Paletitzat intel·ligent amb l'ús de visió per computador i robot industrial

Guillem Centelles Pavon

Resum—Utilitzant tècniques de visió per computador, programació de robots induatrials i PLC, i comunicació entre dispositius, s'ha aconseguit desenvolupar un sistema físic per a l'automatització del paletitzat d'objectes d'un color concret en cadenes de producció amb cintes transportadores. El robot antropomòrfic de 6 eixos (model Staübli TX-60), a més de ser l'element central de tot el sistema desenvolupat, és l'encarregat d'agafar els objectes de la cinta i paletitzar en la seva posició i orientació correcta, gràcies al sistema de visió per computador sobre aquests objectes, i la comunicació amb el PLC de la cinta transportadora podem saber quan arriba un objecte si és el correcte, i si ha de continuar avançant. El sistema de visió per computador, desenvolupat amb la llibreria OpenCV de Python, s'encarrega de segmentar, sobre la imatge feta de l'objecte sobre la cinta transportadora, les regions d'aquesta que continguin el color buscat, separant-les de la resta de la imatge, per posteriorment eliminar les possibles regions de soroll que contingui, i finalment identificar la regió corresponent a l'objecte, per extreure'n la seva posició i orientació sobre la cinta.

Paraules clau—Paletitzat, Visió per Computador, Robot industrial, Programació, Sockets, Automatització, Cadena de producció, Robot antropomòrfic, PLC, OpenCV, Python, segmentació per color.

Abstract—By utilizing computer vision techniques, industrial robot and PLC programming, and device comunication, it has been achieved the development of a physical system capable of automating the palletization of colour-specific objects in production chains with conveyor belts. The 6 axis antropomorphic robot (model Staübli TX-60), apart from being the central element of the whole developed system, is in charge of picking up the objects from the conveyor belt and palletize them onto the correct position and orientation, thanks to the the computer vision system, regarding said object, and to communicate with the conveyor belt's PLC to know when the correct object is in postition, and if it needs to continue moving. The computer vision system, developed by using Python's OpenCV library, starts by segmenting the imatge of the object on the conveyor belt, to get the regions that contain the searched color, separating them from the rest of the image, to later remove the possible noise regions it contains, and finally identify the corresponding region of the object, in order to extract its position and orientation on the conveyor belt.

Index Terms—Palletization, Computer Vision, Industrial Robot, Programming, Sockets, Automation, Producction Chain, Antropomorphic Robot, PLC, OpenCV, Python, color segmentation.

1 Introducció

EL MÓN industrial està vivint actualment la seva quarta revolució provocada per la integració de noves tecnologies com la intel·ligència artificial, la robòtica i l'Internet de les Coses (Iot), entre altres, provocant l'aparició de l'anomenada "Indústria 4.0"[1].

Com a conseqüència, moltes empreses estan optant per a aplicar aquestes tecnologies en les seves cadenes de producció i instal·lacions, delegant tasques que actualment realitzen operaris, a les màquines amb sistemes intel·ligents per tal d'acoseguir augmentar la productivitat, reduir-ne els costos de producció, i poder donar una resposta més àgil als constants canvis en la demanda de mercat.

Amb aquestes idees en ment, aquest és un projecte concebut amb la intenció i finalitat d'aconseguir oferir la creació i implementació d'un sistema físic funcional, que combini tècinques de visió per computador per a reconèixer els objectes segons el seu color en una cinta transportadora, per posteriorment permetre seleccionar al robot quins d'aquests ha d'agafar per a realitzar-ne el paletitzat correctament.

Per al muntatge físic, es disposa d'un robot antropomòrfic Stäubli TX60[2][3], amb un manipulador de tipus ventosa de succió per buit per poder agafar els objectes de manera ràpida i còmoda. També es farà ús d'un sistema de visió per computador muntat sobre la cinta transportadora, la qual anirà transportant els diferents objectes a ser col·locats de manera ordenada en el palet.

En quant al sistema de visió per computador, aquest serà programat en Python, i es farà ús de la llibreria OpenCV[4] per a la detecció dels objectes i la seva orientació corresponent. Es farà ús d'una càmera connectada via USB a un ordinador per a poder fer les imatges necessàries.

- *E-mail de contacte: guillem.centelles@autonoma.cat*
- *Menció realitzada: Computació*
- *Treball tutoritzat per: Carlos García Calvo (Ciències de la Computació)*
- *Curs 2022/23*

2 OBJECTIUS

El principal objectiu d'aquest projecte és aconseguir un sistema físic capaç de paletitzar objectes d'un color específic, transportats per una cinta transportadora, segons les dimensions especificades prèviament sobre el palet final.

Per aconseguir-ho, es van determinar una sèrie de *milestones* durant la planificació del projecte, les quals representens les diferents fites respecte els mòduls i funcionalitats necessaris que s'utilitzaran en el sistema final per tal que aquest funcioni de la manera desitjada. Aquests objectius van des de l'aprenentatge de la programació específica del robot utilitzat, fins a les corresponents proves d'integració finals, passant per la creació del sistema de detecció d'objectes segons el seu color i la correcta comunicació dels elements del sistema entre ells.

Per a més detalls, es pot consultar la llista d'objectius del projecte en l'apèndix A1. Objectius del projecte.

3 PLANIFICACIÓ

En aquesta secció es detalla l'esquema de la planificació de les tasques realitzades durant el desenvolupament del projecte per tal d'aconseguir arribar a uns resultats desitjables i aquests siguin de bona qualitat, juntament amb els objectius que satisfan i la seva durada en setmanes.

Aquestes tasques es poden agrupar en les següents fases:

1. Primer contacte amb el robot i planificació del projecte.
2. Aprenentatge de la programació específica del robot.
3. Desenvolupament del codi del robot en simulador.
4. Desenvolupament del sistema de visió per computador.
5. Comunicació entre els elements del sistema.
6. Integració dels mòduls desenvolupats.
7. Proves de funcionament.

Per a més detalls, es pot consultar la taula de planificació en l'apèndix A2. Planificació del projecte.

4 MATERIALS

En aquesta secció s'explicaran el diferents dispositius, objectes i elements físics utilitzats en la realització del projecte, i necessaris per la seva implementació física final.

4.1 Robot antropomòrfic TX-60

Aquest robot industrial de 6 eixos de la marca Stäubli, model TX-60, és l'eix central de tot el projecte i el component més important d'aquest, ja que és el principal actor de tot el procés de paletitzat.

Entre les seves característiques es pot destacar la seva càrrega nominal de 3.5kg, que el converteix en una gran opció per a realitzar tasques de transport d'objectes de tamany i pes reduït, com per exemple taques de *'pick and place'* (agafar i col·locar). Gràcies a la seva configuració amb 6 graus de llibertat que li proporcionen tots els seus eixos, el robot té una molt bona maniobrabilitat, la qual cosa li permet moure's i arribar a tots els punts disponibles dins la seva àrea de treball esfèrica, centrada en la base del propi robot. També remarcar que, al tenir una repetibilitat de ± 0,02mm i una velocitat màxima a la que pot transportar la seva càrrega de 8m/s, el TX-60 és capaç de realitzar les tasques a gran velocitat, mentres que manté una alta precisió en els seus moviments. Finalment puntualitzar que pel desenvolupament d'aquest projecte s'ha escollit aquest robot per sobre de l'altra opció disponible, un robot SCARA[5] model TS-80 de Stäubli, perquè el TX-60 disposa del control per manipuladors de tipus pneumàtic o ventosa, i ofereix la possibilitat de connectar-hi un encoder per poder llegir-ne la seva velocitat de gir i poder agafar objectes en moviment.

Per a més detalls sobre les característiques del robot i la seva àrea de treball, es pot consultar en l'apèndix A3. Fitxa tècnica del robot TX-60.

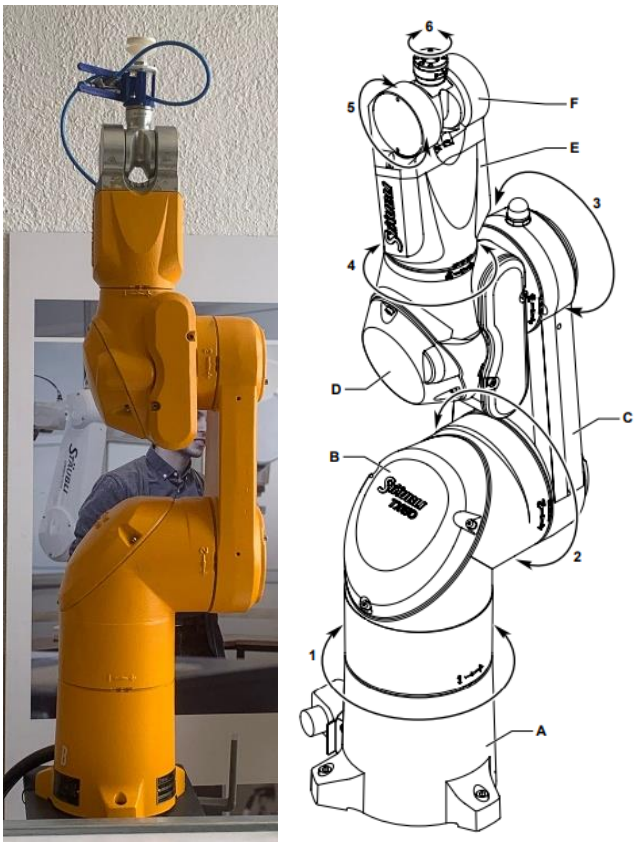

Fig 1. Robot TX-60 i el seus eixos de moviment.

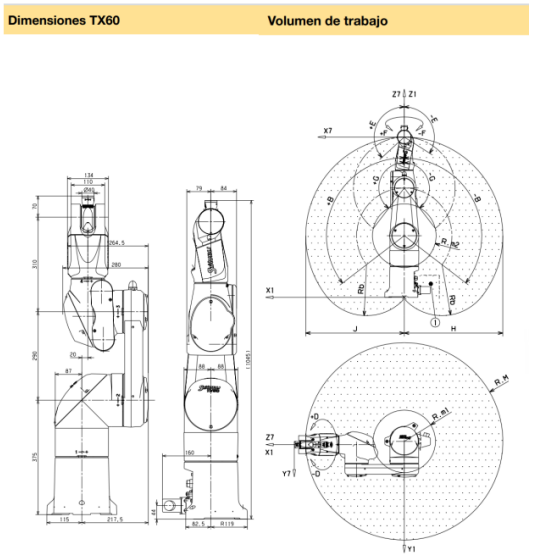

Fig. 2 Dimensions i àrea de treball del TX-60.

A més, per poder funcionar el robot necessita estar connectat a un controlador, que és l'encarregat de processar tota la cinemàtica necessària per a que el robot pugui fer els moviments necessaris, tenint en compte que aquestes accions i/o els punts on s'ha de moure siguin possibles dins l'àrea de treball, ja que els eixos tenen una rotació màxima possible. En el nostre cas, aquest controlador és un model CS8C de la mateixa Stäubli, el qual, a més de tenir connectat un panell de control per operaris, el qual conté una interfície gràfica per a poder controlar el robot i la seva configuració manualment, també disposa de totes les entrades i ports necessaris per aconseguir enviar i rebre dades externes al propi robot.

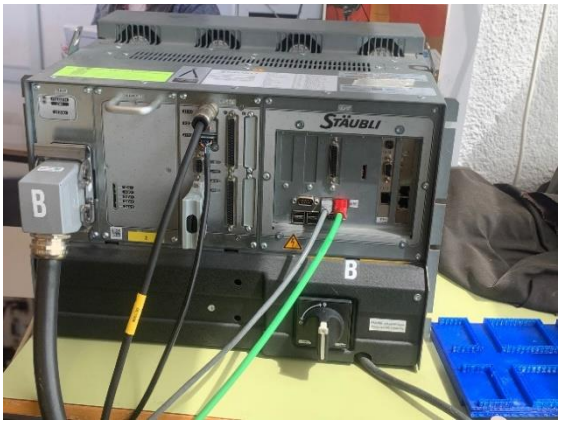

Fig. 3 Controlador CS8C del robot.

Tal i com es pot veure en la part superior de la Figura 1, sobre l'eix número 6 del robot es troba muntat el manipulador utilitzat en el projecte. En aquest cas es tracta d'un manipulador de tipus ventosa, el qual utilitza la tècnica del buit per mitjà de l'ús d'una bomba de buit, la qual consisteix a extreure l'aire que queda entre la ventosa i l'objecte a agafar, creant un espai de baixa pressió que acaba deixant l'objecte "enganxat" a la ventosa per així poder-lo moure al gust.[6]

4.2 Stäubli Robotics Suite

El Stäubli Robotics Suite (SRS)[7] és el simulador propi que ofereix la mateixa Stäubli per a la programació dels seus robots de maners remota. Tot el codi referent a la funcionalitat del robot d'aquest projecte ha estat desenvolupat dins d'aquest simulador, per a posteriorment poder exportar-ne les configuracions i programes desenvolupats des del simulador i carregar-los al robot real. Tal i com el defineixen des de Stäubli, "Stäubli Robotics Suite és un software ampli de programació per a PC que simplifica la gestió del sistema del robot. Des de la concepció i simulació fins a la configuració i producció, també permet una millora permanent de les aplicacions del robot. El software guia clarament a través del procés de simulació, desenvolupament i depuració dels programes del robot fins a la validació final i la transferència de les dades del sistema. A causa de la seva versatilitat d'aplicacions, és l'eina de producció intel·ligent ideal que ofereix suport multipantalla que s'executa al sistema operatiu Windows."

A més de permetre escriure el codi per la programació del robot i de mostrar-ne les coordenades de la seva posició i la configuració dels seus eixos en tot moment, el simulador també ofereix la possibilitat de configurar una escena en 3D, la qual serveix per a recrear la configuració real amb la que treballarà el robot, amb la possibilitat del usuari d'afegir models 3D, com poden ser altres elements amb els que interactuarà el robot i el manipulador que s'utilitzarà, entre altres, a més de poder activar el control de col·lisions entre objectes per evitar malemtre algun element en la realitat.

En quant a la programació específica del robot, tant el simulador com el controlador del robot es programen amb el llenguatge d'alt nivell propi de Stäubli, anomenat VAL3[8][9]. VAL3 és un llenguatge de programació modular, el qual està pensat per facilitar-ne la reutilització de codi i del coneixement après, a més de oferir una simplificació de la programació dels robots, ja que tots els càlculs necessaris per als moviments, tant de cinemàtica directa o inversa[10], els porta implementats directament en funcions pròpies del llenguatge, i només es necessita fer una crida a aquestes amb els paràmetres necessaris. També porta itegrades les funcions necessàries per la comunicació amb altres dispositius per mitjà de sockets, i ofereix la possibilitat de personalitzar la pantalla d'usuari quan s'executi un aplicació, la qual mostra en el panell de control integrat del controlador els missatges que es desitjin.

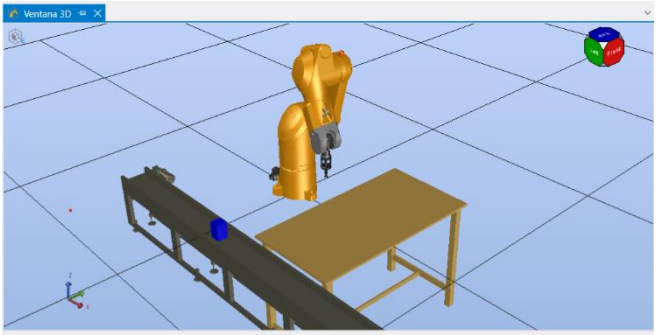

Fig. 4 Escena 3D en el simulador.

4.3 Cinta transportadora

La cinta trasportadora es l'encarregada de moure els objectes que es troben sobre aquesta fins al punt on el robot pugui agafar-los.

Connectats a la cinta, tenim primer un variador de freqüència model OMRON-MX2, el qual s'encarrega de controlar la velocitat de gir del motor de la cinta transportadora, i poder controlar-ne el moviment d'aquesta.

L'altre component connectat a la cinta és un PLC (Controlador Lògic Programable) model Siemens-1200, el qual, a més d'encarregar-se de regular el funcionament del variador de freqüència, també té connectat el sensor de presència muntat en la cinta transportadora per saber quan arriba un objecte que es pugui agafar. També és el PLC el que està directament connectat per mitjà d'un cable de Ethernet al controlador del robot, per tal d'habilitar-ne la comunicació entre els dos dispositius i que el robot pugui saber quan ha arribat un objecte en la cinta i aquesta s'aturi.

Importat puntualitzar que en aquest projecte no s'ha hagut de fer la programació explícita de cap d'aquests dos components, ja que aquests venien preparats amb les seves funcionalitats abans de l'inici del projecte, però tot i així s'ha après el seu funcionament i programació per tal de conèixer a fons els seu comportament i característiques.

Fig. 5 PLC Siemens 1200 (esquerra) i variador de freqüència OMRON-MX2 (dreta).

4.4 Càmera USB

Per al funcionament del sistema de visió per computador, s'ha decidit fer ús d'una càmera USB model Microsoft LifeCam Studio, la qual disposa d'un sensor d'alta definició de 1080p HD per major qualitat d'imatge. La càmera està muntada sobre del punt de la cinta transportadora on es troba el sensor de presència d'objectes i enfocant a aquesta, a una altura segura per a què el robot no pugui xocar contra els elements de suport de la càmera, per tal de capturar la imatge un cop arribi l'objecte i enviar-li al sistema de visió. Degut a l'altura a la que s'ha de col·locar la càmera, les imatges que aquesta faci mostren una àrea de la cinta transportadora i els elements del voltant superior a l'àrea de la regió d'interès a on es troba l'objecte. L'extracció de l'àrea d'interès i la detecció del color de l'objecte són tasques que s'encarregarà de resoldre el propi sistema de visió.

Per poder enviar la imatge, la càmera es troba connectada via USB a l'ordinador encarregat d'executar el sistema de visió perquè aquest rebi directament la imatge a tractar, i posteriorment envïi els resultats al controlador del robot, amb el que es troba connectat per mitjà d'un altre cable d'Ethernet.

4.5 Prismes de colors

Els objectes utilitzats per a fer el paletitzat segons el seu color han estat prismes rectangulars de base 5×5 cm i 8 cm d'altura, de colors blau, blanc i negre, impresos amb una impressora 3D, per tal de comprovar que el sistema funciona amb múltiples colors diferents, i detecta només aquells del color que interessi.

El model 3D d'aquests prismes ha estat també utilitzat en el simulador durant el desenvolupament del codi del robot per comprovar-ne el correcte funcionament abans de traspassar-ho a la realitat.

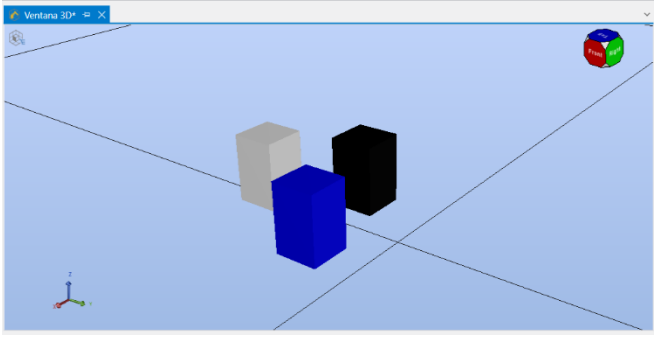

Fig. 6 Prismes de colors usats (vista del simulador).

5 DESEONVOLUPAMENT

A continuació s'explicarà i es detallarà les tasques i feina realitzada durant cada una de les diferents fases del desenvolupament del projecte.

5.1 Primera visita a les instal·lacions

En aquesta primera fase es va duu a terme la primera presa de contacte amb els dos robots industrials disponibles, els quals m'han facilitat l'accés des de l'Institut Anna Gironella de Mundet a Barcelona, en la seva aula de robòtica. Durant aquesta fase va ser quan es va prendre la decisió de treballar sobre el robot TX-60 pels motius especificats anteriorment en l'apartat 4.1 d'aquest document.

També va ser durant aquest primera fase quan es va poder tenir una visita guiada a la planta de Stäubli Espanya SAU, durant la qual ens van explicar i mostrar els seus productes i robots que ofereixen, a més de poder veure de primera mà el seu funcionament gràcies a unes demos tècniques que tenien preparades. També vaig tenir el privilegi

de parlar amb la gent de l'empresa sobre la tendència actual de la robòtica en la Indústria 4.0, que cada cop es busca integrar més funcionalitats de visió per computador en els robots per tal de resoldre tasques que fins l'augment en l'ús de tècniques com per exemple el Deep Learning dels últims anys no eren possibles d'automatitzar eficaçment, sobretot orientat a problemes de classificació d'objectes.

5.2 Experimentació amb el simulador

Durant aquesta fase es va aprendre el llenguatge de programació VAL3 específic del robot, juntament amb la seva sintaxis, les funcions principals per a controlar el moviment i comunicació del robot a través de l'ús del simulador SRS de Stäubli, a més d'aprendre el sistema de referència i les corresponents coordenades dels punts dins l'espai de treball disponible del propi robot.

Per aconseguir això, abans de començar a desenvolupar el codi final del projecte, es van fer una sèrie de programes més petits per fer les corresponents proves i entendre el funcionament, tant del VAL3, com el del propi simulador i la seva configuració.

5.2.1 Funcions principals de VAL3

L'objectiu d'aquest subapartat és explicar algunes de les funcions més utilitzades en el codi definitiu del robot per a mostrar-ne les seves funcionalitats.

movej(): Funció que rep per paràmetres la posició en graus dels 6 eixos del robot, el manipulador utilitzat, i la velocitat de moviment amb la que es farà el moviment, per a aplicar la cinemàtica directa al robot a partir dels angles especificats.

movel(): Funció que rep per paràmetres la posició i orientació en coordenades del robot (x,y,z,rx,ry,rx), el manipulador utilitzat, i la velocitat de moviment amb la que es farà el moviment, per a aplicar calcular la cinemàtica inversa del robot a partir dels punts especificats i moure's desde la posició inicial fins al destí en línia recta.

waitEndMove(): Funció que obliga al robot a acabar d'executar l'acció que tingui en progrés abans de poder continuar executant-ne de noves.

sioGet(): Funció que té com a paràmetres un socket de comunicació, i una variable a on guardar el valor. La funció guarda en la variable el valor que rebi pel socket especificat.

sioSet(): Funció que té com a paràmetres un socket de comunicació, i una variable amb el valor a enviar. La funció envia el valor de la variable especificada al socket corresponent.

clearBuffer(): Funció que té com a paràmetres un socket, i s'encarrega de buidar-ne totes les dades que hi hagin en aquest.

open(): Funció que rep per paràmetre el manipulador, i li canvia l'estat a obert, en el cas de la ventosa al vuit, aquest és l'estat en el que creal el buit per agafar l'objecte.

close(): Funció que rep per paràmetre el manipulador, i li canvia l'estat a tancat, en el cas de la ventosa al buit, aquest és l'estat en el que deixa de crear el buit i deixa d'agafar l'objecte.

userPage(): Funció que canvia la pantalla mostrada en el panell de control del controlador a la pàgina d'usuari, sobre la que es poden mostrar missatges personalitzats.

gotoxy(): Funció que rep per paràmetres les coordenades (x,y) de la posició en la pantalla d'ususari a on situar el cursor per puguer inserir un missatge.

put(): Funció que rep per paràmetres un string amb el missatge que es vol mostrar per la pantalla d'usuari a la posició on es trobi el cursor actualment.

5.3 Primera versió del codi de paletitzat

Durant aquesta fase i, un cop conegut el funcionament i programació del robot amb el simulador, es va decidir realitzar una primera versió del codi necessari per paletitzar objectes a partir de saber-ne la seva posició inicial i les dimensions del paletitzat final, però sense la implementació del sistema de visió ni la comunicació del robot amb el PLC.

5.3.1 Esquema de connexions

A més, en aquest punt del projecte ja es va fer el disseny del esquema de connexions del projecte final.

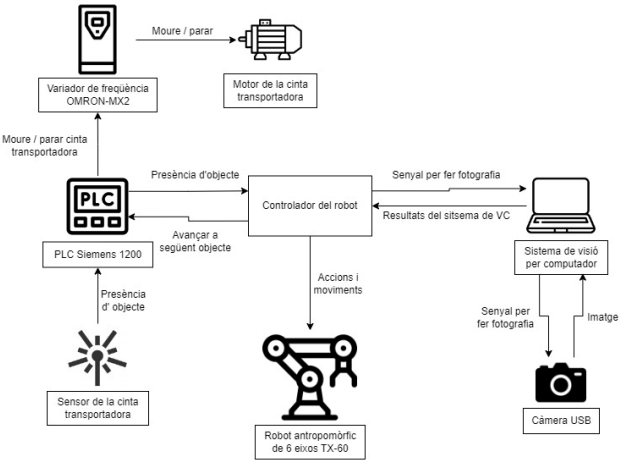

Fig. 7 Esquema de connexions de la solució final.

Com es pot veure en la Figura 6, l'esquema de connexions situa al robot TX-60, i més específicament el seu controlador, al centre de tota la solució final del projecte, fent que el codi del robot sigui l'encarregat de fer el control de tots els diferents passos i accions que ha de realitzar el robot, i servir de servidor/client a la resta de components.

Connectat al controlador del robot tenim, per una banda l'ordinador amb el sistema de visió per computador, el qual un cop rebi el senyal des del robot, s'encarregarà de fer la fotografia amb l'ús de la càmera USB connectada, per posteriorment aplicar-li les corresponents tècniques de visió per computador per, primer trobar si l'objecte s'ha de paletitzar, i si és el cas, trobar-ne la seva posició i orientació corresponents, i retornar-ne els resultats al robot.

Per altra banda, trobem la connexió entre el controlador del robot i el PLC de la cinta transportadora, el qual està

constantment enviant al robot si algun objecte passa pel sensor de la cinta, i a la vegada, aturant el moviment de la cinta quan es detecti un objecte. El robot també por enviar-li un senyal en el cas que l'objecte no s'hagi de paletitzar per tornar a posar en marxa la cinta.

5.3.2 Codi del robot desenvolupat

En quant a la primera versió del codi del paletitzat, aquesta comença amb la declaració dels 3 bucles for anidats, cadascun dels quals s'encarreguen d'anar canviant la posició a on s'ha de col·locar el pròxim objecte del palet en cada iteració, essent la seva jerarquia de bucle més intern cap al més extern en el següent ordre: Y, X, Z. Seguint aquest ordre, el codi comença recorrent totes les posicions primer del eix Y, després es desplaça sobre l'eix X, i finalment sobre l'altura marcada per l'eix Z. Important mencionar també que el motiu pel qual s'ha decidit recórrer els eixos del palet en aquest ordre ha estat per tal d'aconseguir que cada cop que es col·loqui un objecte en el palet, durant el recollegut d'aquest fins la seva posició final no s'hagi de creuar per la posició d'algun altre objecte que ja hagi sigut paletitzat per tal de eliminar la possibilitat de col·lisió entre objectes.

Un cop tenim els bucles delcarats, el programa comença movent al robot al punt d'apropament a la posició on es troba l'objecte sobre la cinta, també anomenat punt de 'approach' de la cinta, el qual és el mateix valor en posició (x,y) que a on es troba l'objecte sobre la cinta, però en aquest cas a una altura de 8cm superior al punt de recollida de l'objecte. Es segueix aquest procediment abans d'anar al punt de recollida de l'objecte sobre la cinta per tal d'assegurar-nos que el robot no faci cap moviment estrany que pugui malmetre altres elements, o xoqui directament amb algun element dins de l'àrea de treball.

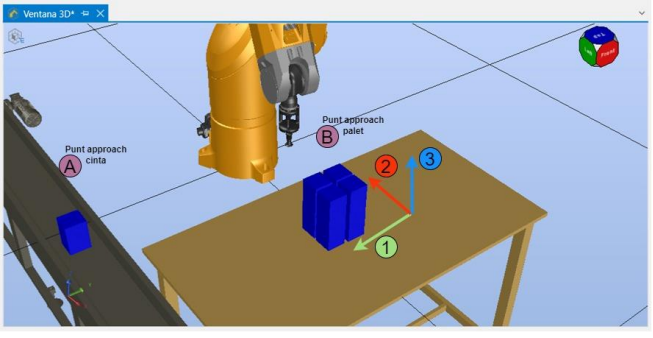

Fig. 8 Ordre de recorregut dels eixos en el paletitzat.

Quan el robot ha arribat a la posició de 'approach' al objecte de la cinta (Fig.8, punt A), llavors és quan es mou fins al punt d'agafada, situant la ventosa sobre la superfície superior de l'objecte, i es crea el buit per tal de poder agafar-lo.

Amb l'objecte agafat, el robot retorna a la posició de 'approach' de la cinta transportadora, per tal d'assegurar-nos que en posteriors moviments aquest no col·lisioni contra els laterals de la cinta, i posteriorment mou l'objecte al punt de 'approach' del palet (Fig.8 punt B).

Aquest punt encara no és el punt de 'approach' de la pròpia posició del palet que li pertoca a l'objecte, ja que un altre cop, per evitar que el robot pugui xocar contra altres elements del seu entorn, o amb alguns objectes ja paletitzats. Afegir també que un cop el robot ha agafat l'objecte de la cinta i l'aixeca d'aquesta, a part de tornar al punt de 'approach' també aprofita per corregir-ne l'orientació que pugui tenir l'objecte perquè pugui col·locar-lo totalment recte posteriorment.

Quan el robot ja està situat en el punt de 'approach' al palet (Fig. 9, punt 4), llavors és quan desplaça l'objecte fins al punt d'aproximació a la posició del palet que li pertoca (Fig. 9, punt 5), situat a 3cm d'altura respecte el punt de col·locació sobre el palet (Fig. 9, punt 6). S'ha escollit aquest valor d'altura, perquè en cas que hi hagi un objecte prèviament a l'inici del paletitzat sobre el palet, el mateix robot l'aparti a l'hora d'anar a col·locar l'objecte, evitant que es pugui trencar el manipulador en el procés.

Ara ja només li queda al robot anar finalment al punt a on ha de situar l'objecte sobre el palet, tornar a deixar entrar aire en la ventosa del manipulador per tal de deixar anar l'objecte, i realitzar els mateixos moviments per col·locar l'objecte, però en ordre invers (Fig. 9, punts 6, 7, 8), acabant en el punt 8 de la Figura 9, el qual també funciona com a posició d'espera pel següent objecte.

Aquest procés es va repetint per cada objecte que arribi a la cinta i s'hagi de paletitzar, fins que el robot acabi el paletitzat amb les dimensions especificades inicialment.

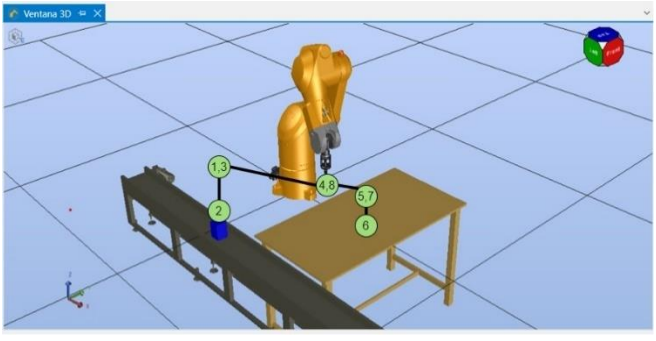

Fig. 9 Escena en el simulador amb la representació del ordre de moviments seguits pel robot.

5.4 Sistema de visió per computador

Durant aquesta fase, s'ha desenvolupat tot el codi relacionat amb els mòduls necessaris per a la implementació del sistema de visió amb un ordinador i la càmera USB. El sistema és capaç de detectar si un objecte és d'un color en concret, i després saber-ne la seva orientació i les coordenades del seu centre en la imatge[11]. Posteriorment, calcula les coordenades de l'objecte en l'espai de treball del robot. Totes les funcionalitats del sistema de visió han estat desenvolupades en Python, per mitjà de l'ús de la llibreria OpenCV, juntament amb les llibreries Math i NumPy.

5.4.1 Imatge inicial

Un cop el sistema de visió rep el senyal corresponent, la càmera pren la imatge inicial de l'estat de la cinta transportadora amb l'objecte.

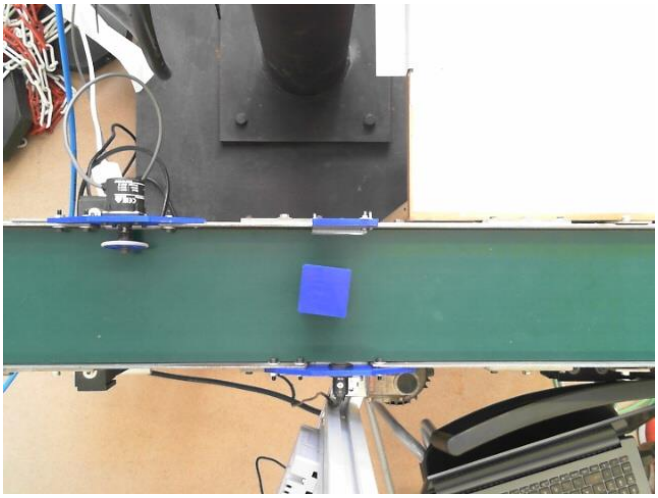

Fig. 10 Imatge inicial de l'objecte.

Tal i com es pot apreciar en la Figura 9, en la imatge es pot veure l'objecte sobre la cinta transportadora, però degut al ampli camp de visió de la càmera, sumat a la distància a la que es troba de la pròpia cinta, provoca que en la imatge hi apareixin molts elements que no interessen, i que podrien afectar negativament al rendiment del sistema de visió. Per solucionar-ho, es retalla d'aquesta imatge una àrea de 120×120 px dins la que es troba l'objecte en la cinta, i es fa un reescalat a 500×500 px per a augmentar-ne les seves dimensions. S'ha optat per realitzar les operacions de retall i escalat respecte la imatge original perquè, degut a les limitacions de la pròpia càmera utilitzada, ja que aquesta no permet canviar-li la òptica a una de focal més llarga. També, i tal com s'ha especificat en l'apartat 4.4 d'aquest document, la càmera està situada a una altura de seguretat per evitar col·lisions amb el robot i no és possible apropar-la més a la cinta per tenir solament l'àrea d'interès on es troba l'objecte en la imatge inicial.

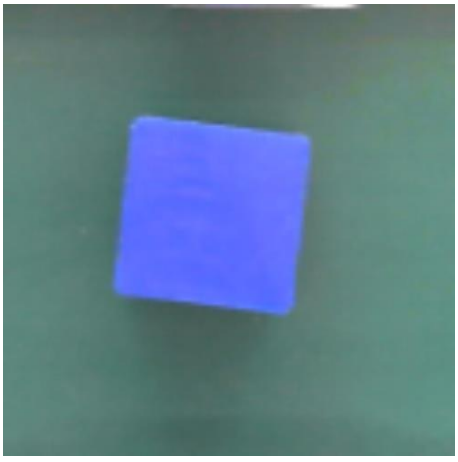

Fig. 11 Retall reescalat de la imatge original.

5.4.2 Segmentació per colors

Un cop s'ha aconseguit el retall de l'àrea d'interès, a continuació es necessita segmentar les regions de la imatge que siguin del color buscat. En el cas d'aquest projecte, el color dels prismes a paletitzar és el blanc, encara que també s'ha provat amb objectes de color blau com el que es pot veure en la Figura 10.

Per aconseguir fer aquesta segmentació, el retall es passa primer de l'espai de colors RGB al HSV[12] per facilitar-ne la definició del rang de valors acceptables. Amb la imatge en HSV, se li aplica la màscara del color corresponent, la qual selecciona aquells píxels de color blanc més purs, fina a aquells blancs amb un valor de lluniositat mínim del 66 %. Amb això s'aconsegueix una nova imatge binària amb els píxels del color buscat a 1 (blancs) i la resta a 0 (negre).

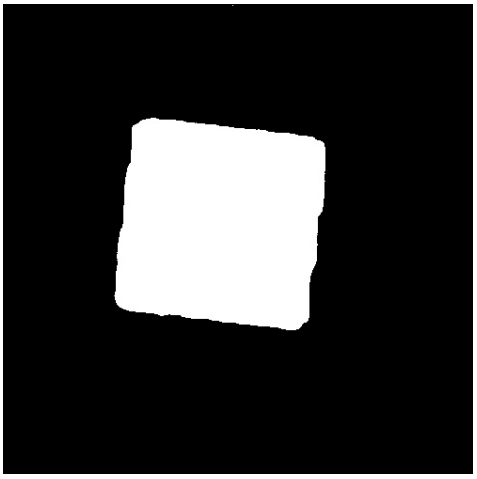

Fig. 12 Imatge binària de la segmentació de color.

5.4.3 Eliminació de soroll

Ara que ja tenim la imatge binària segmentada, aplicarem a la imatge la operació morfològica[13] anomenada 'open', la qual consisteix en aplicar-li a la imatge una opertura morfològica, fent primer una operació de erosió, per seguidament aplicar-li una altra operació de dilatació, utilitzant el mateix element estructural en les dues operacions. L'objectiu d'aquesta operació és eliminar elements que aporten soroll a la imatge, com ho són objectes petits i línies fines, a la vegada que es conserva la forma i mida dels objectes més grans.

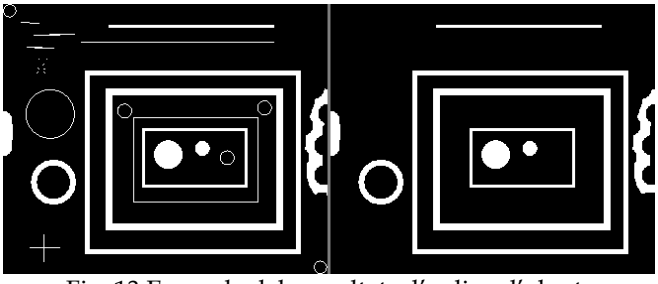

Fig. 13 Exemple dels resultats d'aplicar l'obertura morfològica a una imatge binària.

5.4.4 Posició de l'objecte

Una vegada hem eliminat el soroll de la imatge, ara ens queda buscar el contorn de l'objecte en la imatge. Si no es troba cap contorn, significa que l'objecte de la imatge no és del color que busquem i, per tant, no s'ha de paletitzar. En cas de trobar el contorn de l'objecte, es buscarà calcular-ne el rectangle que es correspongui al voltant de la regió on s'ha detectat l'objecte i dibuixar-lo sobre la imatge original. A partir dels punts que conformen els vèrtexs del nou rectangle, es calcula el centre d'aquest, el qual correspon al centre de l'objecte en la imatge ja que es treballa amb quadrats. En el cas de treballar amb formes més complexes, caldria ajustar aquest càlcul i trabr el centre de masses de l'objecte.

A partir de les coordenades del centre de l'objecte en la imatge (x_i, y_i), es pot calcular les coordenades en l'espai de treball del robot (x_r, y_r). Per fer-ho, es necessita conèixer en quins punts en coordenades de l'espai del robot comencen i acaben els costats de la imatge, per tal de poder calcular el factor de relació entre píxel-realitat de la següent manera:

$$f_r = Distància\ real/Número\ píxels.$$

Amb el factor de relació calculat, sabent a partir de quins valors de coordenades dels eixos del robot comencen els laterals de la imatge i coneixent les coordenades en la imatge del centre de l'objecte, es calcula la posició real de la següent manera:

$$(x_r, y_r) = ((x_i \times f_r + I_x), (y_i \times f_r + I_y))$$

(x_r, y_r): Coordenades en l'espai del robot del centre de l'objecte.
(x_i, y_i): Coordenades del centre de l'objecte en la imatge.
f_r: Factor de relació píxel-realitat.
(I_x, I_y): Coordenades en l'espai del robot de les cantonades de la imatge.

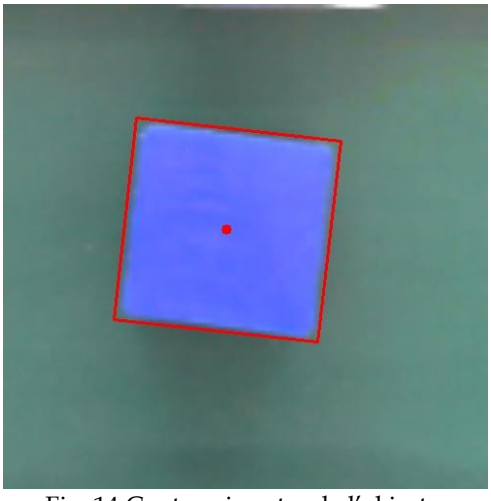

Fig. 14 Contorn i centre de l'objecte.

5.4.5 Orientació de l'objecte

Un cop ja tenim les coordenades reals de l'objecte a partir dels càlculs sobre el seu contorn, es calcula la orientació de l'objecte a partir de calcular-ne l'angle que hi ha entre el pròpi contorn, i la cantonada de la imatge i, en cas que el valor de la orientació sigui negatiu, es calcula el seu valor equivalent en positiu que, com la superfície del prisma és quadrada, el valor positiu equivalent surt del càlcul $90 - |angle\ negatiu|$.

Per a més detalls sobre les funcions desenvolupades, es pot consultar l'apèndix A4. Funcions del sistema de VC.

5.5 Comunicació entre els dispositius

Una vegada ja es tenen els diferents mòduls del projecte funcionant correctament per separat, toca aconseguir que aquests es puguin comunicar entre ells correctament, i tots els mòduls rebin els senyals que pertoquen.

Per començar, i tal com ja s'ha esmentat amb anterioritat, la comunicació entre el PLC i el controlador del robot, connectats entre ells per un cable de Ethernet, es realitza per mitjà d'un socket en el que s'envien les dades necessàries. Mentres el sensor de la cinta transportadora no detecta cap objecte, el PLC va enviant contínuament el valor 99 ('c' en ASCII) cap al robot, i ho fa un total de 10 vegades per segon. Un cop el sensor de la cinta detecta la presència d'un objecte, el PLC passa a enviar contínuament el valor 100 ('d' en ASCII) cap al robot, per indicar-li que ja hi ha un objecte en posició. Després de fer tot el processat de la imatge amb el sistema de VC, si es dona el cas que l'objecte no s'ha de paletitzar, és llavors el controlador del robot qui envia cap al PLC el valor 110 ('n' en ASCII) per indicar-li que la cinta ha de continuar avançant i desactivar el sensor temporalment fins a rebre un altre valor diferent a 110.

Tornar a recordar que la programació del PLC, tant per la comunicació amb el robot com el propi funcionament lògic del PLC han estat programats prèviament a l'inici del projecte, però igualment s'ha investigat per a conèixer el seu funcionament.

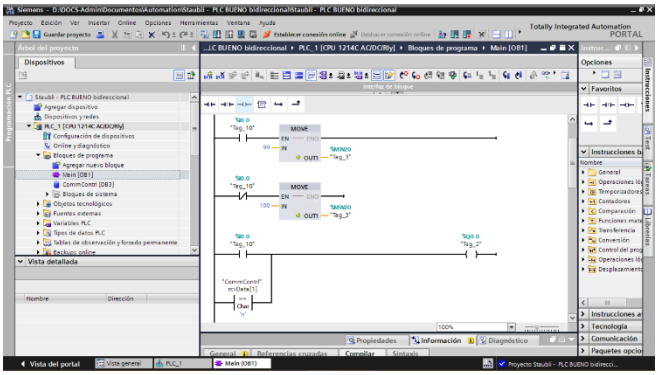

Fig. 15 Interfície de configuració del PLC.

En quant a la comunicació entre el robot i el sistema de visió per computador, s'ha programat tota les funcionalitats amb Python, amb l'ús de la llibreria Socket especialitzada en la comunicació per sockets. Al igual que amb el PLC, el controlador del robot i el ordinador estan conectats per mitjà d'un cable Ethernet.

Entrant en detalls de la programació d'aquesta comunicació, s'ha configurat el controlador del robot com a servidor de la connexió, i l'ordinador com a client, el qual es connecta a la direcció IP del robot (172.31.0.1), i s'obre la comunicació al port 11001 en ambdós casos. Un cop configurada la connexió, l'ordinador com a client es queda escoltant a aquest port, esperant a que el robot li envii el valor 70 ('F' en ASCII), per tal de fer la fotografia i fer ús del sistema de visió per detectar l'objecte. Si no es detecta l'objecte, l'ordinador retorna el valor 110 ('n' en ASCII) al robot per que aquest faci continuar a la cinta. Si el sistema de VC detecta l'objecte a paletitzar, l'ordinador envia el valor 89 ('Y' en ASCII) al robot, seguit de les dades de l'objecte posició en l'eix X, posició en l'eix Y, i la orientació.

Un cop enviades les dades de l'objecte, l'ordinador es queda en espera escoltant el port fins a que el robot li envii el valor 88 ('X' en ASCII) per tal de indicar-li el fi de la connexió i es tanqui el socket utilitzat.

5.6 Instal·lació del sistema final

A partir del moment en el que tots el mòduls del projecte funcionen correctament, i es poden comunicar entre ells sense problemes, s'ha fet la integració final del sistema físic de paletitzat intel·ligent.

Per començar, s'ha actualitzat la versió del codi del robot antiga per tal de tenir implentades totes les funcions necessàries per a la comunicació amb els altres dispositius, i se li ha afegit el codi necessari per a que es pugui mostar en el panell de control informació a l'operari sobre l'estat en tot moment del paletitzat, així com les dades que rep dels objectes que s'han de paletitzar.

Per a més detalls sobre el codi del robot, es pot consultar l'apèndix A5. Codi font definitiu del robot TX-60.

Un altre element important és la cal·libració de la càmera a l'hora de muntar-la en la seva posició sobre la cinta, ja que al retallar part de la imatge que pren per fer totes les operacions i càlculs necessaris, aquests són altamente dependents de la correcta posició i orientació de la càmera, ja que, si aquesta no está ben instal·lada, el sistema de visió pot generar resultats incorrectes. A més, també és important que la il·luminació de la sala sigui amb una font de llum difusa, ja que en cas contrari la reflexió de la llum sobre la cinta o l'objecte pot portar a errors del sistema de visió.

També és en aquest punt del projecte en el qual s'han realitzat totes les proves de funcionament i d'integració necessàries per a asegurar el bon funcionament de tot el sitems final.

6 Resultats

Després d'haver realitzat les proves de funcionament del sistema final, s'ha comprovat com el sistema de visió reconeixia com a objectes a paletitzar aquells prismes de color blanc, mentres que els utilitzats per les proves de colors blau i negre identificava que no eren els buscats, comportament totalment aliniat amb uns dels objectius principals d'aquest projecte.

Un altre dels objectius del projecta era que el sistema de visió aconseguís extreure'n la posició i orientació de l'objecte per a que el robot pogués agafar-lo en el punt central i en corregís la orientació, per tal que en el palet final tots els objectes estiguin orientats de la mateixa manera, i separats entre ells de manera equidistant. Després de les proves de funcionament, s'ha vist que en el palet final, alguns dels objectes tenen petits errors en la posició final de ±5mm, i en la seva orientació errors màxims de $\pm 5^{o}$, que provenen de la errada típica de la ventosa al agafar i deixar els objectes. Encara que aquests errors estiguin presents, considero que són pràcticament menyspreables, ja que aquestes petites variacions afecten en una molt petita mesura als resultats finals del paletitzat, per tant, els resultats d'aquest objectius han estat satisfactoris.

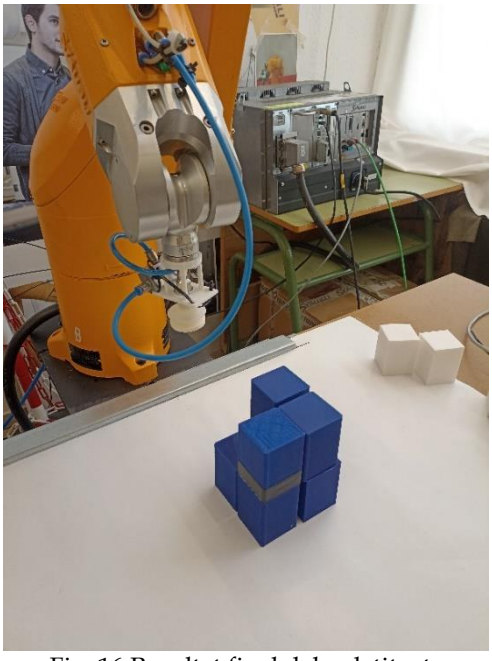

Fig. 16 Resultat final del paletitzat.

L'objectiu corresponent a aconseguir la correcta comunicació entre tots els dispositius del sistema ha estat completament assolit, ja que sense complir-lo, no s'hagués pogut arribar a fer les proves del sistema final, i molt menys aconseguir que tot el sistema físic final funcioni correctament.

7 Conclusió i possibles millores

Tot i que, com he esmentat en l'apartat anterior, hi ha alguns errors a l'hora de la col·locació dels objectes del paletitzat i les seves orientacions penso que els valors d'errors són molt acceptables, ja que hi ha molts elements del

sistema que, si no estan funcionant perfectament, poden acabar sent una font d'errors com els trobats en aquest projecte. Un exemple molt clar d'això és el muntatge de la càmera, ja que si no es delimita pràcticament a la perfecció les coordenades reals del robot de l'inici de les cantonades de la imatge, el càlcul de la posició no serà perfecte i generarà ja errors en la posició real de l'objecte, al igual que el propi manipulador del robot a vegades no acaba posicionant perfectament els objectes en la posició que pertoca a l'hora de deixar-lo anar, afegint també un petit error als moviments.

En quant a les possibles millores que es podrien fer al projecte, la primera seria fer-ne una revisió dels components, principalment del muntatge de la càmera i el propi manipulador de tipus ventosa, per tal d'acconsegui eliminar els petits errors presents actualment.

Una altra possible millora seria la incorporació d'un encoder situat sobre la cinta transportadora, i connectar aquest directament al controlador del robot per tal de saber la velocitat de moviment de la cinta i així poder modificar la programació del robot per tal que aquest sigui capaç d'agafar els objectes en moviment de la pròpia cinta. Aquesta millora era una de les idees inicials ques es volia intentar implementar en aquest projecte, però degut a la falta del component necessari per a connectar el encoder al controlador del robot es va decidir abandonar.

També es podria intentar afegir un segon sistema de visió per tal de poder tenir una identificació automàtica de les posicions disponibles del palet i la seva orientació, per tal de tenir la possibilitat de paletitzar els objectes en un palet que no estigui sempre en la mateixa posició i orientació.

Per acabar i en manera de resum, el projecte ha estat completat amb uns resultats molt satisfactoris, ja que s'han complert pràcticament tots els objectius que s'havien proposat inicialment al projecte, i els errors que hi ha són molt petits, i no acaben afectant significativament al resultat final. També puntualitzar que ha estat un projecte molt interessant des de la vista personal, ja que m'ha permès, a més de la programació habitual d'aquest tipus de projectes, no només poder treballar amb els components físicament, com amb el robot TX-60, sinó que poder veure com el sistema físic funciona correctament al igual que en les simulacions, la gent que he conegut del món de la robòtica, i el haver pogut aprofundir els meus coneixements en aquest camp ha sigut molt satisfactori i enriquidor personalment.

AGRAÏMENTS

En aquesta secció m'agradaria donar el merescut reconeixement a totes aquelles persones que, d'alguna manera o altra, han ajudat durant el desenvolupament del projecte i que, sense la seva presència, no hagués estat possible arribar als resultats final d'aquest treball.

A Carlos García Calvo, qui ha estat el meu tutor durant tot el trajecte que ha suposat la realització d'aquest projecte, i la seva gran implicació perquè les coses sortissin endavant, sempre estant disponible per a qualsevol problema que hagi sorgit i oferint les solucions adequades en cada situació.

A la gent de l'Institut Anna Gironella de Mundet a Barcelona per permetre'm treballar en una de les seves aules, i fer ús del el robot industrial i el seu material necessari pel funcionament del sistema final, a més de deixar-nos preparat i programat el PLC de la cinta transportadora.

A la gent de la planta de Stäubli Espanya SAU, qui ens van oferir una visita guiada per aquesta, juntament amb les explicacions del funcionament dels seus robots i, en especial, agrair a Josep Mª Serra que m'expliqués la tendència actual d'integrar sistemes de VC en el món de la robòtica i oferir diverses idees de possibles projectes. També agrair a Albert Olivella, qui m'ha proporcionat ajuda en totes les qüestions relatives al simulador de Stäubli.

BIBLIOGRAFIA

[1] «¿Qué es la Industria 4.0?», Deloitte. [Online] Disponible a: https://www2.deloitte.com/es/es/pages/manufacturing/articles/que-es-la-industria-4.0.html [Accedit: 08-mar2023]

[2] «OTHER ROBOTS STAUBLI TX60 CONTROL CS8C», Eurobots. [Online] Disponible a: https://www.eurobots.net/Other-Robots-robots-Staubli-TX60--control-CS8C-p67-en.html [Accedit: 08-mar2023]

[3] «TX2-60 industrial robot range», Stäubli. [Online] Disponible a: https://www.staubli.com/tw/en/robotics/products/industrial-robots/tx2-60.html [Accedit: 08-mar2023]

[4] «OpenCV modules», OpenCV. [Online] Disponible a: https://docs.opencv.org/4.x/index.html [Accedit: 12-mar2023]

[5] «¿Qué es un robot Scara? Aplicaciones, fabricantes y ejemplos», Revista de Robots. [Online] Disponible a: https://revistaderobots.com/robots-y-robotica/robot-scara-articulados-caracteristicas-y-marcas/ [Accedit: 08-mar2023]

[6] «Técnica del vacío», VAMA. [Online] Disponible a: https://www.vama.de/es/tecnica/tecnica-del-vacio/ [Accedit: 15-mar2023]

[7] «Stäubli Robotics Suite», Stäubli. [Online] Disponible a: https://www.staubli.com/cn/en/robotics/products/robot-software/staeubli-robotics-suite.html [Accedit: 05-abr2023]

[8] «Manual formación VAL3"», Stäubli. Manual de formació d'usuari. 2017.

[9] «Val3 Reference Manual», Stäubli. [Online] Disponible a: https://usermanual.wiki/Document/val3referencemanual.275627616/help [Accedit: 10-jun2023]

[10] «Cinemática de robots», Universitat Miguel Hernández. [Online] Disponible a: https://nbio.umh.es/files/2012/04/practica2.pdf [Accedit: 10-jun2023]

[11] «Object detection», Wikipedia. [Online] Disponible a: https://en.wikipedia.org/wiki/Object_detection [Accedit: 08-mar2023]

[12] «Entendiendo los Espacios de Color», DataSmarts. [Online] Disponible a: https://datasmarts.net/es/entendiendo-los-espacios-de-color/ [Accedit: 12-mar2023]

[13] «Tipos de operaciones morfológicas», The MathWorks. [Online] Disponible a: https://es.mathworks.com/help/images/morphological-dilation-and-erosion.html [Accedit: 08-mar2023]

APÈNDIX

A1. OBJECTIUS DEL PROJECTE

CODI	DEFINICIÓ D'OBJECTIU
[O1]	Treballar i aprendre el funcionament i programació del robot antropomòrfic.
[O2]	Realitzar el '*pick and place*' dels objectes en el simulador, juntament amb el seu posterior paletitzat.
[O3]	Realitzar el reconeixement de colors i orientació dels objectes mitjançant visió per computador.
[O4]	Comunicar el sistema de visió amb el robot antropomòrfic.
[O5]	Implementació física d'una primera versió del paletitzat amb els objectes estàtics.
[O6]	Realitzar totes les proves d'integració necessàries.

A2. PLANIFICACIÓ DEL PROJECTE

Planificació			
Fase	Descripció	Objectius	Durada aprox.
1	Primera visita a les instal·lacions per veure els robots.	-	1 setmana
2	Començar amb la programació del robot i familiarització amb el simulador.	[O1]	3 setmanes
3	Crear una primera versió del programa per a agafar i paletitzar els objectes correctament.	[O2]	2 setmanes
4	Creació del sistema de visió per computador per al reconeixement dels colors dels objectes i la seva orientació.	[O3]	3 setmanes
5	Aconseguir la correcta comunicació entre el sistema de visió i el robot antropomòrfic.	[O4]	2 setmanes
6	Construcció i implementació física de la primera versió del sistema, amb tots el components necessaris funcionant.	[O5]	2 setmanes
7	Realitzar la instal·lació física final del projecte.	[O6]	3 setmanes

A3. FITXA TÈCNICA DEL ROBOT TX-60

MODEL	TX60
Maximum load	4.5 kg, 9.9 lb (9 kg,18.8 lb under condition)
Nominal load	3.5 kg, 7.7 lb
Reach (between axis 1 and 6)	670 mm, 26.3 in
Number of degrees of freedom	6
Repeatability - ISO 9283	± 0.02 mm
Stäubli series controller	CS8C
Weight	51.4 kg, 113.32 lb
MAXIMUM SPEED	
Axis 1	435°/s
Axis 2	410°/s
Axis 3	540°/s
Axis 4	995°/s
Axis 5	1065°/s
Axis 6	1445°/s
Maximum speed at load gravity center	8 m/s
Maximum inertia axis 5	0.325 kg.m²
Maximum inertia axis 6	0.1 kg.m²
Brakes	All a
WORK ENVELOPE	
Maximum reach between axis 1 and 5 (R. M)	600 mm, 23.6 in
Minimum reach between axis 1 and 5 (R.m1)	190 mm, 7.4 in
Minimum reach between axis 2 and 5 (R.m2)	189 mm, 7.4 in
Reach between axis 3 and 5 (R.b)	310 mm, 12.2 in
RANGE OF MOTION	
Axis 1 (A)	± 180°
Axis 2 (B)	± 127,5°
Axis 3 (C)	± 142,5°
Axis 4 (D)	± 270°
Axis 5 (E)	+132.5°/-122.5°
Axis 6 (F)	± 270°(1)

A4. FUNCIONS DEL SISTEMA DE VC

```
def take_picture():
    cap = cv2.VideoCapture(1)
    for i in range(40):
        ret, frame = cap.read()
        rgb = cv2.cvtColor(frame,
cv2.COLOR_BGR2BGRA)
```

```
    return rgb

def select_interest_area(img):
    im_size = img.shape
    img2 = img[int((im_size[0] / 2) -
15):int((im_size[0] / 2) + 105),
               int((im_size[1] / 2)) -
60:int((im_size[1] / 2) + 60), :]
    # Resize the new image
    img2 = cv2.resize(img2, (500, 500))
    return img2


def color_segment(img2):
    # convert to hsv colorspace
    hsv = cv2.cvtColor(img2, cv2.CO-
LOR_BGR2HSV)
    # lower bound and upper bound for
White color
    lower_bound = np.array([0,0,168])
    upper_bound = np.ar-
ray([172,111,255])
    # find the colors within the bounda-
ries
    mask = cv2.inRange(hsv, lower_bound,
upper_bound)
    return mask


def morphological_transform(img2, mask):
    # define kernel size
    kernel = np.ones((21, 21), np.uint8)
    # Remove unnecessary noise from mask
    mask = cv2.morphologyEx(mask,
cv2.MORPH_CLOSE, kernel)
    mask = cv2.morphologyEx(mask,
cv2.MORPH_OPEN, kernel)
    # Segment only the detected region
    segmented_img =
cv2.bitwise_and(img2, img2, mask=mask)
    return segmented_img

def detect_object():
    # Take picture
    img = take_picture()
    # Showing the output
    cv2.imshow("Image", img)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
    # Get area of interest
    img2 = select_interest_area(img)
    # Segment the desired color
    mask = color_segment(img2)
    # Apply the morphological transfor-
mation
    segmented_img = morphological_trans-
form(img2, mask)
    # threshold the grayscale image
    ret, thresh = cv2.threshold(segmen-
ted_img, 0, 255, 0)
    thresh = cv2.cvtColor(thresh,
cv2.COLOR_BGR2GRAY)
    # find outer contour
    cntrs = cv2.findContours(thresh,
cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
    cntrs = cntrs[0] if len(cntrs) == 2
else cntrs[1]
    # Check if object is detected
    if len(cntrs) == 0:
        # Manage case when object does
not need top be picked up
        print("No object detected")
        return "n", ["", "", ""]
    else:
        # get rotated rectangle from ou-
ter contour
        rotrect = cv2.minArea-
Rect(cntrs[0])
        box = cv2.boxPoints(rotrect)
        box = np.intp(box)
        # Compute the center of the ob-
ject
        sum_x = 0
        sum_y = 0
        for tuple in box:
            sum_x += int(tuple[0])
            sum_y += int(tuple[1])
        n_values = box.shape[0]
        center = [int(sum_x / n_values),
int(sum_y / n_values)]
        # draw rotated rectangle on copy
of img as result
        result = img2.copy()
        cv2.drawContours(result, [box],
0, (0, 0, 255), 2)
        result = cv2.circle(result, cen-
ter, radius=5, color=(0, 0, 255), thick-
ness=-1)

        # get angle from rotated rectan-
gle
        angle = rotrect[-1]
        # from https://www.pyima-
gesearch.com/2017/02/20/text-skew-cor-
rection-opencv-python/
        # the `cv2.minAreaRect` function
returns values in the
        # range [-90, 0); as the rectan-
gle rotates clockwise the
        # returned angle trends to 0 --
in this special case we
        # need to add 90 degrees to the
angle
        if angle < -45:
            angle = -(90 + angle)
        # otherwise, just take the in-
verse of the angle to make
        # it positive
        else:
            angle = -angl
        # Convert angle to positive va-
lue
```

```
        if angle < 0:
            angle = 90 + angle

        real_point_x = ((center[0] *
0.28) + 10)
        real_point_y = ((center[1] *
0.28) + 450)
        print(angle, "deg")
        print("Center of the object: ",
center)
        print("Real life point value X:
", real_point_x)
        print("Real life point value Y:
", real_point_y)
        # Round final object values
        real_point_x =
int(round(real_point_x, 0))
        real_point_y =
int(round(real_point_y, 0))
        angle = int(round(angle, 0))
        return "Y", [real_point_x,
real_point_y, angle]
```

A5. CODI FONT DEFINITIU DEL ROBOT TX-60

```
begin
 // Move arm to starting position
 movej(jHome,tToolVentosa,mNomSpeed)
 waitEndMove()
 movej(jCintaApproach,tToolVentosa,mLowspeed)
 waitEndMove()
 // Ready the sockets and variables
 clearBuffer(sSocket)
 clearBuffer(sSocketPLC)
 l_nNumero=0
 l_nPLC=0
 l_sValX=""
 l_sValY=""
 l_sValOrient=""
 // Ready the user page
 userPage()
 cls()
 gotoxy(12,0)
 put("Main Paletizado")
 gotoxy(0,2)
 put("Dimensiones del paletizado (x,y,z):")
 gotoxy(0,3)
 put(nXMax)
 put(" x ")
 put(nYMax)
 put(" x ")
 put(nZMax)


 // Main loop
 for nZCount=0 to nZMax-1
  for nXCount=0 to nXMax-1
   for nYCount=0 to nYMax-1
    do
     // Await for the PLC signal
     do
      sioGet(sSocketPLC,l_nPLC)
     until l_nPLC==100
     //sioGet(sSocketPLC, l_nPLC)
     // Take photo and look for object
     // 70(F) = Take photo
     l_nNumero=70
     wait(sioSet(sSocket,l_nNumero)==1)
     // 89(Y) = Object, 110(n) = No object
     wait(sioGet(sSocket,l_nNumero)==1)
     // In none object detected, send move signal to PLC
     if l_nNumero==110
      clearBuffer(sSocketPLC)
      wait(sioSet(sSocketPLC,l_nNumero)==1)
      delay(2)
      l_nPLC=0
      wait(sioSet(sSocketPLC,l_nPLC)==1)
      // Show message to user
      cls()
      gotoxy(3,1)
      put("Ningun objeto deseado detectado")
      gotoxy(0,3)
      put("Pasando al siguiente objeto...")
     endIf
    until l_nNumero==89
    // Get the object coordinates and rotation
    l_sValX=""
    l_sValY=""
    l_sValOrient=""
    // X value
    do
     sioGet(sSocket,nObjectX)
     l_sValX=l_sValX+chr(nObjectX)
    until nObjectX==13
    // Y value
    do
     sioGet(sSocket,nObjectY)
     l_sValY=l_sValY+chr(nObjectY)
    until nObjectY==13
    // Orientation value
    do
     sioGet(sSocket,nObjectOrient)
     l_sValOrient=l_sValOrient+chr(nObjectOrient)
    until nObjectOrient==13
    // Convert string to numerical value
    nObjectX=0
    nObjectY=0
    nObjectOrient=0
    // X value (negative)
    for nNumCount=0 to len(l_sValX)-2
     nPowerIndex=(len(l_sValX)-2)-nNumCount
     nTempValue=asc(l_sValX,nNumCount)-48
     nTempValue=nTempValue*nPower10[nPowerIndex]
     nObjectX=nObjectX-nTempValue
    endFor
    // Y value (negative)
    for nNumCount=0 to len(l_sValY)-2
     nPowerIndex=(len(l_sValY)-2)-nNumCount
```

```
        nTempValue=asc(l_sValY,nNumCount)-48
        nTempValue=nTempValue*nPower10[nPowerIndex]
        nObjectY=nObjectY-nTempValue
      endFor
      // Orient value
      for nNumCount=0 to len(l_sValOrient)-2
        nPowerIndex=(len(l_sValOrient)-2)-nNumCount
        nTempValue=asc(l_sValOrient,nNumCount)-48
        nTempValue=nTempValue*nPower10[nPowerIndex]
        nObjectOrient=nObjectOrient+nTempValue
      endFor
      // Show the user the recieved data
      cls()
      gotoxy(10,0)
      put("Objeto detectado")
      gotoxy(0,2)
      put("Pos X: ")
      put(nObjectX)
      gotoxy(0,3)
      put("Pos Y: ")
      put(nObjectY)
      gotoxy(0,4)
      put("Orient: ")
      put(nObjectOrient)
      // Pick up the object and correct the rotation
      movel(pCintaApproach,tToolVentosa,mLowspeed)
      waitEndMove()
      open(tToolVentosa)
      movel(appro(pPointCinta,{nObjectX,nObje-
ctY,0,0,0,0}),tToolVentosa,mLowspeed)
      waitEndMove()
      movel(appro(pCintaApproach,{0,0,0,0,0,nObjectO-
rient}),tToolVentosa,mLowspeed)
      waitEndMove()
      // Move the object to the pallet
      movel(appro(pHeightApproach,{0,0,-
80*nZCount,0,0,nObjectOrient}),tToolVen-
tosa,mLowspeed)
      waitEndMove()
      movel(appro(pPaletApproach,{55*nXCount,-55*nY-
Count+(55*abs(nYMax/2)),-70*nZCount,0,0,nObjectO-
rient}),tToolVentosa,mLowspeed)
      waitEndMove()
      movel(appro(pPaletPosition,{55*nXCount,-55*nY-
Count+(55*abs(nYMax/2)),-70*nZCount,0,0,nObjectO-
rient}),tToolVentosa,mLowspeed)
      waitEndMove()
      close(tToolVentosa)
      delay(0.5)
      // Return to the waiting position
      movel(appro(pPaletApproach,{55*nXCount,-55*nY-
Count+(55*abs(nYMax/2)),-70*nZCount,0,0,nObjectO-
rient}),tToolVentosa,mLowspeed)
      waitEndMove()
      movel(appro(pHeightApproach,{0,0,-
80*nZCount,0,0,nObjectOrient}),tToolVen-
tosa,mLowspeed)
      waitEndMove()
      gotoxy(0,6)
      put("Esperando al siguiente objeto...")
    endFor
  endFor
endFor
// Program end message
cls()
gotoxy(9,0)
put("Paletizado finalizado")
// Closing socket signal
// 88(X)
l_nNumero=88
wait(sioSet(sSocket,l_nNumero)==1)
end
```