

# Predelava programa AlphaPose za izogibanje trkom v sodelujoči robotiki

Gašper Šavle<sup>1</sup>

<sup>1</sup> Univerza v Ljubljani, Fakulteta za Elektrotehniko  
E-pošta: gaspersavle@yahoo.com

## Abstract

*Collaborative robotics, a growing subfield of industrial robotics, is expected to reach \$7.5 billion by 2027, driven by a shortage of skilled workers, flexible automation needs, and demand for diverse product batches. Unlike standard robots, collaborative robots feature safer designs with torque sensors for collision detection.*

*This project explores machine vision for collision prevention using the Intel Realsense stereoscopic camera and Franka Emika Panda robot. By integrating the AlphaPose program for human pose detection, the system tracks joint positions and slows the robot to a safe speed when a collision risk is detected.*

## 1 Uvod

Sodelujoča robotika, hitro rastoča podzvrst industrijske robotike, naj bi do leta 2027 doseгла vrednost 7,5 milijarde dolarjev in predstavlja 29 % trga. Rast poganjajo pomajkanje usposobljenih delavcev, prilagodljivost avtomatike in manjše, raznolike serije izdelkov.

Sodelujoči roboti se razlikujejo po človeku prijazni, pogosto zaobljeni zasnovi strojne opreme ter senzorjih navora, ki omogočajo zaznavanje sile in preprečujejo trke.

Diplomsko delo predstavi uporabo strojnega vida za preprečevanje trkov z uporabo Intel Realsense kamere, robota Franka Emika Panda in programa AlphaPose. Sistem zaznava položaje človekovih sklepov in upočasni gibanje robota ob tveganju trčenja.

## 2 Pregled ključnih komponent rešitve

V tem poglavju bom predstavil pregled ključnih komponent rešitve, ki so bistvene za delovanje in uspešno implementacijo projekta. Analiziral bom posamezne elemente, njihove funkcije ter medsebojne povezave, ki omogočajo celovito in učinkovito rešitev zastavljenega problema.

### 2.1 Robotski operacijski sistem (ROS)

ROS (Robot Operating System) je odprtakodna programska oprema, ki je zasnovana za hitro implementacijo robotskih aplikacij. Deluje na računalnikih z operacijskim sistemom Linux in je zasnovana na distribuirani arhitekturi. To pomeni, da so lahko ROS aplikacije porazdeljene na večih računalnikih, ki delujejo kot povezana celota. Delo z ROS aplikacijami bistveno olajša uporaba Docker

kontejnerjev, ki poenostavijo upravljanje različnih knjižnic in gonilnikov.

### 2.2 Robotska roka Franka Emika Panda

Robotska roka Franka Emika Panda je sodelujoči robot s 7 prostostnimi stopnjami, razvit kot prvi korak v viziji podjetja, ki predvideva napredovanje od raziskovalnih orodij do robotskih pomočnikov v vsakdanjem življenju. Leta 2017 so predstavili sistem "Panda power tool" enostavno programiranje nalog brez programerskih izkušenj. Robot omogoča programiranje z demonstracijo, kjer ponavlja uporabnikove gibe. Paket "Panda research package" omogoča dvosmerno komunikacijo z robotom v realnem času, uporabo namenskih krmilnikov in integracijo v okolje ROS prek vmesnika franka\_ROS. Sam sem uporabljal ta način krmiljenja, saj sem za dinamično prilaganje gibanja potreboval krmilnik z dinamičnimi generatorji gibov (DMP).

### 2.3 Dinamični generatorji gibov (DMP)

Dinamični generatorji gibov (angl. "Dynamic Movement Primitives" - DMP) omogočajo zapis in prilaganje kompleksnih robotskih gibanj, ki so ključna za naloge, kot so premiki rok ali hoja. Glavni cilj DMP-jev je omogočiti reprodukcijo teh gibanj v dinamičnih in nestrukturiranih okoljih.

DMP-ji temeljijo na dveh komponentah: nelinearnih bazičnih funkcijah (običajno Gaussovih), ki določajo obliko gibanja, in linearnih diferencialnih enačbah, ki zagotavljajo konvergenco proti cilju.

Njihova glavna prednost je sposobnost prilaganja trajektorij glede na spremembe v okolju, kar omogoča zanesljivo izvajanje nalog tudi v nepredvidljivih pogojih.

DMP je sistem, zgrajen na osnovi kritično dušene diferencialne enačbe

$$\tau \dot{z} = \alpha_z(\beta_z(g - y) - z), \quad (1)$$

$$\tau \dot{y} = z. \quad (2)$$

V zgornjih enačbah je  $y$  ena od prostostnih stopenj robota, pomožna spremenljivka  $z$  je skalirana hitrost prostostne stopnje  $y$ ,  $g$  je želena ciljna točka, v katero želimo, da se premakne dotična prostostna stopnja,  $\alpha_z, \beta_z > 0$  pa sta ojačanji.  $\tau$  v zgornji enačbi predstavlja časovni skalirni faktor, s katerim lahko upočasnimo ali pospešimo

gibanje robota. Zgornji enačbi opisujeta dinamični sistem mase, vzmeti in blažilca, kjer vzmet trenutno stanje sistema  $y$  "vleče" proti ciljni točki  $g$ . Ko sistem pride do končne točke, postaneta  $z$  in  $y$  enaka.

$$(z, y) = (0, g) \quad (3)$$

in se ne spreminja več.

Diferencialni enačbi (1) in (2) zadostujeta za definicijo enostavnih trajektorij od poljubnega začetnega stanja  $y_0$  do ciljne točke  $g$ . Za definicijo bolj zapletenih trajektorij pa moramo uvesti dodatno nelinearno funkcijo  $f$  (angl. forcing term), ki omogoči oblikovanje želene trajektorije

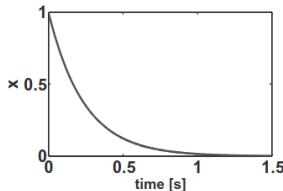
$$\tau \dot{z} = \alpha_z (\beta_z (g - z) - z) + f(x), \quad (4)$$

$$\tau \dot{y} = z. \quad (5)$$

$f$  je nelinearna funkcija, ki je odvisna od faze  $x$ . S tem se izognemo direktni vpeljavi časa v enačbi (4) in (5). Faza  $x$  je definirana z enačbo

$$\tau \dot{x} = -\alpha_x x. \quad (6)$$

To dodatno enačbo imenujemo *kanonični dinamični sistem*.



Slika 1: Graf spremembe fazne spremenljivke po času, kot je opisano v enačbi (6)[1].

Nelinearna funkcija  $f$  je definirana kot utežena superpozicija Gaussovih funkcij  $\Psi_i$  v odvisnosti od fazne spremenljivke  $x$

$$f(x) = \frac{\sum_{i=1}^N w_i \Psi_i(x)}{\sum_{i=1}^N \Psi_i(x)} x(g - y_0) \quad (7)$$

$$\Psi_i(x) = \exp(-h_i(x - c_i)^2). \quad (8)$$

$w_i$  so uteži Gaussovih funkcij, ki so centrirane okoli točk  $c_i$  z varianco  $h_i$ .  $f$  je torej sestavljena iz nabora Gaussovih funkcij, ki so aktivirane zaporedno, ko fazni potek giba doseže posamične centre  $c_i$ . Končno množenje z  $x(g - y_0)$  skrbi za skaliranje funkcije  $f$  glede na potek faze in amplitudo giba, ki jo definira razlika med ciljem  $g$  in začetno lego prostostne stopnje  $y_0$ . To skaliranje je pomembno, saj pri normalnem delovanju sistema želimo zagotoviti, da je vrednost nelinearne funkcije na  $f$  na koncu giba enaka 0. Za to poskrbi množenje s fazno spremenljivko  $x$ , ki od začetne vrednosti 1 konvergira proti 0.

DMP omogoča prilaganje naučenih trajektorij novim robnim pogojem. To omogoča uporabo enega posnetega giba za izvedbo več podobnih nalog z različnimi cilji, začetnimi pogoji ali trajanjem gibanja.

## 2.4 Program za zaznavanje položaja človeka – AlphaPose

AlphaPose je sistem za natančno sledenje položaju celotnega človeškega telesa v realnem času [2]. Na področju detekcije položaja človeškega telesa je ta sistem uvedel dve novi tehniki.



Slika 2: Obdelava slike s sistemom AlphaPose

### 2.4.1 Simetrična integralna regresija (Symmetric Integral Keypoint Regression)

Integralna regresija je metoda predvidevanja izhoda nevronske mreže, ki so jo avtorji članka [2] implementirali kot zamenjavo za metodo "Heatmap" (glej Sliko 3), zaradi njenih dveh najbolj perečih problemov.



Slika 3: "Heatmap" predstavitev verjetnosti detekcije smrčka [2].

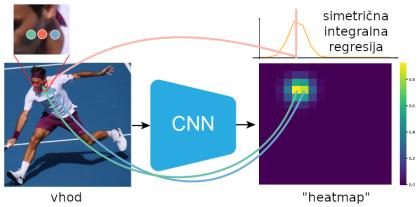
- Ne odvedljivost izhoda:** "Heatmap" način zapisa verjetnostne porazdelitve razredov na slikah ni zvezzen in posledično ni odvedljiv. "Heatmap" namreč predstavlja diskretno verjetnostno porazdelitev, saj konvolucijske nevronske mreže (CNN) podajo izhod v obliki dvodimenzionalne matrike verjetnosti [2].
- Kvantizacijska napaka:** je pogojevana z načinom delovanja CNN in je neizogibna pri diskretizaciji. Kvantizacijska napaka povzroči izgubo informacij, ki se lahko manifestira kot nenatančno zaznani robovi objektov [2].

Simetrična integralna regresija je metoda za razpoznavanje lokacij sklepov na podlagi Heatmap predstavitev, ki jo generira konvolucijska nevronska mreža (CNN). V tem postopku izračunamo utežen integral vseh lokacij v Heatmap matriki verjetnosti. Metoda združuje prednosti direktnega pristopa s Heatmap matriko in integralne

regresije, hkrati pa se izogne njunim slabostim, kot sta računska zahtevnost integralne regresije in neodvedljivost Heatmap predstavitev [2].

Simetrična integralna regresija temelji na običajni integralni regresiji, pri čemer upošteva prostorsko simetrijo ključnih točk človeškega telesa. Kot je omenjeno v [2], je metoda enako hitra in natančna kot pristopi, ki uporabljajo Heatmap reprezentacijo, vendar odpravi kvantizacijsko napako teh metod.

Metoda se je izkazala kot učinkovit in zanesljiv pristop za določanje položaja sklepov človeškega telesa. Posebej je uporabna pri obdelavi velikih naborov podatkov in v aplikacijah, kjer je pomembno delovanje v realnem času.



Slika 4: Vizualizacija kvantizacijske napake pristopa s "Heatmap" reprezentacijo in primerjava s simetrično integralsko regresijo[2].

#### 2.4.2 Parametrično dušenje ne-maksimalnih vrednosti (Parametric Pose Non-Maximum Suppression - PP-NMS)

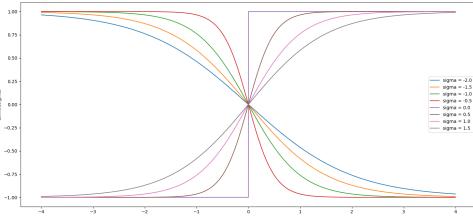
PP-NMS je algoritem za filtriranje nepravilno zaznanih položajev človeškega telesa. Algoritmi za zaznavanje položajev pogosto izračunajo več različic položaja iste osebe, kar je posledica delno zakritega telesa ali šuma v sliki. Pred uvedbo sistema AlphaPose so avtorji teh algoritmov poskušali rešiti ta problem z dvigom praga zaupanja, potrebnega za pozitiven izhod zaznavanja. Ta pristop pa ni bil dovolj robusten in je pogosto povzročil, da so nekateri položaji ostali nezaznani, kar je še posebej nezaželeno pri razpoznavanju večjega števila ljudi.

Sistem AlphaPose je ta problem rešil z znižanjem minimalne stopnje zaupanja, kar pa poveča tveganje za nepravilna zaznavanja položajev. Za odstranjevanje teh nepravilnih zaznavanj algoritem PP-NMS izbere položaj z najvišjo stopnjo zaupanja kot referenco. Položaji, ki so blizu temu referenčnemu položaju, postanejo kandidati za odstranitev. Odločitev o odstranitvi temelji na dveh *eliminacijskih kriterijih*, ki ocenjujeta ključne dejavnike.

**1. Podobnost položajev:** pri primerjanju podobnosti dveh zaznanih položajev primerjamo zaupanja detekcij posameznih sklepov znotraj okvirja  $B_i$ , ki zajema celoten zaznan položaj  $P_i$ . Znotraj danega okvirja je definirana funkcija ujemanja med položajema.

$$K_{sim}(P_i, P_j | \sigma_1) = \begin{cases} \sum_n \tanh \frac{c_i^n}{\sigma_1} \cdot \tanh \frac{c_j^n}{\sigma_1}, & \text{if } k_j^n \text{ is within } B(k_i^n) \\ 0, & \text{otherwise} \end{cases}, \quad (9)$$

kjer je  $k_i^n$  lokacija  $n$ -tega sklepa  $i$ -tega položaja, torej sredina okvirja  $B_i$ ,  $c_i^n$  pa predstavlja zaupanje v detekciji  $n$ -tega sklepa  $i$ -tega položaja. Zaupanja detekcij so z uporabo funkcije  $\tanh$  normalizirana na vrednosti med -1 in 1 in pri tem vrednosti z nizkim zaupanjem potisne proti 0, vloga konstante  $\sigma_1$  je, da narekuje strmino funkcije in s tem uteži zaupanja detekcij sklepov. Ko imata  $n$ -ta sklepa visoko stopnjo zaupanja, bo rezultat funkcije za ta sklep blizu 1. Z uporabo te funkcije "mehko" preštejemo sklepe, ki se ujemajo med zaznanimi položajema.



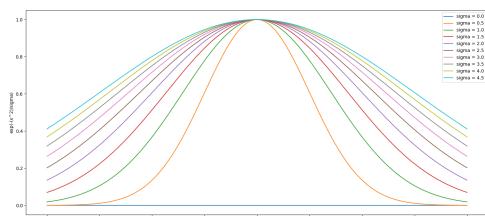
Slika 5: Prikaz spremembe strmine funkcije  $\tanh(x)$  glede na vrednost  $\sigma_1$

#### 2.4.3 Razdalja položajev

Poleg podobnosti položajev je za izključitev ne-unikatnih položajev potrebna tudi metrika razdalje med primerjanimi zaznavama. Če se položaja prekrivata, obstaja večja možnost, da je ena izmed primerjanih zaznav lažna. Formula za oceno razdalje med zaznanimi položajema se glasi:

$$H_{sim}(P_i, P_j | \sigma_2) = \sum_n \exp\left[-\frac{(k_i^n - k_j^n)^2}{\sigma_2}\right], \quad (10)$$

kjer sta  $k_i^n$  in  $k_j^n$   $n$ -ta sklepa posameznega primerjavnega položaja,  $\sigma_2$  pa parameter Gaussovega uteževanja, ki določa, kolikšen vpliv na končni rezultat bodo imele velike in majhne razdalje med sklepi. Večja vrednost parametersa  $\sigma_2$  pomeni večji vpliv velikih razdalj med položaji sklepov, manjša vrednost pa pomeni manjšo vplivnost položajev z velikimi medsebojnimi razdaljami.



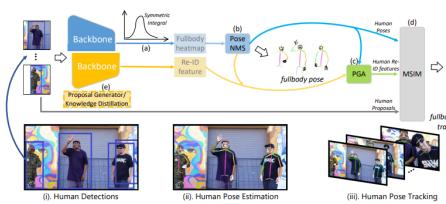
Slika 6: Prikaz spremembe vpliva oddaljenosti pri različnih vrednostih faktorja  $\sigma_2$

S kombinacijo zgornjih enačb (9) in (10) lahko postavimo končni eliminacijski kriterij, ki vključuje prispevke obeh faktorjev.

$$d(P_i, P_j | \Lambda) = K_{sim}(P_i, P_j | \sigma_1) + \lambda H_{sim}(P_i, P_j | \sigma_2), \quad (11)$$

kjer je  $\Lambda$  nabor parametrov  $\{\sigma_1, \sigma_2, \lambda\}$ ,  $\lambda$  pa faktor, ki uteži vpliv posamezne metrike za eliminacijo položajev. Izločeni so položaji, pri katerih je vrednost eliminacijskega kriterija nižja od mejne vrednosti  $\eta$

$$f(P_i, P_j | \Lambda, \eta) = \begin{cases} 1, & d(P_i, P_j | \Lambda) \leq \eta \\ 0, & \text{otherwise} \end{cases}. \quad (12)$$



Slika 7: Arhitektura sistema AlphaPose[2].

### 3 Predelava programa AlphaPose

Program AlphaPose ima dva načina delovanja, deluje lahko v realnem času ali pa obdeluje že posnete slike oziroma videoposnetke. Za načrtovano področje uporabe smo potrebovali delovanje v realnem času. Pri tem se je pojavil problem, da sistem tako delovanje omogoča le z uporabo spletnne kamere, ki je prek USB vodila povezano direktno na računalnik, na katerem teče program AlphaPose. Zaradi tovrstnega delovanja nastaneta dva različna problema.

- **Docker** Ker je moja implementacija sistema AlphaPose morala delovati v Docker okolju, bi bilo potrebno le-temu omogočiti dostop do USB vhodov in izhodov, kar predstavlja varnostno tveganje, saj ima Docker okolje administratorske privilegije.
- **ROS** Sistem AlphaPose smo vgradili v obstoječi projekt, kjer je bil velik poudarek na samostojnosti posameznega dela rešitve in enostavni integraciji v ROS (glej razdelek 2.1). Pri tem moramo imeti možnost izbire kamer, kar najenostavnije dosežemo z zajemanjem slik v samostojnem ROS vozlišču.

AlphaPose izračuna le lokacije posameznih človeških sklepov na dvodimenzionalni sliki. Za celovito zaznavanje človeškega položaja v robotski celici je potrebnih več podatkov, kot so tridimenzionalna informacija o lokaciji sklepov in koti človeških sklepov za modeliranje človeškega položaja. V tem delu predstavljam prilagoditev originalnega programa AlphaPose v skladu z zgoraj omenjenimi zahtevami in pomanjkljivostmi originalnega sistema.

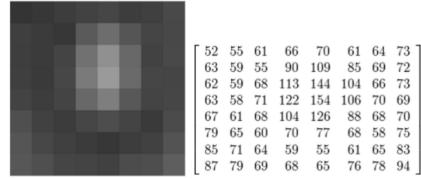
### 3.1 Integracija s knjižnico ROSPy

Za implementacijo rešitve sem uporabil okolje ROS in programski jezik Python. Komunikacijo z ROS sem omogočil z uvozom knjižnice ROSPy, ki omogoča upravljanje s temami, storitvami in parametri (glej razdelek 2.1).

Razvil sem funkcijo za vzpostavitev povezave z vozlišči mojega ROS programa ter definiral odjemalce in ponudnike. V njej sem nastavil storitve za zagon in ustavljanje programa AlphaPose ter za kalibracijo in izbiro kamere, kar omogoča spreminjanje parametrov brez ponovnega zagona.

To je pomembno za YOLOv3 nevronsko mrežo, ki jo AlphaPose uporablja za zaznavanje človeškega položaja, saj lahko ostane naložena v grafični kartici. Prav tako lahko začasno prekinem pretok informacij in sprostim pa sovno širino ROS za časovno kritične naloge.

Prva pomanjkljivost AlphaPose, ki jo je bilo potrebno odpraviti, je bilo zajemanje slik z ROS vozliščem kamere. V ta namen sem zaobšel DataLoader in sliko z orodjem CvBridge().imgmsg\_to\_cv2 pretvoril iz slikovnega sporočila v ROS formatu (sensor\_msgs.Image) in jo pretvoril v format, ki ga uporablja knjižnica OpenCV (glej Sliko 8).



Slika 8: Format OpenCV slike (np.ndarray)

ROSPy za prejemanje datotek z ROS tem uporablja tako imenovane povratne (callback) funkcije. Te funkcije izvedejo določeno operacijo ob nastanku dogodka, v mojem primeru ob pojavu nove slike na slikovni temi, ki jo spremjam. Uporaba teh funkcij omogoča asinhrono delovanje programa. Ker AlphaPose ni zasnovan za tovrstno delovanje, sem njegovo delovanje razdelil na inicjalizacijski del, ki ob prvem zagonu naloži nevronsko mrežo in njene uteži, ter na obdelovalni del, kjer poteka zaznavanje človeškega položaja na prejeti sliki.

Slike sem zajemal s stereoskopsko kamero Intel RealSense, ki jo ROS podpira. ROS objavlja barvno in globinsko sliko na dveh ločenih temah, zato sem za njuno obdelavo uporabil dve povratni funkciji. Barvno sliko sem posredoval funkciji za zaznavanje človeškega položaja in jo vključil v AlphaPose, iz globinske slike pa sem pridobil podatke za izračun položajev sklepov in njihove oddaljenosti od robota.

#### 3.1.1 Barvna slika

Za pridobivanje barvne slike je zadolžena povratna funkcija pose\_CB. Ta funkcija sprejeto sliko najprej pretvori iz ROS slikovnega sporočila v format openCV, nato jo posreduje sistemu AlphaPose, ki nam vrne dvodimenzionalne lokacije sklepov na sliki. Ti podatki so bili uporabljeni tudi za animacijo zaznanih človeških sklepov v robotske vizualizacijskem orodju RViz.

### 3.1.2 Globinska slika

Za pridobitev globinska slike je zadolžena povratna funkcija `depth_CB`, ki prav tako kot prejšnja povratna funkcija najprej pretvori sliko iz formata slikovnega sporočila v `openCV` format. V globinskih slikah so globine posameznih točk predstavljene s sivinskimi nivoji, kjer je sivinski nivo točke enak njegovi globini v milimetrih. Točka, ki je od kamere oddaljena 5 m, ima torej sivinski nivo 5000. Na 16-bitni lestvici je 0 popolnoma črno, 65536 pa popolnoma belo. Vendar stereoskopske globinske kamere ustvarjajo artefakte v obrisih, kar se prikaže kot popolnoma črna obroba predmetov na sliki. Za minimizacijo vpliva teh artefaktov sem globinsko sliko pred uporabo zameglil z uporabo `openCV` metode `cv2.blur()`. Glavna naloga te povratne funkcije je razbiranje globinskih vrednosti posameznih sklepov. Funkcija kot vhod prejme podatke s položaji posameznih sklepov in nato za vsak sklep prebere sivinsko vrednost točke s temo koordinatama in jo doda v slovar `body_dict`, namenjen shranjevanju vseh potrebnih informacij o položaju uporabnika (lege sklepov, koti sklepov, drevesna struktura).

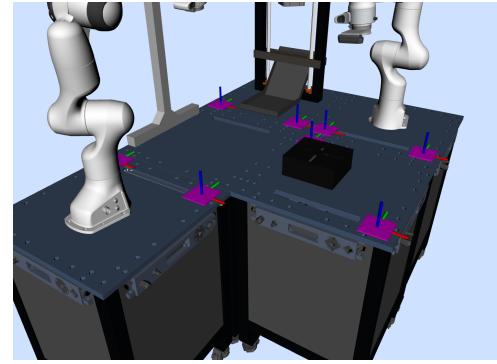
## 4 Določitev človeškega položaja

Prednost strojnega vida v primerjavi z drugimi sistemmi za merjenje položaja človeka je hitra in neobčutljiva postavitev sistema. Pri uporabi svetlobne zaves moramo, na primer, biti pazljivi pri orientaciji modulov svetlobne zaves, saj moramo paziti na poravnano in zagotavljanje jasnega vidnega polja obema deloma svetlobne zaves. Oba dela morata biti postavljena na specifični razdalji, ki je ne moremo spremnijati. To je problematično za industrijske aplikacije, ki stremijo k vedno bolj dinamičnemu pristopu do robotskeih celic. Za razliko od sistemov s svetlobnimi zavesami ali detektorji človeške prisotnosti, lahko z uporabo strojnegavida določamo človeško oddaljenost od trenutnega položaja robota iz različnih zornih kotov, iz katerih sta razvidna tako človek kot tudi celica. Kljub temu pa moramo paziti, da je kamera postavljena na fiksen položaj in da je transformacija med koordinatnim sistemom kamere in robota znana.

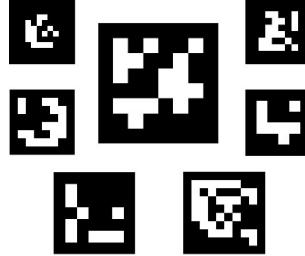
### 4.1 Določitev položaja kamere

Za natančno računanje razdalje med človekom in robotom je nujno potreben natančen podatek o položaju kamere. V mojem sistemu sem uporabil dve kameri, od katerih je ena bila nameščena na koncu robotske roke, druga kamera pa pod stropom. Določanje položaja prve kamere v robotskega koordinatnega sistemu je bilo enostavno, saj je bil položaj določen s transformacijsko matriko od zadnjega člena robotske roke. Položaj druge kamere sem izračunal s uporabili ArUco markerjev (glej Sliko 10). ArUco markerji spadajo med referenčne markerje s točno določenimi dimenzijami, na podlagi katerih lahko določimo velikost, oddaljenost in orientacijo markerjev.

Na delovno površino v robotske celici sem na lokacije, ki so vidne tudi med delovanjem robota, namestil nabor ArUco markerjev (glej Sliko 9), ki sem jih izdelal s 3D tiskalnikom. Umestil sem jih v prav tako natisnjene



Slika 9: Prikaz markerjev v vizualizacijskem oknu RViz



Slika 10: Primer ArUco markerjev

nosilce, ki so se popolnoma prilegali luknjam, ki so bile na 0,01 mm natančno izvrte v robotsko mizo. S tem sem zagotovil natančno pozicioniranje markerjev. V predelani program AlphaPose sem vnesel njihove položaje glede na bazni koordinatni sistem robotske celice. Nato sem z uporabo knjižnice `openCV` ArUco izračunal njihove lokacije v dvodimenzionalnih slikah. Iz znanih razdalj med markerji v koordinatnem sistemu robotske celice sem nato z uporabo  $n$ -točkovne perspektive določil položaja kamere v koordinatnem sistemu robotske celice in koordinatnem sistemu robota. Pri tem sem uporabil transformacijo

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}. \quad (13)$$

Funkcija (13) opisuje transformacijo točke iz koordinatnega sistema robotske celice v koordinatni sistem kamere, kjer so  $X_w$ ,  $Y_w$  in  $Z_w$  koordinate točke v koordinatnem sistemu robotske celice,  $f_x$  in  $f_y$  goriščni razdalji kamere,  $c_x$  in  $c_y$  pa optična centra kamere.  $U$  in  $V$  predstavljata projekcijo točke na 2D sliko,  $r_{11} \dots r_{33}$  rotacijsko matriko transformacije,  $t_x$ ,  $t_y$  in  $t_z$  pa komponente translacijskega vektorja od točke v koordinatnem sistemu sveta do koordinatnega sistema kamere.

Za določitev položaja kamere je potreben inverz zgornje funkcije, ki pa lahko vrne dvoumno rešitev, če vsebuje le minimalno število točk (3). Ta pojav, znan kot dvoumnost 3-točkovne perspektive, sem rešil z uporabo 6 markerjev, kar je odpravilo problem. Zaradi ločljivosti ka-

mere in večje oddaljenosti pa so rezultati bili nekoliko zamenjani. Te zamike sem odpravil s funkcijo `CamPoseCorr`. Funkcija najprej določi središča zaznanih markerjev in z globinsko kamero izmeri njihovo razdaljo. Nato transformira koordinate markerjev iz koordinatnega sistema kamere v koordinatni sistem robotske celice in izračuna napako med transformiranimi in dejanskimi lokacijami markerjev. Na podlagi teh napak funkcija določi povprečno deviacijo in prilagodi položaj kamere, kar je razvidno iz njenega izhoda.

Ko je položaj kamere natančno določen, program prične sporočati lokacije posameznih sklepov, ki jih na sliki določi orodje AlphaPose (glej Razdelek 2.4). Globinske vrednosti sklepov pridobim iz globinske slike.

$$T_{\text{camera to global}} = \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} & x \\ r_{yx} & r_{yy} & r_{yz} & y \\ r_{zx} & r_{zy} & r_{zz} & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (14)$$

Transformacijsko matriko od kamere do globalnega koordinatnega sistema pridobimo z uporabo n-točkovne perspektive, kot je opisano v razdelku (4.1). Lokacijo človeškega sklepa v globalnem sistemu torej pridobimo z matričnim produktom med transformacijsko matriko iz globalnega koordinatnega sistema v koordinatni sistem kamere in vektorjem položaja dotičnega sklepa v koordinatnem sistemu kamere.

$$\mathbf{P}_{\text{global}} = \mathbf{T}_{\text{global to camera}} \cdot \mathbf{P}_{\text{camera}} \quad (15)$$

## 4.2 Izračun razdalje med človekom in robotom

Izogibanje trkom sem implementiral tako, da se robot zvezno upočasni v skladu z oddaljenostjo do človeka. To razdaljo izračunamo s pomočjo evklidske razdalje med vsemi sklepi in vrhom robota ter glede na najmanjšo od njih prilagodimo hitrostni parameter DMP sistema. Posamične razdalje torej izračunamo po sledeči formuli:

$$d_i = \|\mathbf{r} - \mathbf{h}_i\|. \quad (16)$$

$d_i$  so tu razdalje med vrhom robota  $\mathbf{r}$  in i-tim sklepolom človeškega telesa  $\mathbf{h}_i$ . Od tod izračunamo minimalno razdaljo

$$d_{\min} = \min_i \{d_1, d_2, \dots, d_n\}. \quad (17)$$

Razdaljo, kjer robot prične upočasnjevati, določimo v skladu z ISO standardom za strojno varnost, **ISO 13855** [3], [4], ki narekuje, na kolikšni razdalji od naprave, ki potencialno ogroža človeka, morajo biti postavljeni varnostni ukrepi glede na hitrosti približevanja. Varnostno razdaljo izračunamo po sledeči formuli

$$S = K \cdot T + 8 \cdot (d - 14), \quad (18)$$

kjer  $S$  predstavlja izračunano varnostno razdaljo [mm],  $K$  hitrost približevanja [mm/s],  $T$  reakcijski čas sistema [s] in  $d$  ločljivost sistema za merjenje razdalje [mm] [5].

Povprečna hitrost približevanja v testni trajektoriji, s statičnim subjektom je bila 350 mm/s, ločljivost sistema za zaznavanje razdalje je bila v rangu 5 mm, odzivni reakcijski čas sistema pa je bil 3 s. Formula za izračun varnostne razdalje se je torej glasila:

$$S = 350 \cdot 3 + 8 \cdot (5 - 14) \quad (19)$$

$$S = 978 \text{ mm}. \quad (20)$$

## 4.3 Prilaganje hitrosti gibanja robota z DMP-ji

Glede na izračunano varnostno razdaljo postavimo mejo, znotraj katere DMP sistemu spremojamo hitrostni skalirni faktor  $\tau$ . Njegovo vrednost spremojamo na podlagi razdalje med človekom in robotom, izračunane po postopku, opisanem v prejšnjem razdelku (4.2). Hitrostni skalirni faktor spremojamo samo, ko je izmerjena razdalja med človekom in vrhom robota manjša ali enaka varnostni razdalji po enačbi (18).

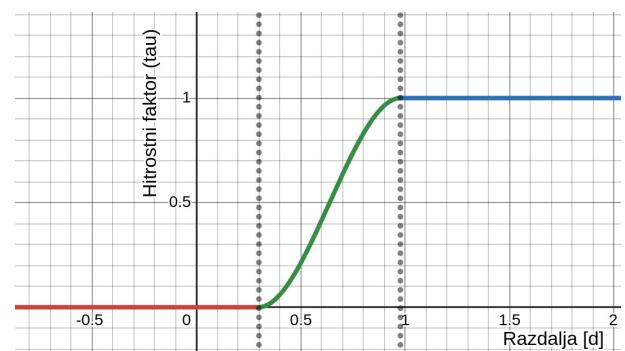
Naj bo  $\tau_0$  časovni skalirni faktor za dani DMP. Definirajmo naslednjo funkcijo razdalje, razvidno na grafu (11).

$$f(d) = \begin{cases} 0 & \text{if } d \leq a \\ 3 \left( \frac{d-a}{b-a} \right)^2 - 2 \left( \frac{d-a}{b-a} \right)^3 & \text{if } a < d < b \\ 1 & \text{if } d \geq b \end{cases} \quad (21)$$

Ta funkcija ima vrednost 0 pri  $a$  in 1 pri  $b$ . Če torej definiramo  $a = 0.3$  in  $b = 0.978$  in izračunamo  $\tau$  po naslednji formuli

$$\tau(d) = \tau_0 f(d), \quad (22)$$

bo  $\tau$  enak 0 na razdalji 0.3 m ali manj, kar pomeni, da se bo gibanje robota pri takšni razdalji povsem ustavilo. Pri razdalji 0.978 m ali več pa postane  $\tau$  enak začetnemu skalirnemu faktorju  $\tau_0$ , kar pomeni, da se bo robot spet gibal z načrtovano hitrostjo. Pri vmesnih razdaljah je gibanje robota upočasnjeno. Na ta način lahko zagotovimo varno in zvezno gibanje robota.



Slika 11: Graf spremembe hitrostnega skalirnega faktorja  $\tau$  glede na razdaljo, po formuli (21).

## 5 Rezultati

V tem poglavju so predstavljeni rezultati delovanja predstavljene rešitve za zagotavljanje varnosti s pomočjo

računalniškega vida. Primarni cilj je ocena zanesljivosti in učinkovitosti odziva sistema na potencialne trke med operaterjem in robotom v realnem času. Rezultati so organizirani glede na hitrost odziva sistema na prisotnost človeka v delovnem prostoru odzivni čas med vstopom človeka v območje ogroženosti in reakcijo sistema (bodisi upočasnitev ali popolna zaustavitev gibanja robota). Zanesljivost delovanja sistema je bila testirana pri različnih osvetlitvah okolja, več zaznanih osebah v vidnem polju kamere, enostavnih ali kompleksnih ozadjijih in pri različnih hitrostih in poteh premikanja operaterja.

	OpenPose (CMU-Pose)	Detectron (Mask R-CNN)	AlphaPose
AP @0.5-0.95	61.8	67.0	73.3
AP @0.5	84.9	88.0	89.2
AP @0.75	67.5	73.1	79.1
AP medium	57.1	62.2	69.0
AP large	68.2	75.6	78.6

Tabela 1: Evalvacija delovanja modela AlphaPose na testnem naboru podatkov COCO [6].

	OpenPose (CMU-Pose)	Newell & Deng	AlphaPose
Glava	91.2	92.1	91.3
Rama	87.6	89.3	90.5
Komolec	77.7	78.9	84.0
Zapestje	66.8	69.8	76.4
Kolk	75.4	76.2	80.3
Koleno	68.9	71.6	79.9
Gleženj	61.7	64.7	72.4
Povprečno	75.6	77.5	82.1

Tabela 2: Evalvacija delovanja modela AlphaPose na testnem naboru podatkov Max Planck Institut Informatik (MPII) Human pose [7].

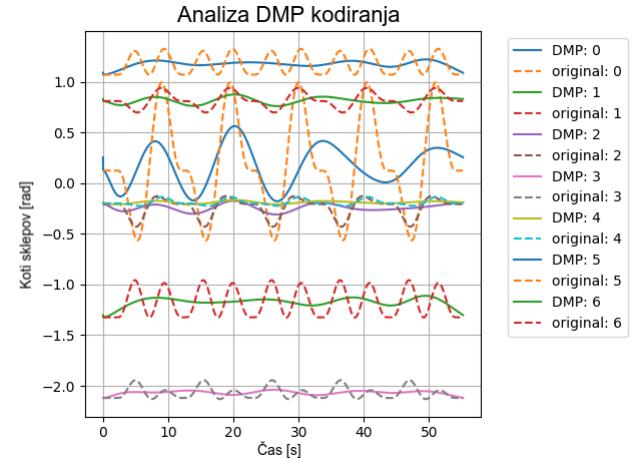
### 5.1 Opis primerov in predstavitev rezultatov

Meritve so bile izvedene na način, ki nazorno izpostavlja lastnosti sistema, ki so ključne za uspešno in učinkovito delovanje. Prav tako sem želel izpostaviti lastnosti, ki omogočajo nadomestiti tradicionalne rešitve z implementiranim sistemom. Za uspešno delovanje sistema sta predvsem pomembna odzivni čas in natančnost računalniškegavida. Za konkurenčnost varnostnega sistema so pomembni tudi njegova prilagodljivost, robustnost in adaptivnost. Meritve so bile opravljene s testno robotsko trajektorijo, ki je bila v vseh poskusih enaka, v eksperimentih pa so bili merjeni položaj uporabnika, položaji robotskih sklepov, razdalja med uporabnikom in vrhom robota in hitrostni faktor  $\tau$ . Med posameznimi izvedbami eksperimenta so bili spremenljivi faktorji pristop uporabnika, položaj kamere, osvetlitev in število oseb v vidnem polju kamere.

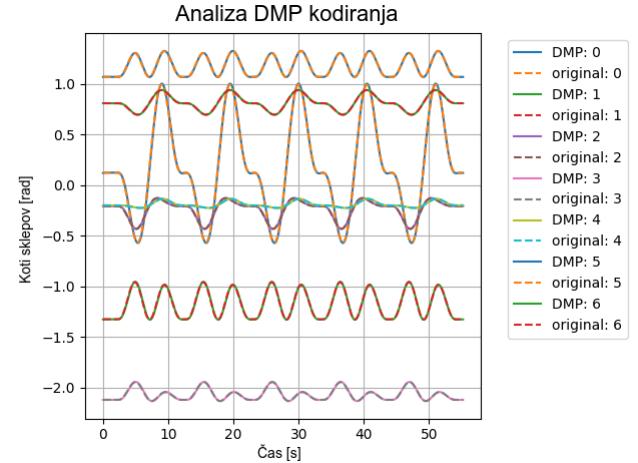
### 5.2 Evalvacija delovanja DMP-jev

Ta razdelek obravnava evalvacijo delovanja DMP-jev. Pri oceni delovanja je poudarek na pregledu pozicijskega in hitrostnega pogreška pri izvajanjtu trajektorije. Evalvacija teh napak, razvidnih na sliki (13) je ključen faktor pri

razumevanju natančnosti reprodukcije gibanja z DMP-ji. Meritve so bile opravljene s testno trajektorijo, prirejeno za prikaz delovanja najpomembnejših metrik sistema. Trajektorija je bila zasnovana tako, da se premikajo vsi zglobi robota, prav tako pa je morala trajektorija nihat, da je bil razviden odziv na spremjanje razdalje v primeru, ko se operater ne premika.



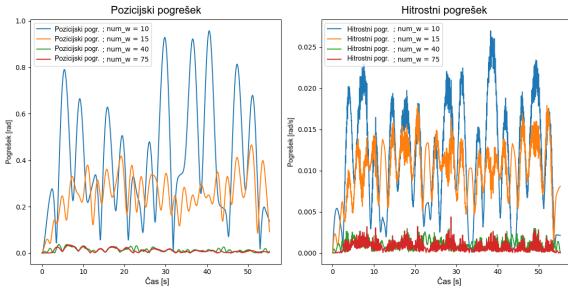
Slika 12: Primerjava izvorne trajektorije in njene DMP reprodukcije: Izvedba z 10 Gaussovimi funkcijami



Slika 13: Primerjava izvorne trajektorije in njene DMP reprodukcije: Izvedba s 75 Gaussovimi funkcijami

Na slikah 14 so predstavljene normirane vrednosti pozicijskega in hitrostnega pogreška vseh sklepov med nominalno trajektorijo in reprodukcijo le-te z uporabo DMP-jev pri različnih številih uteži. Napake smo izračunali z enačbama (5.2)

$$\begin{aligned}
 e_q &= \sqrt{\sum_{n=0}^6 (q_n(\text{orig.}) - q_n(\text{dmp}))^2}, \\
 e_{\dot{q}} &= \sqrt{\sum_{n=0}^6 (\dot{q}_n(\text{orig.}) - \dot{q}_n(\text{dmp}))^2}, \\
 \dot{q}_n &= \frac{q_n(i) - q_n(i-1)}{t(i) - t(i-1)}
 \end{aligned} \tag{23}$$



Slika 14: Primerjava normiranih pozicijskih  $e_q$  in hitrostnih  $e_{\dot{q}}$  napak DMP reprodukcije trajektorije pri različnih številih vzbujalnih funkcij

### 5.3 Evalvacija odziva sistema

V tem razdelku je opisan odziv, ki je bil merjen pri dveh različnih vstopih operaterja v robotsko celico. Pri prvem vstopu operater vstopi v vidno polje kamere s strani, pri drugem pa se pojavi v sredini vidnega polja, ko vstopi skozi vrata. Cilj meritve je prikazati sposobnost delovanja sistema pri različnih osvetlitvah in zornih kotih.

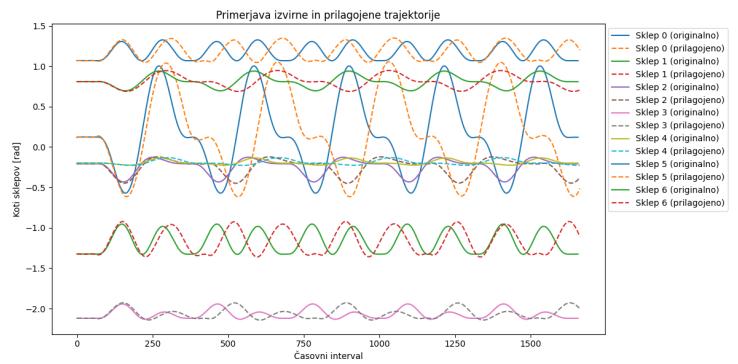
Graf 15 prikazuje razliko v gibanju robotskih sklepov pri delovanju brez prisotnosti operaterja v delovnem prostoru robota (polne črte) in pri delovanju, kjer se je operater nahajal znotraj robotskega delavnega prostora (črtkane črte). Iz grafa je razvidno, da je gibanje ob prisotnosti operaterja upočasnjeno in da se upočasnitev spreminja z oddaljenostjo od operaterja. V tem eksperimentu je namreč robot izvajal gib, s katerim se je periodično približeval in oddaljeval od statičnega operaterja.

Grafa na sliki 16 prikazujeta delovanje varnostnega sistema pri idealnih svetlobnih pogojih z ambientalno svetlostjo 1200 lux. Ob vstopu operaterja v sredino vidnega polja kamere je opazen šum, ko pride do prekrivanj in program izgubi sled za operaterjem. Kljub temu se robot pri vsakem približanju pod dovoljeno mejo pravočasno ustavi.

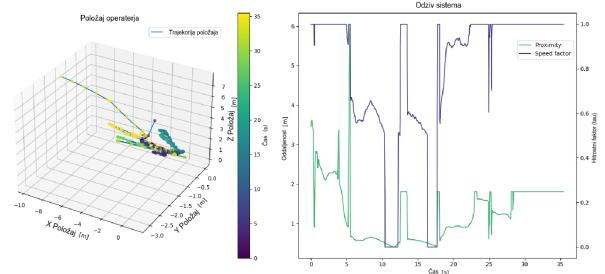
Na sliki 17 je prikazan odziv varnostnega sistema pri slabih svetlobnih pogojih, kjer je ambientalna svetlost znašala le 2 lux. Edina svetloba v laboratoriju je prihajala skozi okna, vsa umetna svetila pa so bila izključena.

Pri teh pogojih se v kamerah, posebno tistih z manjšimi senzorji in manjšimi zaslonkami začne pojavljati digitalni šum. Pri stereoskopskih globinskih kamerah se vpliv leta še poveča, saj se paralaksa računa med dvema slikama z velikim šumom, kar robove zamegli in algoritmi določanja globine ne delujejo optimalno. Odrav tega je večji šum pri zaznavanju globine na sliki 18. Prav tako digitalni šum škoduje tudi barvni sliki, na kateri AlphaPose zaznava človeške sklepe. Zaradi večjega šuma na sliki so vrednosti zaupanja zaznanih sklepov občutno nižje, pride pa lahko tudi do lažnih pozitivnih zaznav.

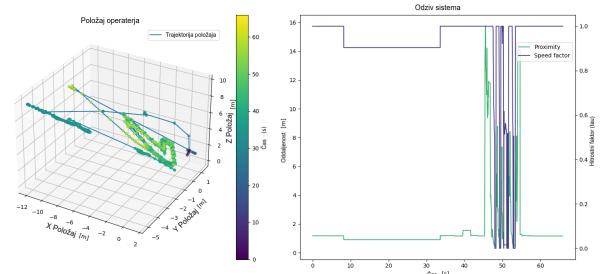
Na sliki 17 je razvidno, da je prišlo do lažne detekcije. Program AlphaPose je napačno zaznal človeka na detektorju dima. Iz grafa odziva je razvidno da je lažna detekcija trajala prvih 45 sekund meritve, preden je uporabnik vstopil v vidno polje kamere, nakar je bila zaznava občutno slabša, kot v primeru z optimalnimi svetlobnimi pogoji (razvidno iz nenadnih 'skokov' lokacije operaterja



Slika 15: Prikaz spremembe trajektorije zaradi prisotnosti operaterja v delovnem območju robota.



Slika 16: Prikaz premikanja operaterja v vidnem polju kamere (levi graf) in reakcija varnostnega sistema na vstop operaterja v vidno polje kamere (desni graf) pri optimalni osvetlitvi.



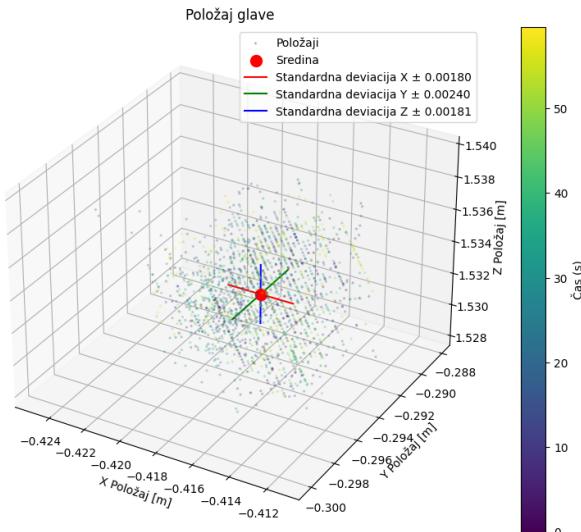
Slika 17: Prikaz premikanja operaterja v vidnem polju kamere (levi graf) in reakcija varnostnega sistema na vstop operaterja v vidno polje kamere (desni graf) pri slabi osvetlitvi

na levem grafu).

## 6 Zaključek

V tem projektu sem raziskal uporabo računalniškega vida za zagotavljanje varnosti v sodelujoči robotiki. Analiza obstoječih rešitev je pokazala, da je računalniški vid lahko kakovostna zamenjava za trenutne industrijske rešitve in lahko igra ključno vlogo pri zaznavanju in izogibanju trkom, prepoznavanju ljudi ter dinamičnem prilagajanju delovanja robotov v realnem času. Z uporabo sodobnih nevronskih mrež je mogoče doseči visoko natančnost pri obdelavi slikovnih podatkov, kar lahko bistveno olajša postavitev varnostnih sistemov in omogoči enostavnejšo ter varnejšo interakcijo med ljudmi in roboti.

Meritve nakazujejo konkurenčnost sistemov, ki temeljijo na uporabi računalniškega vida za zagotavljanje var-



Slika 18: Raztres lokacije detekcije glave operaterja



Slika 19: Lažna detekcija operaterja v slabih svetlobnih pogojih

nosti v sodeljujoči robotiki. To je posebej pomembno v času, ko se povpraševanje po pametnih in prilagodljivih proizvodnih procesih povečuje. Vendar pa je uporaba teh tehnologij še vedno omejena zaradi visoke računske zahetnosti in potrebe po robustnem delovanju v vseh pogojih.

## Literatura

- [1] G. Schöner, “Dynamic movement primitives.” Dosegljivo: [https://www.ini.rub.de/upload/file/1657204397\\_8b8ff6f0f9bea287ea25/10\\_dynamic\\_movement\\_primitives.pdf](https://www.ini.rub.de/upload/file/1657204397_8b8ff6f0f9bea287ea25/10_dynamic_movement_primitives.pdf). [Dostopano: 19. 4. 2024].
- [2] H.-S. Fang, J. Li, H. Tang, C. Xu, H. Zhu, Y. Xiu, Y.-L. Li, and C. Lu, “Alphapose: Whole-body regional multi-person pose estimation and tracking in real-time,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–17, 2022.
- [3] ISO 13855:2010, “Safety of machinery - Positioning of safeguards with respect to the approach speeds of parts of the human body,” 2015.
- [4] GT Engineering - consultants in machinery safety, “EN ISO 13855: Positioning of safeguards .” Dosegljivo: <https://www.gt-engineering.it/en/technical-standards/en-iso-standards/en-iso-13855/5-general-equations-for-the-calculation-of-the> [Dostopano: 11. 8. 2024].

gljivo: <https://www.gt-engineering.it/en/technical-standards/en-iso-standards/en-iso-13855/5-general-equations-for-the-calculation-of-the> [Dostopano: 11. 8. 2024].

- [5] M. Deniša, A. Ude, M. Simonič, T. Kaarlela, T. Pitkäaho, S. Pieskä, J. Arents, J. Judvaitis, K. Ozols, L. Raj, A. Czmerk, M. Dianatfar, J. Latokartano, P. A. Schmidt, A. Mauersberger, A. Singer, H. Arnarson, B. Shu, D. Dimosthenopoulos, P. Karagiannis, T.-P. Ahonen, V. Valjus, and M. Lanz, “Technology modules providing solutions for agile manufacturing,” *Machines*, vol. 11, no. 9, 2023.
- [6] Shanghai Jiao Tong University, Machine Vision and Intelligence Group, “AlphaPose.” Dosegljivo: <https://github.com/MVIG-SJTU/AlphaPose/tree/master>. [Dostopano: 22. 6. 2024].
- [7] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele, “2d human pose estimation: New benchmark and state of the art analysis,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.