjeriti kameru, ovo vrijedi i za postavljeni WorkObject-a na robotu zato što se Z os zajedničkog koordinatnog sustava određuje pomoću WorkObject-a na robotu.

3.5. Rad s alatima vizijskog sustava

Kada sumiramo posao koji smo do sada napravili – povezivanje s kamerom, uređivanje parametara slike, stvaranje job datoteke i umjeravanja kamere – možemo vidjeti da je to proces kojeg treba ponoviti svaki puta kada kreiramo posao za kameru u novom okolišu. Cilj dosadašnjeg rada je pružiti kameri što bolju kvalitetu slike kako bi informacije koje dobivamo s nje bile što točnije i lakše obradive.

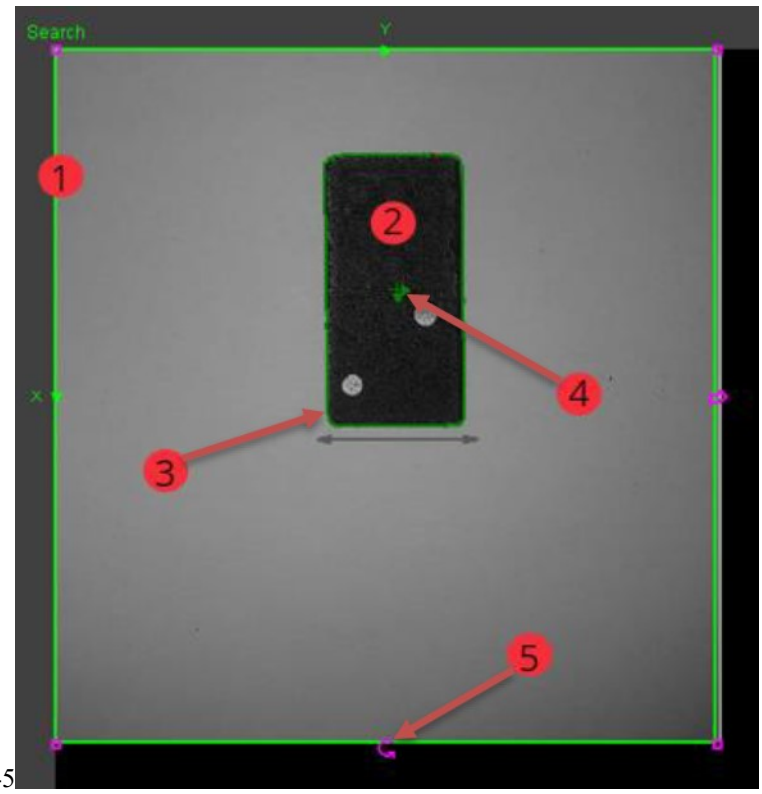
Sada se možemo upoznati sa zanimljivim alatima kojima analiziramo sliku s kamere. Alati koji se nude u sklopu integriranog vizijskog sustava su vrlo moćni i pomoću njih se mogu izvesti razne operacije. Za početak, alati za obradu slike dijele se u dvije kategorije i nalaze se na „configure job“ grupi vision kartice:

1. Alati za određivanje položaja predmeta koji nas zanima
2. Alati za inspekciju predmeta koji nas zanima



Slika 29 – Kategorije alata

U većini slučajeva cilj rada s kamerom je pronaći koordinate točke u prostoru na koju želimo poslati robota da izvrši nekakav posao. Osim ako nam se predmet od interesa uvijek pojavljuje na istom položaju na slici, uvijek prvo postavljamo alat za lociranje predmeta. Ova skupina alata prepozna trenirani oblik i locira ga na slici te nam kao povratnu informaciju šalje njegove koordinate u prostoru. Kada prepozna trenirani oblik na slici na njegovim rubovima stvara okvir koji se naziva „Fixture“. Ovo je prikazano na slici 30. Kasnije možemo pozicionirati druge alate u odnosu na rezultat alata za lociranje predmeta što će se pokazati vrlo korisno u rješavanju problema koji se pojavljuju u radu s kamerom.

45 

Slika 30 – Dijelovi pogleda alata za određivanje položaja

Oznaka	Opis
1	Područje traženja objekta
2	Traženi objekt
3	Okvir/ “fixture“ alata za određivanje položaja
4	točka čije koordinate dobivamo
5	Ručice za rotaciju i skaliranje područja za traženje

Tablica 9 – Opis elemenata sa slike 30

Integrirani vizijski sustav nudi nam više opcija kod odabira alata za određivanje položaja. Biramo onaj koji najbolje opisuje naš traženi objekt.

Alati za određivanje položaja razvrstavaju se u sljedeće skupine[6]:

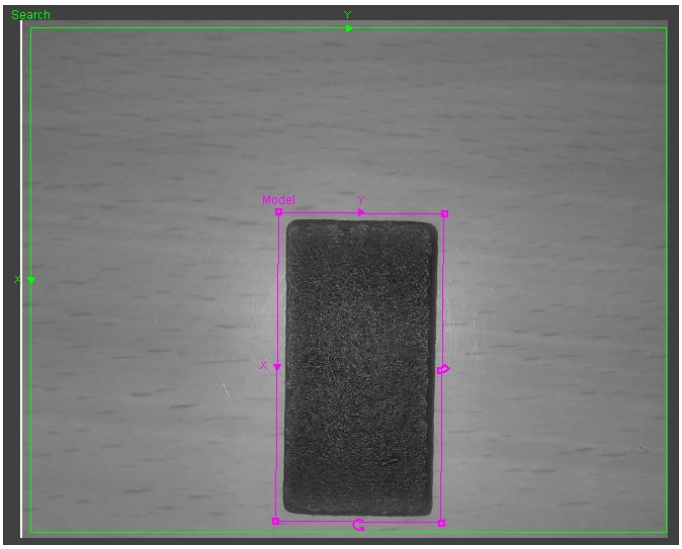
1. Pattern – traži trenirani uzorak
2. Blob – traži specifične nakupine tamnih ili svijetlih piksela
3. Edge detection – prepoznaje rubove na slici
4. Edge intersection – prepoznaje sjecišta rubova na slici
5. Circle – prepoznaje kružnice na slici

6. Compute Fixture – može se koristiti za prilagođeno računanje rezultata korištenjem rezultata drugih alata i matematičkih operacija

Krajnji rezultat svih alata za određivanje položaja je jednak, stvoriti okvir nad traženim oblikom/dijelom i prijaviti njegovu prisutnost i koordinate. Izbor alata za određivanje položaja je također iterativan proces pa se do onog najboljeg dolazi eksperimentiranjem i iskustvom. Kao primjer bolje ćemo proučiti alat za traženje uzoraka „Pattern“.

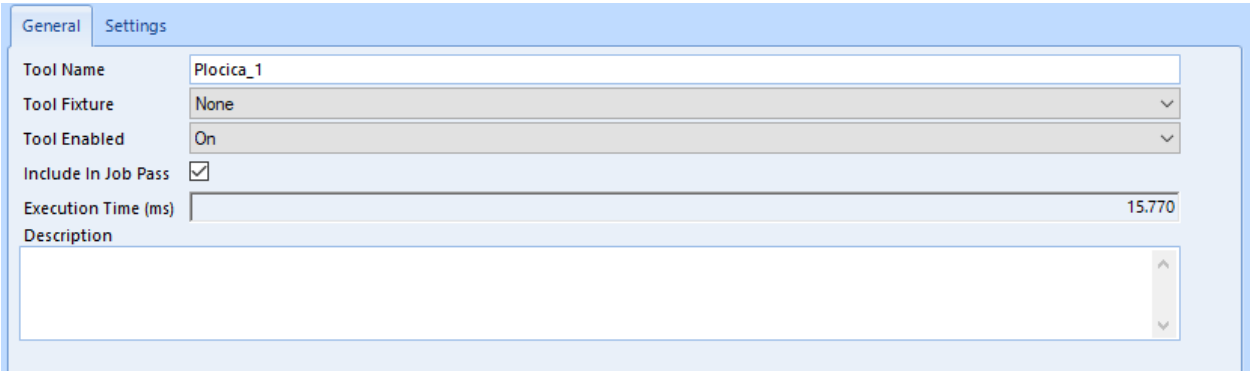
3.5.1. Pattern i PatMax pattern alati za određivanje položaja

Alat dodajemo iz njegovog pripadajućeg izbornika te pratimo upute njegovog čarobnjaka. U izborniku nam se nudi više različitih alata za traženje uzoraka, razlikuju se po broju uzoraka koje mogu naći i algoritmu traženja. Ovdje ćemo opisati alat „PatMax Pattern (1-10) with SortPatterns“. Nakon odabira alata u prozoru kamere stvaraju se dva pravokutnika. „Search“ pravokutnik određuje područje u kojemu će se odvijati traženje predmeta na slikama. Važno ga je pozicionirati tako da se objekt kojeg tražimo na slikama uvijek nalazi unutar njegovih granica. Ako tražimo samo jedan objekt koji je dobro osvjetljen i postoji veliki kontrast između njega i podloge slobodno ostavimo ovaj pravokutnik preko cijelog prozora kamere. Drugi, „Model“ pravokutnik postavljamo na predmet kojeg želimo naučiti prepoznavati. [6]



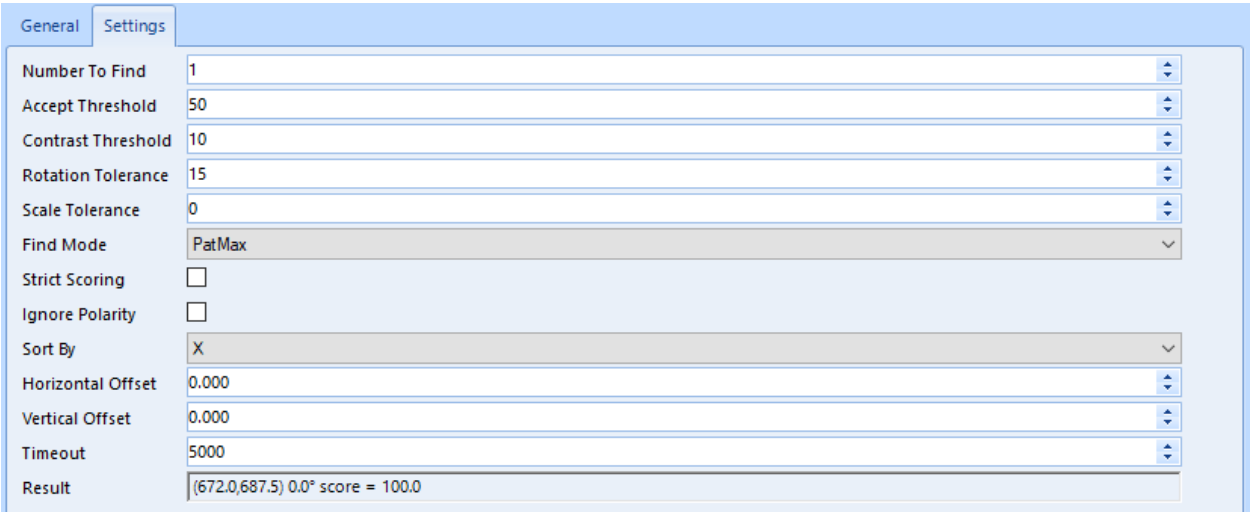
Slika 31 – Primjer učenja predmeta koji nas zanima

Ako tražimo samo jedan predmet kroz cijeli program, ne moramo se brinuti o optimizaciji alata, ali ako tražimo mnogo predmeta na jednoj slici korisno je smanjiti područje traganja na što manju površinu kako bi algoritam imao što manje podataka za procesiranje, a time bi se skratilo vrijeme potrebno da pronađe trenirani predmet. Na jednom i drugom okviru možemo podešavati dimenzije i geometriju tako da što više odgovara našem slučaju. Kada smo trenirali predmet interesa, alat se dodaje u popis korištenih alata na desnoj strani prozora i možemo mu podesiti opcije.



Slika 32 – Primjer dodanog alata

Prije testiranja novo dodanog alata možemo postaviti neke opcije kako bi se dobila bolja slika o funkcioniranju alata. Na kartici općenitih postavki možemo preimenovati alat za kasnije lakše snalaženje, također možemo postaviti okidač za naš alat na jedan od ulaznih signala ako ne želimo da je aktivan cijelo vrijeme. Dalje, na kartici postavki možemo postavili toleranciju na rotaciju treniranog predmeta. Po standardnom, rotacijska tolerancija postavljena je na 15 stupnjeva što znači da ako rotiramo, u ovom slučaju domino pločicu, za 15 stupnjeva u lijevo ili desno ona će još uvijek biti prepoznata s ovim alatom, ali sve izvan tog intervala smatrati će se ne važeće. Stoga opciju „**Rotation Tolerance**“ postavljamo na vrijednost 180 kako bi pločica bila prepoznata pod bilo kojim kutem zato što nam točan kut rotacije pločice u ovom trenutku nije bitna stavka. Prozor postavki alata je element kojeg dijele svi alati koje dodajemo u naš



Slika 33 – Prozor s postavkama alata

skup alata za kameru. Osim ove opcije često ćemo u radu s ovim, a i s drugim alatima mijenjati još neke stavke u ovom prozoru. „**Scale tolerance**“ parametar mijenja osjetljivost na promjenu veličine traženog predmeta, a izražen je u postotcima. Ovu vrijednost povećavamo u slučaju kada imamo komade koji nisu potpuno identičnih fizičkih proporcija, ali svejedno želimo da ih program prepoznaje kao identične. Suprotno vrijedi ako imamo komade koji se razlikuju po fizičkim proporcija i ne želimo da ih program prepoznaje kao identične.

Sad smo spremni za prvo testiranje našeg alata za prepoznavanje domino pločica. Predmet ćemo pomaknuti ili ga rotirati prije nego uzmemo novu sliku kako bismo se uvjerili da alat funkcionira. Ako alat ima problema sa prepoznavanjem promijenjene slike moramo ga dodatno optimizirati korak po korak. Prvo što treba potvrditi je da se predmet nalazi unutar okvira za traženje. Slijedeće, možemo povećati osjetljivost na skaliranje, no ako još uvijek ne dobivamo zadovoljavajuće rezultate možemo smanjiti prag prolaznosti („**Accept Threshold**“). Alat pri svakom analiziranju slike rezultatima alata dodjeljuje neku vrijednost kao ocjenu kvalitete rezultata analize. Rezultat alata sa početne slike predmeta s koje smo trenirali alat ima ocjenu 100 kao najbolja što može biti, sve kasnije slike s tim predmetom na njemu uspoređuju se s početnom kako bi alat zaključio dali je predmet kojeg tražimo na njima. Ako nemamo dobre uvjete u kojima kamera radi (osvjetljenje, vibracije) ili postoji varijacija između izgleda predmeta kojeg tražimo, rezultat alata na tim slikama dobit će lošiju ocjenu. Usprkos tome, nama možda ne smeta što je ocjena rezultata lošija ako svejedno prepoznaje predmet koji nas zanima, stoga ako smanjimo vrijednost praga prolaznosti i taj lošiji rezultat proći će naš prag i uzeti se kao točan.

Ovaj alat nudi i traženje više inačica istog predmeta te sprema njihove koordinate u tablicu, to možemo vidjeti s kartice „Table“ unutar postavki spomenutog alata. Rezultati (inačice

traženog predmeta) bit će spremljeni odabranim redoslijedom te se na njih kasnije može pozvati u programskom toku ili pri povezivanju s alatima za inspekciju. Kasnije ćemo se vratiti na ovaj dio i vidjeti kako se koristi.

Alati za određivanje položaja Blob, Blobs(1-10), Pattern, Patterns (1-10), PatMax Pattern, PatMax Patterns (1-10) i PatMax Patterns (1-50) postavljaju se istom procedurom kao i opisani alat.

3.5.2. Alati za inspekciju

Alati za inspekciju su svoja kategorija alata koji nam pomažu detaljnije analizirati sliku ili predmet. Unutar RobotStudio-a nudi nam se pregršt alata koji će nam biti korisni u bilo kojoj situaciji se našli. Navest ćemo kategorije alata za inspekciju da bi se stvorila slika mogućnosti ovog sistema. Ako želimo koristiti neki od mnogih alata koji nisu korišteni u ovom radu, dodatne informacije o tome kako ga implementirati možemo pronaći u dokumentu za pomoć integriranog u Vision karticu RobotStudio-a. Bitno je znati da se većina alata može međusobno povezati i nadovezivati, što znači da jedan alat može obrađivati rezultate dane od drugog alata. Ova mogućnost stvara set snažnih alata sposobnih za rješavanje mnoštva industrijskih problema.

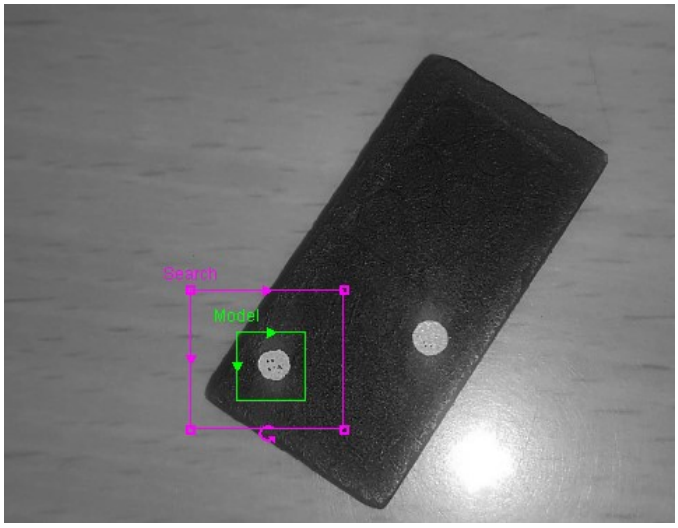
Alati za inspekciju slike dijele se u sljedeće kategorije[6]:

1. Alati za umjeravanje
2. Alati za prebrojavanje
3. Alati za traženje defekata
4. Geometrijski alati
5. Identifikacijski alati
6. Filteri slike
7. Matematički i logički alati
8. Alati za mjerenje
9. Alati za konstrukciju
10. Alati za očitavanje prisutnosti

Svaki od inspekcijskih alata nudi svoje mogućnosti, kako bismo ih bolje upoznali objasnit ćemo rad s jednim od alata za prebrojavanje.

3.5.3. Alat za prebrojavanje uzoraka – Patterns i umrežavanje alata

Započinjemo s dodavanjem novog inspekcijskog alata u naš sustav u kojem već imamo postavljen alat za određivanje položaja – PatMax Patterns (1-10). Kada dodamo inspekcijski alat prvo što trebamo učiniti je kao i prije odrediti uzorak koji će alat tražiti.

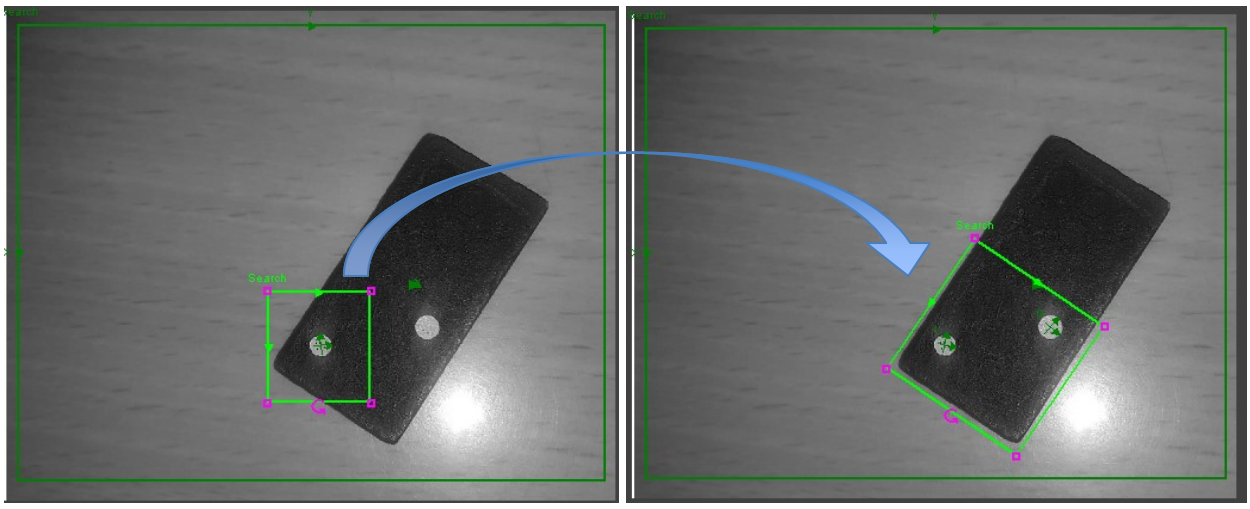


Slika 34 – Primjer treniranja alata za prebrojavanje

U ovom slučaju želimo da nam alat za inspekciju prebrojava broj kružića na domino pločici pa mu kao model zadajemo jedan od kružića za koji mislimo da je dobro uočljiv i time će biti dobar početni model. Sljedeći problem je - gdje na slici postaviti okvir za traženje. Ne možemo ga postaviti preko cijelog ekrana jer bi tada očitali broj kružića na cijeloj pločici, a mi želimo dobiti odvojene podatke s brojem kružića na jednoj i na drugoj strani pločice. Isto tako ne možemo ga postaviti preko cijelog ekrana ako u budućnosti želimo analizirati veći broj domino pločica. U ovakvim situacijama pokazuje se korisnom jedna od mogućnosti koje nam nudi integrirani vizijski sustav RobotStudio-a. Kako smo prethodno prikazali, alat za određivanje položaja postavio je okvir – fixture preko treniranog modela domino pločice. Sada taj okvir možemo iskoristiti kako bi alat za inspekciju „zakačili“ na njega i time pratili poziciju pločice na svakoj slici.[6]

Ta se radi tako da prvo potvrdimo treniranje modela kružića zatim uđemo u postavke inspekcijskog alata i postavimo „**Plocica_1.Fixture**“ („Naziv alaza za lokaciju“.Fixture) kao stavku iz padajućeg izbornika za „**Tool Fixture**“ postavku. Ovime smo odredili poziciju okvira za traženje u odnosu na fixture okvir alata za određivanje položaja – „Plocica_1“ na svakoj slici.

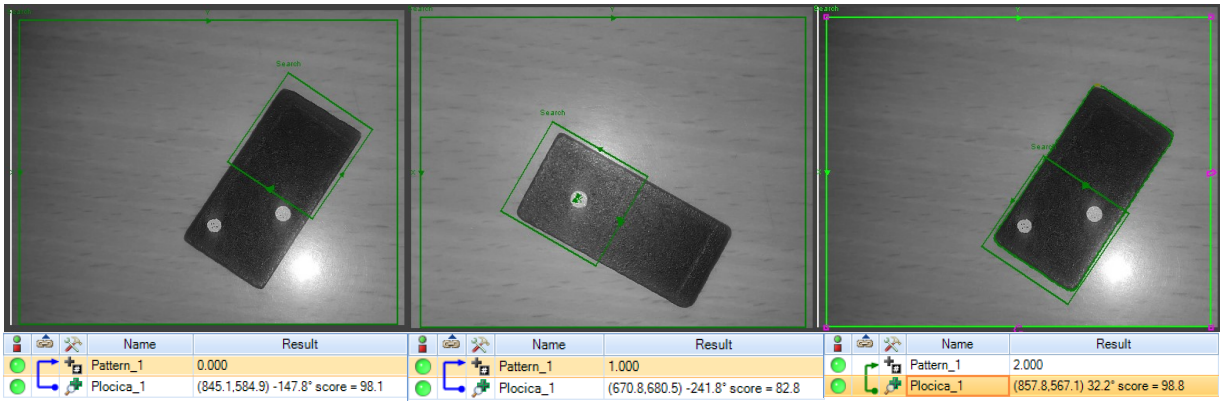
Kao i za prošli alat možemo postaviti vrijednost osjetljivosti na kut zakreta za naučeni model, ali uz ovaj alat dobivamo još jednu karticu u postavkama – „**Range Limits**“. Inspekcijski alati kao i alati za određivanje položaja daju nekakav rezultat, a na kartici „Range limits“



Slika 35 – Primjer pozicioniranja search okvira

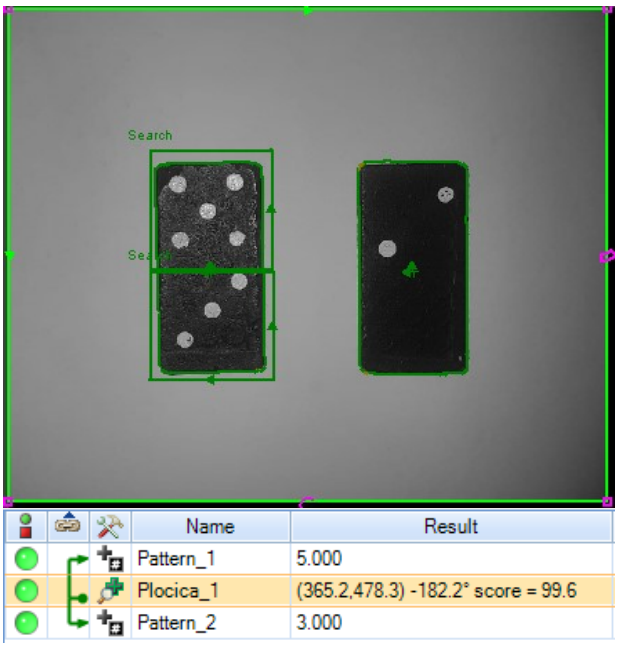
možemo postaviti uvjete koji se trebaju ispuniti da taj rezultat bude prolazan, ovo će biti bolje objašnjeno u daljnjem poglavlju – **3.5. CameraTarget varijabla,** za sada postavit ćemo donju granicu na 0 i gornju na 6 kao broj točkica koje možemo naći na jednoj strani domino pločice. Sada možemo odrediti poziciju „Search“ okvira tamo gdje želimo da bude na svakoj slici domino pločice, ovaj proces je bolje objašnjen slikom (Sl.36).

Kada postavimo search okvir na željeno mjesto ono ostaje zapamćeno i pomiče se u skladu s fixture okvirom na koji je alat povezan, sada to možemo i testirati pomicanjem ili mijenjanjem pločice pod kamerom. Kako mijenjamo položaj pločice pratimo rezultate naših alata u prozoru rezultata na desnoj strani ekrana.

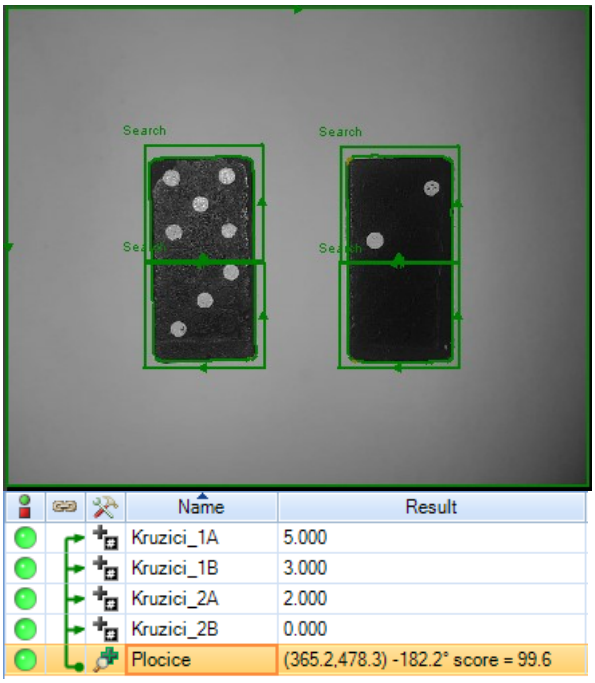


Slika 36 – Primjeri alata u radu i pripadajući rezultati

Sada vidimo kako naš alat prati pozicije domino pločice i uvijek prebrojava broj kružića samo na jednoj strani. Ako želimo prebrojiti i drugu stranu pločice možemo dodati još jednu inačicu alata za inspekciju i „zakačiti“ ju, kao i prvu, na drugu stranu fixture okvira pločice. Korisno je znati da alate možemo brisati i kopirati po želji što uvelike olakšava posao.



Slika 37 - Primjer s dva alata za prebrojavanje uzoraka

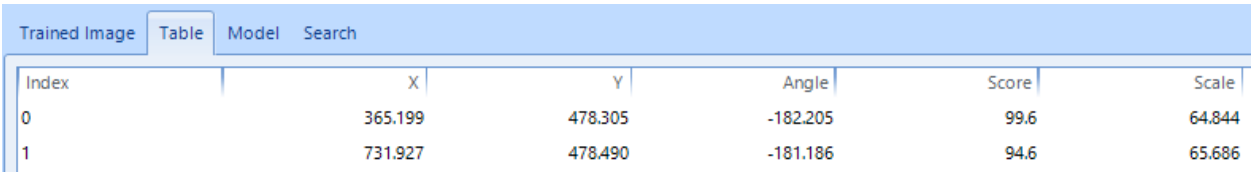


Slika 38 – Primjer s četiri alata i dvije pločice

Na slici(sl.38) možemo vidjeti kako izgleda kada su spojena dva alata za prebrojavanje na isti fixture okvir domino pločice. Isto tako, može se vidjeti kako je alat za određivanje položaja pronašao još jednu domino pločicu i na nju također postavio fixture okvir. Ovo se dogodilo zato što smo za ovu sliku promijenili vrijednost „Number to Find“ postavke „Plocica_1“ alata za

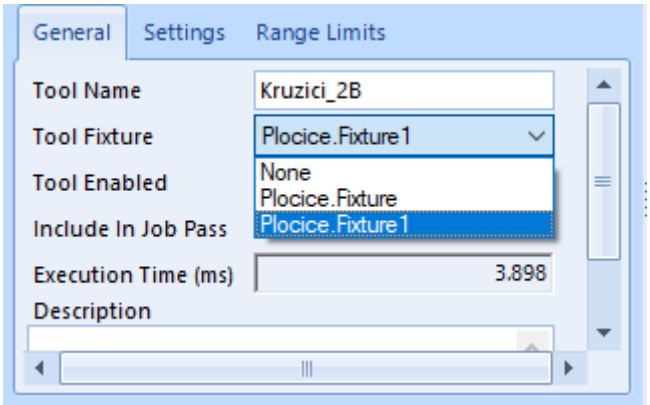
određivanje položaja sa 1 na 2. Alat je automatski prepoznao još jednu pločicu i označio ju. Da bi prebrojali broj kružića na drugoj pločici trebamo dodati još dva alata za prebrojavanje i postaviti ih da prate drugu pločicu. Na slici(Sl.39) može se vidjeti kako izgleda kada su spojena 4 alata za prebrojavanje, 2 na fixture okvir svake domino pločice. Uz to što smo dodali alate za prebrojavanje na drugu pločicu, preimenovali smo dosada korištene alate za lakše snalaženje.

U nastavku pokazujemo kako alati za prebrojavanje broja kružića druge pločice znaju na koji „Fixture“ okvir se „zakačiti“. Pogledajmo sada kako izgleda tablica rezultata u postavkama alata za određivanje položaja – „Plocice“.



Slika 39 – izgled tablice alata „Plocice“ s vrijednosti postavke „Number to Find“ - 2

Za svaku pločicu koju alat za određivanje položaja pronađe i označi će se ovdje pojaviti pripadajući rezultat. Bitno je naznačiti da se rezultati u tablici numeriraju od broja 0 na dalje. Sada, da bi inspekcijski alati namijenjeni drugoj pločici dobili točan položaj na kojem trebaju tražiti kružiće trebamo im proslijediti pripadajući rezultat. U ovom slučaju bit će to drugi po redu rezultat, u tablici označen brojem 1 i to ćemo napraviti putem postavki inspekcijskih alata – Kruzici_2A i Kruzici_2B.



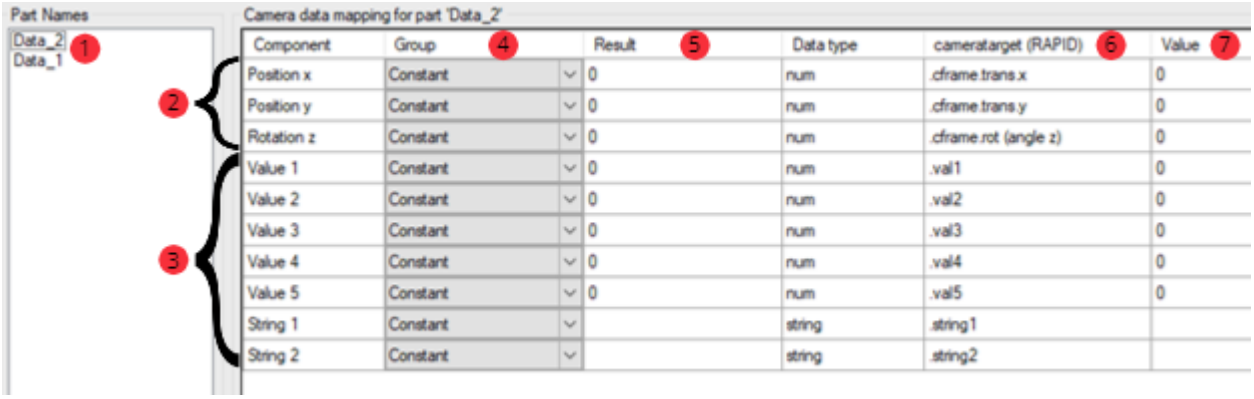
Slika 40 – Prikaz povezivanja „fixture“ okvira

Na slici(Sl.41) vidimo da se nakon pronalaska druge pločice ponudila još jedna opcija fixture okvira na koji možemo „zakačiti“ naše inspekcijske alate. Ako odaberemo opciju „Plocica.Fixture1“ za svaki od dva nova alata što smo ih dodali da bi prebrojali broj kružića na

drugoj pločici, njihovi „Search“ okviri pratit će promjenu položaja druge pločice i ako ih pozicioniramo kako je prikazano na slici (sl.39) trebali bi postići isti rezultat. Ovaj proces možemo ponoviti za svaki novi inspekcijski alat koji dodajemo.

3.6. CameraTarget varijabla

Putem izbornika „Output to Rapid“ možemo se upoznati s izgledom `CameraTarget` varijable. Ova varijabla nam služi za prenošenje svih informacija koje dobijemo putem kamere u naš programski kod. Iz nje vučemo koordinate, broj nađenih uzoraka i sve ostale informacije koje nam mogu poslužiti u slaganju logike rada robota.



Slika 41 – Prikaz tablice za mapiranje cameratarget varijable

Oz.	Opis
1	U RobotStudio-u zvani dijelovi. Može se gledati kao inačica cameratarget varijable sa svojim imenom
2	Dio varijable u koji se spremaju pozicijski rezultati, po redu – Pos_x, Pos_y, Rot_z
3	Dio varijable u koji se spremaju vrijednosti koje odaberemo, 5 numeričkih vrijednosti i 2 rečenice
4	Grupa označava alat s kojeg želimo spremiti vrijednost, npr. Za naš slučaj ovdje bi odabrali „Plocice“
5	Dio informacija koje nam alat pruža koju želimo spremiti na ovu poziciju
6	Izraz kojim se poziva informacija u programskom kodu
7	Vrijednost koja je spremljena na toj poziciji

Tablica 10 – Opis elemenata sa slike 35

Na slici (sl.42) možemo vidjeti kako izgleda prozor za mapiranje cameratarget varijable. Trenutno je varijabla prazna, to jest onakva je kakvu dobijemo kad prvi put otvorimo izbornik. Da bi popunili varijablu potrebno je odrediti njen sadržaj u svakom redu. Prva tri reda su uvijek osigurana za pozicijske informacije i vežu se direktno s jednim od dodanih alata za određivanje položaja. Koristeći ovo možemo povezati prva tri reda s našim alatom za lociranje domino pločica.

Component	Group	Result	Data type	cameratarget (RAPID)	Value
Position x	Plocice	Fixture.X [0-1]	num	.cframe.trans.x	[]
Position y	Plocice	Fixture.Y [0-1]	num	.cframe.trans.y	[]
Rotation z	Plocice	Fixture.Angle [0-1]	num	.cframe.rot (angle z)	[]

Slika 42 – Tablica za mapiranje cameratarget varijable

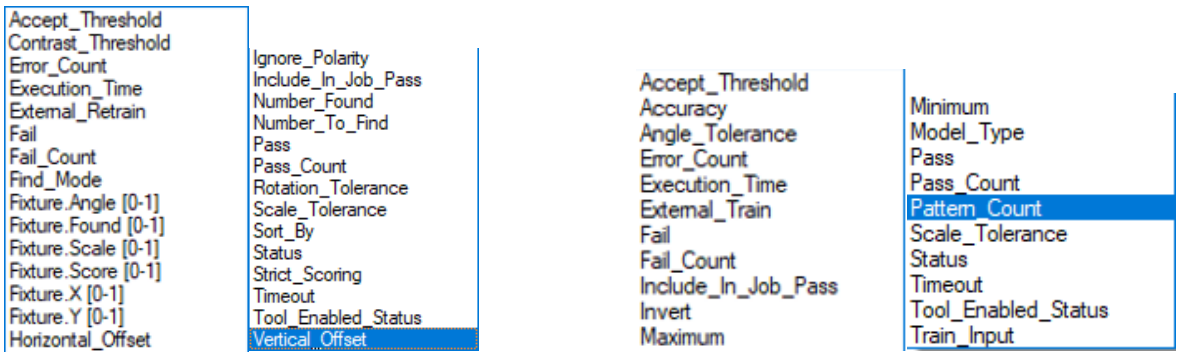
Kada smo ovo napravili, svaki put kad kamera uzme novu fotografiju i procesira njen sadržaj popunjava varijablu s dobivenim informacijama. Sva tri reda popunjavaju se paralelno s dodijeljenim informacijama, ako nema informacija ostaju prazni i pri pozivanju u program dobivamo poruku s greškom[5]. Tu dolazi do malog problema s korištenjem ove varijable s kojim se susrećemo i malo kasnije pri pisanju samog programa. Svaki put kad zatražimo rezultate s kamere putem cameratarget varijable dobivamo koordinate samo jedne pločice koju smo pronašli s alatom za određivanje položaja. Sjetimo se tablice s rezultatima alata za određivanje položaja. Nama je cilj dobiti koordinate svih pronađenih pločica ne samo jedne, stoga ćemo ovaj problem probati riješiti putem programskog koda u kasnijem dijelu rada.

Prve tri komponente cameratarget varijable često se pozivaju zajedno kao element koji se u rapid programskom jeziku naziva – „pose“ (položaj/poza). To je skupni naziv za pozicijske informacije – koordinate x i y osi s pripadajućim kutem rotacije oko z osi. Zbog ovoga je bitno da sve tri komponente dobiju jednaki broj članova, npr, ako tražimo dvije pločice povezujemo informacije istog alata na sve tri komponente kako bi sve tri imale jednaki broj informacija u sebi da ne dođe do problema pri njihovom pozivanju. Ako imamo dvije stavke u komponenti za koordinate x osi, a samo jednu u komponenti za y os može doći do problema.

Value 1	Kruzici_1A	Pattern_Count	num	.val1	[2.000]
Value 2	Kruzici_1B	Pattern_Count	num	.val2	[0.000]
Value 3	Kruzici_2A	Pattern_Count	num	.val3	[2.000]
Value 4	Kruzici_2B	Pattern_Count	num	.val4	[5.000]

Slika 43 – Sortiranje informacija sa alata

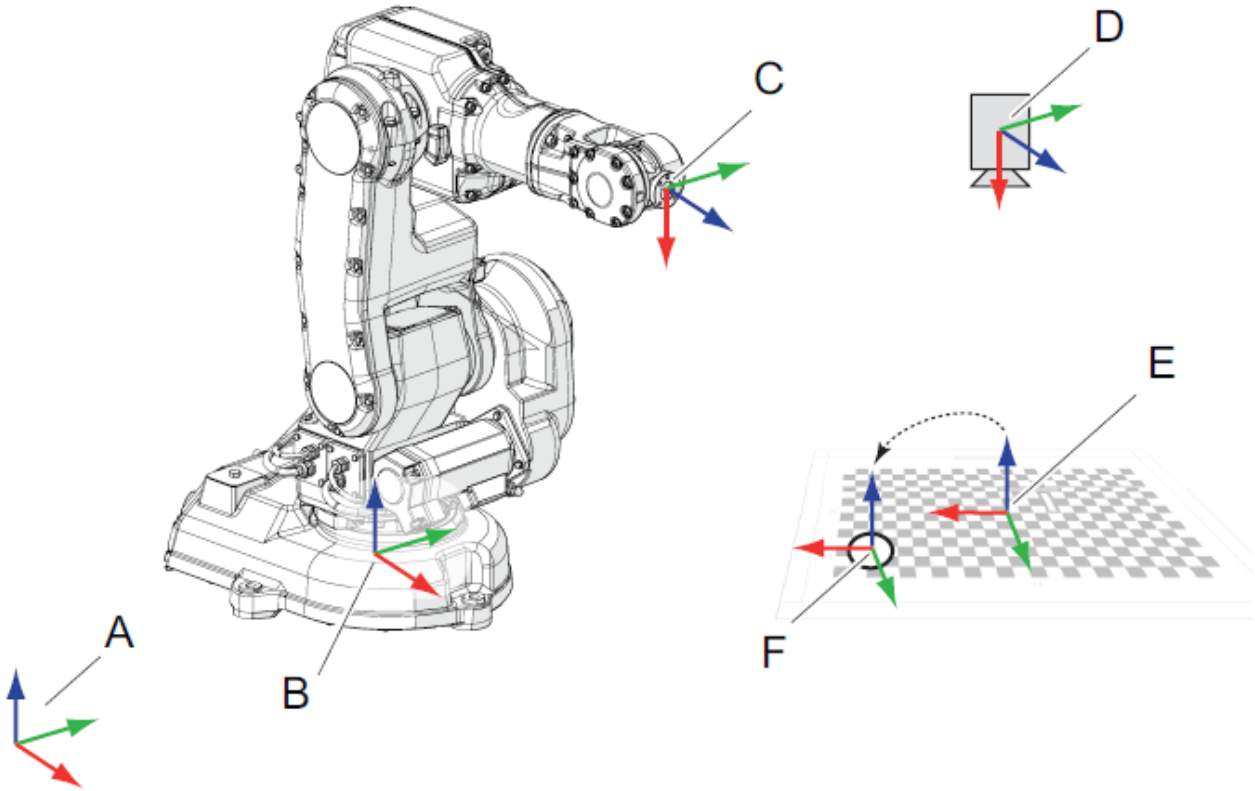
Ostatak varijable možemo popuniti s informacijama kojima želimo imati pristup tokom rada sustava. Sigurni smo da nam trebaju 4 numeričke varijable koje ćemo popuniti s brojem pronađenih kružića na svakoj strani pločica tako da ćemo njima osigurati prva 4 mjesta numeričkih komponenti od Value 1 do Value 4. Ostatak možemo ostaviti praznim, ali ćemo prikazati kakve sve informacije ovdje možemo dodijeliti.



Slika 44 – Primjer za alate za određivanje položaja (lijevo) i inspekcijske alate(desno)

3.7. Koordinatni sustavi

Prije nego što počnemo programirati sustav zadnja stvar koju moramo poznavati pri pisanju programa su koordinatni sustavi robota i kako su međusobno povezani. Najvažniji koordinatni sustavi su WORLD, BASE, tool, work object – user frame i work object – object frame.



Slika 45 – Koordinatni sustavi robota[5]

A	WORLD koordinatni sustav koji je opcionalno definirati, koristi se za rad više robota na jednom objektu
B	BASE koordinatni sustav leži u temelju robota i jedan je od osnovnih koordinatnih sustava
C	Tool koordinatni sustav ishodi u TCP-u i koristi se za navigaciju robota

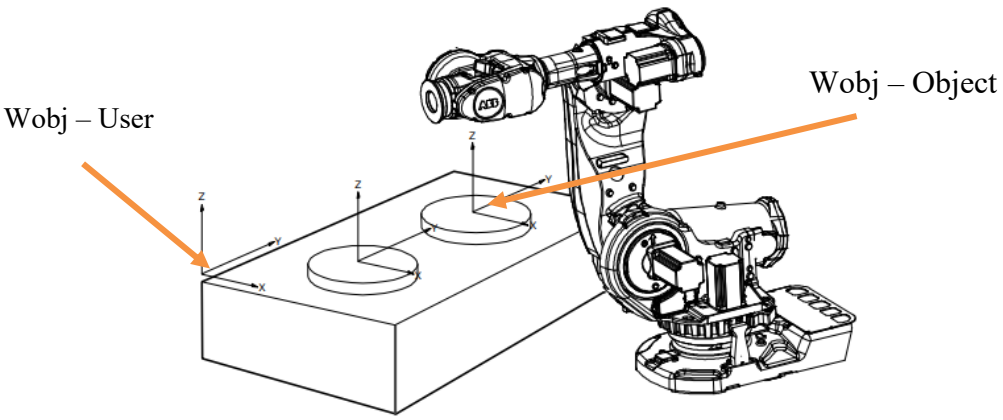
D	Kamerin koordinatni sustav koji nam nije bitan jer je kamera stacionarna i robot ne zna gdje se ona nalazi
E	WorkObject – user frame koordinatni sustav koji je umjeren s kamerom tako da leži unutar kadra kamere
F	Work Object – object frame koordinatni sustav objekta kojem određujemo položaj

Tablica 11 – Opis elemenata sa slike 43

WorkObject – user frame(dalje u radu Wobj.U frame) je koordinatni sustav koji možemo definirati kao plohu na kojoj robot manipulira nekim predmetom, često se koristi i kao pomoćni koordinatni sustav pri gibanju robota po raznim nagibima. WorkObject – object frame(dalje u radu Wobj.O frame) je koordinatni sustav koji se postavlja na sami komad kojim robot manipulira. Ove koordinatne sustave možemo bolje razumjeti ako ih primijenimo na konkretnom primjeru, u našem slučaju, pri radu s kamerom. Wobj.**U** frame koordinatni sustav definirali smo pri povezivanju koordinatnih sustava robota i kamere, leži na plohi na kojoj se nalazio uzorak za umjeravanje i kojem smo odredili središte i koordinatne osi x i y. Wobj.O frame je koordinatni sustav definiran dinamički kroz alat za određivanje položaja. Kada kamera registrira domino pločicu na slici dodjeljuje joj okvir i u centar okvira postavlja središte koordinatnog sustava. Ovaj koordinatni sustav se stoga pomiče i rotira svaki put kad kamera registrira pločicu na novoj poziciji. Da bi doveli robota do pozicije na kojoj može uhvatiti domino pločicu sve što moramo napraviti je definirati koordinate vrha alata (TCP-a) unutar ovog dinamički stvaranog koordinatnog sustava. Na ovaj način robot će uvijek doći u istu poziciju naspram domino pločice.

Kronološki ovaj proces se odvija na sljedeći način[5]:

1. Kamera dobiva sliku s domino pločicom
2. Alat za određivanje položaja prepoznaje domino pločicu
3. Alat za određivanje položaja dodjeljuje okvir domino pločici i postavlja koordinatni sustav u njegovo središte
4. Na temelju odrađenog procesa umjeravanja, kamera u milimetrima određuje poziciju središta koordinatnog sustava domino pločice u odnosu na koordinatni sustav kamere (koji je ujedno i Wobj.U frame koordinatni sustav postavljen kao zajednički koordinatni sustav kamere i robota)
5. Kamera i kontroler sinkroniziraju informacije i robot sada zna gdje se nalazi Wobj.O frame (centar domino pločice) unutar Wobj.U frame-a (radna površina)
6. Program šalje robota u točku za prihvat čije su koordinate izražene u Wobj-O frame koordinatnom sustavu

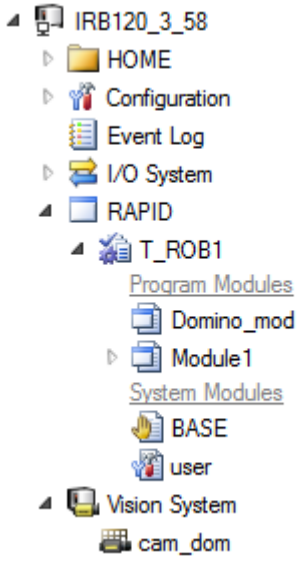


Slika 46 – Prikaz koordinatnih sustava[5]

Ova petlja je osnovni dio svakog programa kojem je cilj dovesti robota u poziciju koja je određena informacijama s kamere. Kako ovaj proces izgleda u programskom kodu bit će prikazano u sljedećem poglavlju.

4. Programiranje u RAPID-u

Nakon što imamo postavljenu cijelu pozadinu u RobotStudio-u možemo se pozabaviti programiranjem samog procesa u RAPID-u. Pisati možemo na FlexPendant-u ili na računalu. Preporuča se pisanje programa na računalu zato što je jednostavnije i preglednije, a kasnije možemo pratiti program i raditi manje preinake na FlexPendant-u. Da bi mogli sinkronizirati RobotStudio na računalu s memorijom kontrolera moramo se povezati ethernet kablom preko switcha ili direktno u računalo. Taj proces je opisan u dokumentaciji kontrolera. Kada smo povezani s kontrolerom možemo otvoriti karticu Rapid u RobotStudio-u gdje je prikazana podatkovna struktura kontrolera.



Slika 47 – Podatkovna struktura kontrolera

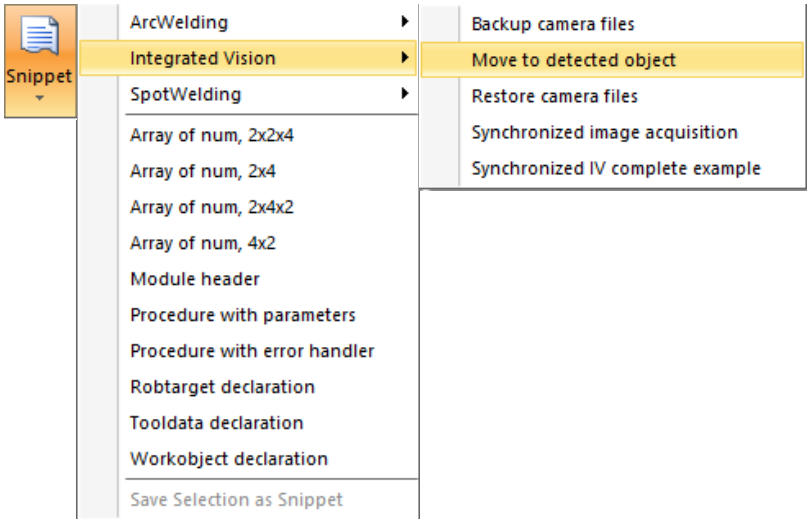
Na slici (Sl.48) je prikazana trenutna podatkovna struktura kontrolera. Stvorili smo novi modul – „Domino_mod“ pod kojim ćemo pisati procedure vezane za naš sustav.

Nakon svakog uređivanja programa potrebno je izvršiti **sinkronizaciju** s kontrolerom, a to radimo putem tipke „Apply“ u Rapid kartici sučelja. Pisanje programa može se raditi od „nule“ ako posjedujemo predznanje u pisanju Rapid programskog jezika, ali često se možemo poslužiti od prije složenim skicama koje RobotStudio nudi. Da bi učitali skicu prvo stvorimo novu proceduru unutar Domino modula preko koje ćemo pozivati naš program za sortiranje.

```
1  MODULE Domino_mod
2      !*********************************************************
3      !
4      ! Modul:  Domino_mod
5      !
6      ! Opis programa: Rutina za sortinje domino pločica.
7      !
8      ! Autor: Karlo Šarić
9      !
10     !*********************************************************
11     !
12     ! Procedura domino() - Ovo je početak našeg programa.
13     !
14     !*********************************************************
15
16     PROC domino()   !procedura preko koje pozivamo podprograme
17         Sort_domino;
18     ENDPROC
19
20
21     PROC Sort_domino() !procedura za sortiranje domino pločica počinje ovdje
22
23     ENDPROC
24
25
26
27 ENDMODULE
```

Slika 48 – Izgled početka programa RAPID-u

Skice se nalaze u padajućem izborniku „Snippet“ iz kojeg možemo navigirati do tražene skice.



Slika 49 – Put za dodavanje skice u RAPID program

Učitavanjem skice – „Move to detected object“ dodajemo ju u proceduru na mjestu postavljenog kursora.

```
!Change the job name
CONST string myjob := "myjob.job";
VAR cameratarget mycameratarget;

!Declare robtarget, workobject, tooldata and, in case the camera is
!mounted on a moving part of the robot, the imaging position.
!CONST robtarget myrobtarget:=[[100,200,300],[1,0,0,0],[0,0,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
!CONST robtarget myimagepos:=[[100,200,300],[1,0,0,0],[0,0,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
!TASK PERS wobjdata mywobj:=[FALSE,TRUE,"",[[0,0,0],[1,0,0,0]],[[0,0,0],[1,0,0,0]]];
!PERS tooldata mytool:=[TRUE,[[0,0,1],[1,0,0,0]],[1,[0,0,0],[1,0,0,0],0,0,0]];

PROC MoveToDetectedObject()
    !Change the camera name
    CamSetProgramMode mycamera;
    CamLoadJob mycamera, myjob;
    CamSetRunMode mycamera;

    !If the camera is mounted on the robot, store this position during setup
    !so that the robot may always return to this position before requesting an image.
    !MoveL myimagepos, v100, fine, tool0;

    CamReqImage mycamera;
    CamGetResult mycamera, mycameratarget;
    mywobj.oframe := mycameratarget.cframe;

    !During the first cycle, run the program until this point,
    !then jog the tool to the desired grip position and modpos myrobtarget.
    MoveL myrobtarget, v100, fine, mytool \WObj:=mywobj;
ENDPROC
```

Slika 50 – Prvobitno stanje skice

Prateći programski kod sa skice možemo podesiti tražene vrijednosti i napraviti manje izmjene kako bi program proradio. Skica nas vodi kroz najosnovniji primjer programa u kojem dobijemo točku s kamere i pošaljemo robota u nju.

Izvan procedure programa, a unutar rutine definiramo koordinate work objecta, informacije o alatu kojeg koristimo i jednu točku koju ćemo modificirati kao poziciju za prihvat predmeta.

```
TASK PERS wobjdata wobj_kalib:=[FALSE,TRUE,"",[[0,0,0],[1,0,0,0]],[[0,0,0],[1,0,0,0]]];
PERS  tooldata PGN_gripper:=[TRUE,[[0,0,1],[1,0,0,0]],[1,[0,0,0],[1,0,0,0],0,0,0]];
CONST robtarget prihvat_tocka:=[[100,200,300],[1,0,0,0],[0,0,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
!CONST robtarget myimagepos:=[[100,200,300],[1,0,0,0],[0,0,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
```

Slika 51 – Deklariranje Workobject-a i točaka u prostoru

Ako work object definiramo kroz privjesak za učenje on će automatski biti dodan u specificiranu rutinu stoga ga nije potrebno ručno dodavati. Isto vrijedi i s alatom kojeg koristimo, alat može biti globalno deklariran te ga samo treba pozvati unutar programa. „Prihvat_tocka“ je jedna točka koju smo stvorili, ali nas trenutno ne brinu njene koordinate jer ćemo ih tek kasnije modificirati. „Myimagepos“ točka ostaje komentirana iz razloga što je naša kamera stacionirana, a ova točka bi nam služila kao referenca za uzimanje fotografija ako je kamera montirana na robota.

```
CONST string myjob := "domino_sort.job";
VAR cameratarget target_plocica;
```

Slika 52 – Deklariranje varijabli

Definiramo konstantu tipa string kao ime naše job datoteke tako da nam je lakše napraviti preinake u budućnosti. Također definiramo jednu varijablu tipa `cameratarget` u koju ćemo spremati rezultate s kamere.

```
CamSetProgramMode cam_dom;
CamLoadJob cam_dom, myjob;
CamSetRunMode cam_dom;
```

Slika 53 – Naredbe za kontrolu kamere

Ove tri naredbe također spadaju u dio programa koji se ne treba izvršavati svaki ciklus već samo pri pokretanju. Prebacujemo kameru u programski način rada kako bi mogli učitati job datoteku te nakon što je učitana prebacujemo kameru u pogonski način rada.

```
CamReqImage cam_dom;
CamGetResult cam_dom, target_plocica;
wobj_kalib.oframe := target_plocica.cframe;
```

Slika 54 – Naredbe za kontrolu kamere i prosljeđivanje informacija

Tri naredbe sa Slike 55 u skici izvršavaju se jedna za drugom i mogu se promatrati kao trenutak u kojem se izvršava cijela zadaća korištenja kamere kao izvor informacija u procesu s robotom. `CamReqImage` naređuje kameri da uzme novu fotografiju i obradi njene informacije stoga moramo biti sigurni da je radna površina iz perspektive kamere jasno vidljiva i bez prepreka u trenutku njenog pozivanja. Slijedi naredba `CamGetResult` koja preslikava trenutno stanje rezultata aktivnih alata i sprema ga u našu `cameratarget` varijablu – „target_plocica“ što znači da varijabla „target_plocica“ nakon ove naredbe sadrži informacije o poziciji predmeta.
Ako je kamera pronašla traženi predmet prva tri retka varijable bit će popunjena s njegovim koordinatama i kutom rotacije u odnosu na središte zajedničkog koordinatnog sustava. Koordinate x i y te rotaciju oko z osi zajedno možemo nazivati – poza predmeta.

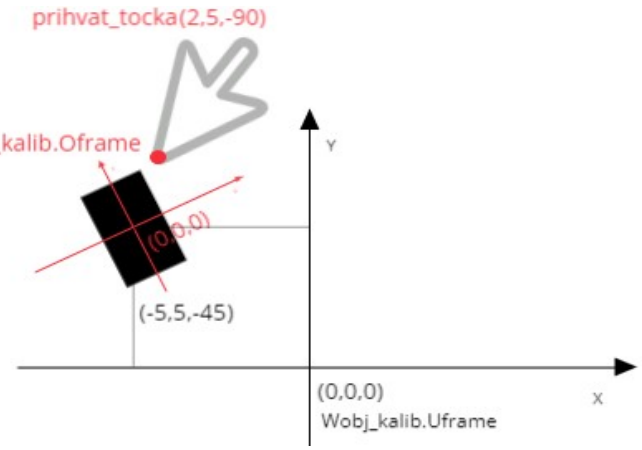
Sljedeća naredba `wobj_kalib.oframe := target_plocica.cframe;` uzima koordinate ove poze i izjednačuje ih s koordinatama wobj_kalib – object frame-a unutar našeg zajedničkog, wobj_kalib – user frame-a.
Pri prvom pokretanju programa skica nam sugerira da nakon ove naredbe postavimo točku prekida i odvedemo robota u poziciju sa koje želimo da robot prihvati traženi predmet.

```
58    wobj_kalib.oframe := target_plocica.cframe;
59    MoveL prihvat_tocka, v100, fine, PGN_gripper \WObj:=wobj_kalib;
```

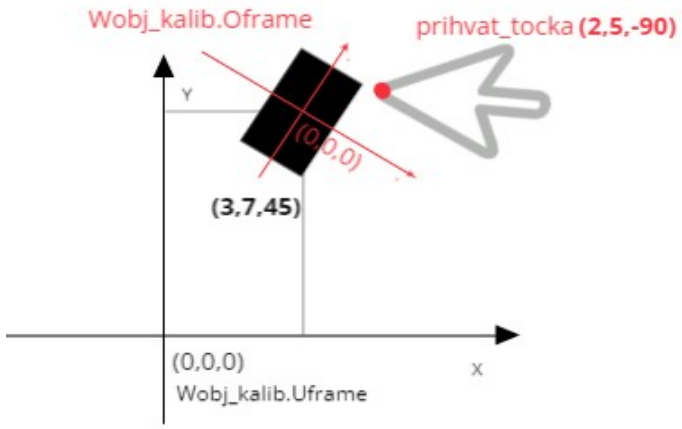
Slika 55 – Naredbe za sinkroniziranje koordinatnih sustava i odlazak u točku

Kada robota dovedemo u željenu poziciju pomoću jog-iranja preko privjeska za učenje, trebamo prebaciti work object sa „base“ na „Wobj_kalib“ te modificirati poziciju naše pomoćne točke – „prihvat_tocka“. Kada smo ovo napravili, „prihvat_tocka“ je sada određena koordinatama u wobj_kalib – object frame koordinatnom sustavu i možemo vratiti robota na mjesto sa kojega ne zaklanja pogled kamere kako bi mogli ponovno pokrenuti program. Ako još postoje nedoumice oko toga na koji način robot svaki put dolazi do točne pozicije bez da se direktno modificira točka u koju se robot šalje, slijedeće slike bi ih se trebale riješiti.

a



b



Slika 56 : a) analiza rezultata prve slike i b) analiza rezultata s pomaknutom pločicom

Primjer programa

Za kraj prikazan je program kao sinteza svih tehnika spomenutih u ovom radu. Program koristi identično umjerenu kameru s robotom kao i alate koje smo dodavali u poglavlju alata za određivanje položaja i identifikacijskih alata. Sastavljen je dio programskog koda koji je zadužen za pozadinsku logiku traženja domino pločice s zadanim brojem točkica. Program skenira strane domino pločice pojedinačno i sprema njihove vrijednosti u pomoćne varijable.

```
PROC domino()

        VAR num br_pronadeno;                    !broj nađenih plocica
        VAR num br_trazeni:=2;                !broj tockica koji trazimo
        VAR num broj_plocice:=1;              !pomocna varijabla koja prikazuje
trenutnu plocicu u skeniranju
        VAR num trazena_plocica;                     !pomocna varijabla koja na kraju
poprima vrijednost sa brojem plocice na kojem je trazeni broj
        VAR string br_pomoc;                  !pomocna string varijabla za izvrsavanje
petlje za spremanje tockica kockica
```

```
        VAR cameratarget plocica_target{3};  !array cameratarget varijable za 3 domino
kockice(1,2,3)
        VAR num tockice_b{3};                !array u koji se spremaju brojevi tockica
na pojedinoj stranici trazene plocice
        VAR num tockice_a{3};                !-||-


        CamSetRunMode cam_dom;
        CamReqImage cam_dom;
        CamFlush cam_dom;

        CamGetParameter cam_dom ,"ploc_find.Number_Found"\NumVar:=
br_pronadeno; !naredba čita parameter postavljen u "output to RAPID" matrici I upisuje
ga u varijablu


        WHILE broj_plocice <= br_pronadeno DO
!petlja za spremanje informacija o plocicama i paralelno trazenje plocice s trazenim
brojem tockica

           CamGetResult cam_dom, plocica_target{broj_plocice};
           br_pomoc:=NumToStr(broj_plocice,0);
           CamGetParameter cam_dom,"ploc"+br_pomoc+"_1.Pattern_Count"
\NumVar:=tockice_a{broj_plocice};      !dio petlja koji sprema broj tockica na svakoj
plocici na pripadajuce mjesto u varijabli
           CamGetParameter cam_dom,"ploc"+br_pomoc+"_2.Pattern_Count"
\NumVar:=tockice_b{broj_plocice};

           IF tockice_a{broj_plocice} = br_trazeni THEN
!IF petlja koja provjerava trenutnu plocicu i usporeduje ju s trazenim brojem
                TPWrite "Trazeni broj je na plocici broj: " \Num:=broj_plocice;
                mywobj.oframe := plocica_target{broj_plocice}.cframe;
                trazena_plocica:=broj_plocice;
                MoveL myrobtarget, v100, fine, PGN_Chunk \WObj:=wobj_kalib; !salje
robota na lokaciju trazene plocice
            ELSE IF tockice_b{broj_plocice} = br_trazeni THEN
                TPWrite "Trazeni broj je na plocici broj: " \Num:=broj_plocice;
                mywobj.oframe := plocica_target{broj_plocice}.cframe;
                MoveL myrobtarget, v100, fine, PGN_Chunk \WObj:=wobj_kalib; !salje
robota na lokaciju trazene plocice
                Trazena_plocica:=broj_plocice;
                ExitCycle;
            ELSE
                broj_plocice:=broj_plocice+1;
            ENDIF
            ENDIF

        ENDWHILE

        IF broj_plocice > br_found THEN                         !!ako nema trazenog
broja u zadnjem ciklusu ce broj_plocice prerasti br_found i zavrsiti petlju

            TPWrite "Od prilozenih plocica niti jedna nije s trazenim brojem";

        ENDIF

        !=================================== ...
         !…ostatak programa ovisno o okolisu

    ENDPROC
```

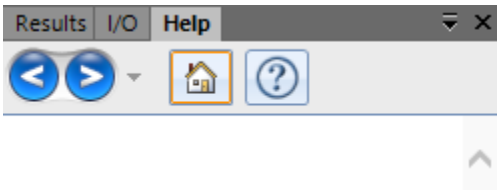
Jedna od novih naredbi koju nismo spomenuli u radu do sada je naredba `CamGetParametar` kojom se možemo služiti za čitanje informacija iz pojedinih ćelija „Output to RAPID“ matrice koju postavljamo unutar „Vision“ kartice RobotStudio-a. Sintaksa za korištenje naredbe je sljedeća:

```
CamGetParametar <ime kamere> , „<ime alata . element koji želimo čitati>„ \<tip varijable u koji spremamo informaciju> := <ime varijable> ;
```

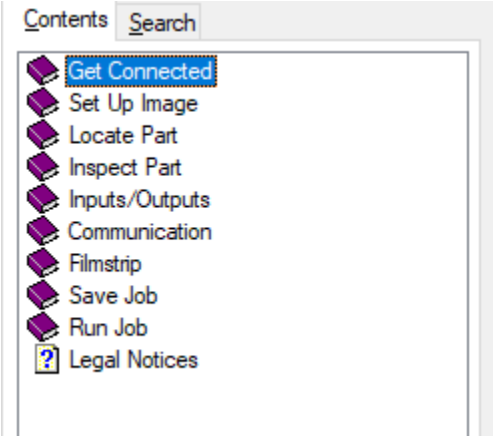
```
VAR num BROJ := 0;
VAR bool TEST := FALSE
!Broj kružića na A strani prve pločice je 3 i alat „Kruzici_2A“ je upaljen.
CamGetParametar cam_dom, „Kruzici_1A.Pattern_Count“ \NumVar:=BROJ;
CamGetParametar cam_dom, „Kruzici_2A.Tool_Enabled_Status“ \BoolVar:= TEST;

TPWrite „Vrijednost varijable BROJ je : „ \Num:=BROJ;
TPWrite „Stanje alata je: „ \Bool:=TEST;
```

ISPIS:
Vrijednost varijable BROJ je : 3
Stanje alata je : TRUE

Dodatne informacije o alatima vizijskog sustava i kako se oni koriste mogu se pronaći na integriranom pomoćnom dokumentu unutar RobotStudio-a. Njemu se može pristupiti preko kartice „Help“ u prozoru rezultata.



Slika 57 – Mjesto dokumenta pomoći



Slika 58 – Sadržaj dokumenta pomoći

Pomoćni dokument sadrži jako puno korisnih informacija o alatima vizijskog sustava koje se ne mogu pronaći nigdje drugo stoga ako se nađemo u situaciji gdje želimo koristiti neke druge alate koji se nisu spominjali u ovom radu pravo mjesto za tražiti pomoć je pomoćni dokument u RobotStudio-u.

5. Zaključak

U radu su je opisan stvoreni automatizirani sustav u čijem centru je industrijski robot IRB 120, dane su korisne informacije o dijelovima koji grade okoliš robota te je objašnjeno kako međusobno komuniciraju. Prikazane su osnovne tehnike potrebne za stvaranje automatiziranog robotskog sustava koji na temelju informacija dobivenih sa vizualnog osjetila može prepoznati i manipulirati zadanim dijelom.

Objasnili smo postupak kojim se povezuje robot s kamerom i kako odrediti pozadinsku logiku koja nam omogućuje pristup nama korisnim informacijama sa fotografije koju uzme kamera. U radu sa ovim sistemom primijetili smo snagu kamere kao alata za rješavanje industrijskih problema u kojima je potrebno stvoriti poveznicu robota sa stvarnim svijetom. Iako je ovo već dugo istraživana i poznata tehnologija njena primjena igra ključnu ulogu u novom industrijskom dobu.

6. Literatura

[1] Product specification – IRB 120, dostupno 9.6.2022.
[2] Product manual – IRB 120, dostupno 9.6.2022.
[3] Product manual – IRC5 Compact, dostupno 9.6.2022.
[4] INTRODUCTION TO MACHINE VISION - A guide to automating process & quality improvements, Cognex, dostupno 9.6.2022.
[5] Application manual – Integrated Vision, dostupno 9.6.2022.
[6] Integrirani dokument pomoći unutar RobotStudio-a Vision kartice
[7] Technical reference manual - RAPID Instructions, Functions and Data types, dostupno 9.6.2022.
[8] In-Sight® 7000 Series Vision System Installation Manual, dostupno 9.6.2022.
[9] Tugomir Šurina, Mladen Crneković: Industrijski roboti, Školska knjiga – Zagreb
[10] Video – dodatni alat, dostupno 9.6.2022.

7. Popis slika

8. Popis tablica

Prilozi

Dodatni primjer programa

```
MODULE Domino_sort
   CONST robtarget cam_tocka:=[[-0.98,2.32,1.08],[0.00950481,-0.999756,0.0159745,-
0.0119591],[-1,-1,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
   CONST robtarget tocka_fix:=[[239.01,-157.52,179.40],[0.00628143,-0.0052333,-0.999967,-
0.000187238],[-1,-1,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];    !Treba se odrediti
wobj0
   TASK PERS tooldata
pgn_spike_prob:=[TRUE,[[0,0,130],[1,0,0,0]],[0.3,[0,0,30],[1,0,0,0],0,0,0]];
   TASK PERS tooldata PGN_Chunk:=[TRUE,[[0,0,87],[1,0,0,0]],[0.3,[0,0,30],[1,0,0,0],0,0,0]];

   PERS robtarget tocka_pick_iznad;    !Tocka iznad tocke dobivene preko kamere
   PERS robtarget tocka_place;         !
   PERS robtarget tocka_place_iznad;
   PERS robtarget tocka_place_par;
   PERS robtarget tocka_place_nepar;

   PERS robtarget tocka_place_iznad_par;
   PERS robtarget tocka_place_iznad_nepar;
   TASK PERS wobjdata wobj_kalib:=[FALSE,TRUE,"",[[390.329,-
43.7501,174.945],[0.999996,-6.08852E-05,-1.31214E-05,-
0.00276972]],[[11.5396,64.5483,0],[0.334396,0,0,-0.942433]]];

   PROC start_prog()
      CamSetProgramMode ABB_Camera1;
      CamLoadJob ABB_Camera1, "domino_test";
      CamSetRunMode ABB_Camera1;
      dom_sort;
   ENDPROC


   PROC dom_sort()


      VAR string br_pomocni_string;   !Varijable za cijeli program
      VAR num br_kruzici{8};
      VAR cameratarget plocica_target{8};
      VAR num unos;
      VAR num time := 1;
      VAR num help := 180;

      VAR num anglex;
      VAR num angley;
      VAR num anglez;

      VAR num br_pronadeno;      !   Varijable za CASE 1 i Varijable za CASE 2
      VAR num br_trazeno;
```

```
    VAR num y :=1;              !Pomocna varijabla za petlju punjenja cameratarget.
    VAR num ploc_index :=1;

    VAR num inc := 0;
    VAR num visina_ploc := 6;
    VAR num pomak_y := 0;
    VAR num pomak_z := 0;

    VAR num pomak_y_par := 0;       !Varijable za CASE 3
    VAR num pomak_z_par := 0;
    VAR num parni_step := 0;

    VAR num pomak_y_nepar := 0;
    VAR num pomak_z_nepar := 0;
    VAR num neparni_step := 0;


    CamReqImage ABB_Camera1;
    CamGetParameter ABB_Camera1 ,"plocice.Number_found"\NumVar:=br_pronadeno;

    WHILE y <= br_pronadeno DO                                  !petlja za spremiti
informacije s kamere u camera_target varijablu
        br_pomocni_string:=NumToStr(y,0);
        CamGetResult ABB_Camera1, plocica_target{y};

        anglex:= EulerZYX(\x,plocica_target{y}.cframe.rot);
        angley:= EulerZYX(\y,plocica_target{y}.cframe.rot);
        anglez:= EulerZYX(\z,plocica_target{y}.cframe.rot);

          IF anglez < help THEN
            anglez := anglez + 180;
          ELSEIF anglez > 180 THEN
            anglez := anglez - 180;
          ELSE
          ENDIF

        plocica_target{y}.cframe.rot := OrientZYX(anglez,angley,anglex);
        CamGetParameter ABB_Camera1,"kruz"+br_pomocni_string+".Blob_Count"
\NumVar:=br_kruzici{y};
        y:=y+1;

    ENDWHILE

    ponovo:
    TPErase;                                    !prikaz izbora radnji na privjesku za ucenje
    TPWrite "1 - Trazenje plocica po broju";
    TPWrite "2 - Slaganje plocica na hrpe";
    TPWrite "3 - Sortiranje plocica na parne i neparne.";
    TPWrite "wobj data:" \Pos:=wobj_kalib.uframe.trans;

    TPReadNum unos, "Izaberite radnju koju zelite izvrsiti: ";      !trazenje unosa od strane
korisnika
```

```
    TEST unos
    CASE 1:                                    !grananje programa ovisno o unosu

    TPErase;
    TPWrite "Izabrali ste broj 1 - Trazenje plocica po broju.";
    WaitTime time;

    TPReadNum br_trazeno, "Izaberite broj koji zelite pronaci: ";
    TPwrite "Izabrali ste broj "\num:=br_trazeno;
    TPWrite "Radnja se izvodi...";
    WaitTime time;

    tocka_place_iznad := offs(tocka_fix,0,0,60);
    tocka_place := offs(tocka_fix,0,0,pomak_z);
    WHILE ploc_index <= br_pronadeno DO                    !petlja za sortiranje po
izabranom broju

      IF br_kruzici{ploc_index} = br_trazeno THEN
        wobj_kalib.oframe := plocica_target{ploc_index}.cframe;
        tocka_pick_iznad := offs(cam_tocka,0,0,20);
        tocka_place := offs(tocka_fix,0,0,pomak_z);
        pomak_z := pomak_z + visina_ploc;

        IF inc = 3 THEN
        pomak_z := 0;
        tocka_place := offs(tocka_fix,0,pomak_y,pomak_z);
        tocka_place_iznad := offs(tocka_place,0,0,60);
        pomak_y := pomak_y + 40;
        inc := 0;
        ENDIF

        MoveL tocka_pick_iznad,v1000, fine, PGN_Chunk\WObj:=wobj_kalib;
        MoveL cam_tocka, v1000, fine, PGN_Chunk\WObj:=wobj_kalib;
        WaitRob \InPos;
        SetDO DO_Ventil,1;
        WaitTime time;
        MoveL tocka_pick_iznad,v1000, fine, PGN_Chunk\WObj:=wobj_kalib;

        MoveL tocka_place_iznad,v1000,fine,PGN_Chunk\WObj:=wobj0;
        MoveL tocka_place,v100,fine,PGN_Chunk\WObj:=wobj0;
        Reset DO_Ventil;
        waittime time;
        MoveL tocka_place_iznad,v1000,fine,PGN_Chunk\WObj:=wobj0;

        ploc_index:=ploc_index+1;
        inc:=inc+1;
      ELSE
        ploc_index:=ploc_index+1;
      ENDIF
    ENDWHILE
```

```
    IF ploc_index > br_pronadeno THEN                    !ako je x inkrementiran za
vise od broja nadenih plocica znaci da ne postoji plocica s trazenim brojem
        TPWrite "Ne postoji plocica s trazenim brojem.";
        WaitTime time;
       ENDIF

    back_home;                                           !Rutina za voznju robota nazad u
poziciju gdje ne smeta kameri.

    CASE 2:
    TPWrite "Izabrali ste broj 2 - Slaganje plocica na hrpe.";
    WaitTime time;
    TPWrite "Radnja se izvodi...";

    tocka_place_iznad := offs(tocka_fix,0,0,60);
    FOR korak FROM 1 TO br_pronadeno DO
       wobj_kalib.oframe:=plocica_target{ploc_index}.cframe;
       tocka_place := offs(tocka_fix,0,pomak_y,pomak_z);
       pomak_z:= pomak_z + visina_ploc;

       IF inc = 3 THEN
          pomak_z := 0;
          pomak_y := pomak_y + 40;
          tocka_place := offs(tocka_fix,0,pomak_y,pomak_z);
          tocka_place_iznad := offs(tocka_place,0,0,60);
          pomak_z := pomak_z + visina_ploc;
          inc := 0;

       ENDIF

          tocka_pick_iznad := offs(cam_tocka,0,0,30);
          MoveL tocka_pick_iznad,v1000, fine, PGN_Chunk\WObj:=wobj_kalib;
          MoveL cam_tocka, v1000, fine, PGN_Chunk\WObj:=wobj_kalib;

          WaitRob \InPos;
          SetDO DO_Ventil,1;
          WaitTime time;
          MoveL tocka_pick_iznad,v1000, fine, PGN_Chunk\WObj:=wobj_kalib;
          MoveL tocka_place_iznad,v1000,fine,PGN_Chunk\WObj:=wobj0;
          MoveL tocka_place,v1000,fine,PGN_Chunk\WObj:=wobj0;
          Reset DO_Ventil;
          waittime time;
          MoveL tocka_place_iznad,v1000,fine,PGN_Chunk\WObj:=wobj0;
          inc:=inc+1;
          ploc_index:=ploc_index+1;

    ENDFOR
    back_home;

    CASE 3:
    TPWrite "Izabrali ste broj 3 - Sortiranja na parne i neparne brojeve.";
    WaitTime time;
```

```
TPWrite "Radnja se izvodi...";


tocka_place_iznad_par := offs(tocka_fix,0,0,60);
tocka_place_iznad_nepar := offs(tocka_fix,180,0,60);

FOR korak FROM 1 TO br_pronadeno DO

  IF br_kruzici{ploc_index} MOD 2 = 0 THEN

    wobj_kalib.oframe := plocica_target{ploc_index}.cframe;
    tocka_place_par := offs(tocka_fix,0,pomak_y_par,pomak_z_par);
    tocka_pick_iznad := offs(cam_tocka,0,0,30);
    pomak_z_par := pomak_z_par + visina_ploc;


    IF parni_step = 3 THEN

      parni_step := 0;
      pomak_z_par := 0;
      pomak_y_par := pomak_y_par + 40;
      tocka_place_par := offs(tocka_fix,0,pomak_y_par,pomak_z_par);
      tocka_place_iznad_par := offs(tocka_place_par,0,0,60);
      pomak_z_par := pomak_z_par + visina_ploc;



    ENDIF

    MoveL tocka_pick_iznad,v1000, fine, PGN_Chunk\WObj:=wobj_kalib;
    MoveL cam_tocka, v1000, fine, PGN_Chunk\WObj:=wobj_kalib;
    WaitRob \InPos;
    SetDO DO_Ventil,1;
    WaitTime time;
    MoveL tocka_pick_iznad,v1000, fine, PGN_Chunk\WObj:=wobj_kalib;

    MoveL tocka_place_iznad_par,v1000,fine,PGN_Chunk\WObj:=wobj0;
    MoveL tocka_place_par,v1000,fine,PGN_Chunk\WObj:=wobj0;
    Reset DO_Ventil;
    waittime time;
    MoveL tocka_place_iznad_par,v1000,fine,PGN_Chunk\WObj:=wobj0;
    parni_step:= parni_step+1;
    ploc_index:=ploc_index+1;

  ELSEIF br_kruzici{ploc_index} MOD 2 > 0 THEN

    wobj_kalib.oframe := plocica_target{ploc_index}.cframe;
    tocka_place_nepar := offs(tocka_fix,180,pomak_y_nepar,pomak_z_nepar);
    tocka_pick_iznad := offs(cam_tocka,0,0,60);
    pomak_z_nepar := pomak_z_nepar + visina_ploc;
```

```
        IF neparni_step = 3 THEN

            neparni_step := 0;
            pomak_z_nepar := 0;
            pomak_y_nepar := pomak_y_nepar + 40;
            tocka_place_nepar := offs(tocka_fix,180,pomak_y_nepar,pomak_z_nepar);
            tocka_place_iznad_nepar := offs(tocka_place_nepar,0,0,60);
            pomak_z_nepar:=pomak_z_nepar + visina_ploc;



        ENDIF

        MoveL tocka_pick_iznad,v1000, fine, PGN_Chunk\WObj:=wobj_kalib;
        MoveL cam_tocka, v1000, fine, PGN_Chunk\WObj:=wobj_kalib;
        WaitRob \InPos;
        SetDO DO_Ventil,1;
        WaitTime time;
        MoveL tocka_pick_iznad,v1000, fine, PGN_Chunk\WObj:=wobj_kalib;

        MoveL tocka_place_iznad_nepar,v1000,fine,PGN_Chunk\WObj:=wobj0;
        MoveL tocka_place_nepar,v1000,fine,PGN_Chunk\WObj:=wobj0;
        Reset DO_Ventil;
        waittime time;
        MoveL tocka_place_iznad_nepar,v1000,fine,PGN_Chunk\WObj:=wobj0;
        neparni_step:= neparni_step+1;
        ploc_index:=ploc_index+1;
      ENDIF
    ENDFOR

    back_home;


    DEFAULT:
    TPWrite "Izabrali ste nepodrzani broj, molimo izaberite ponovo...";
    WaitTime time;
    GOTO ponovo;
    ENDTEST


  ENDPROC
PROC back_home()
CONST robtarget domino_home:=[[76.92,-108.83,409.05],[0.00742434,0.433201,0.901211,-
0.0100245],[-1,-1,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
MoveJ domino_home, v1000, z50, PGN_Chunk;
    WaitRob \InPos;
ENDPROC

  ENDMODULE
```

Sveučilište
Sjever

SVEUČILIŠTE
SJEVER

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, KARLO ŠARIĆ *(ime i prezime)* pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom Programiranje robota IRB120 sa strojnim vidom *(upisati naslov)* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:
(upisati ime i prezime)

(vlastoručni potpis)

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, KARLO ŠARIĆ *(ime i prezime)* neopozivo izjavljujem da sam suglasan/na s javnom objavom završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom Programiranje robota IRB120 sa strojnim vidom *(upisati naslov)* čiji sam autor/ica

Student/ica:
(upisati ime i prezime)

(vlastoručni potpis)