UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCOLA TÈCNICA SUPERIOR ENGINYERS INDUSTRIALS VALÈNCIA

TREBALL FINAL DE GRAU EN ENGINYERIA EN TECNOLOGIES INDUSTRIALS

DESENVOLUPAMENT D'APLICACIONS DE CONTROL DE POSICIÓ I FORÇA PER A UN ROBOT INDUSTRIAL

AUTORA: ANA RODRÍGUEZ ESTEVE

TUTORA: MARINA VALLÉS MIQUEL

COTUTOR: ÁNGEL VALERA FERNÁNDEZ

Curs Acadèmic: 2015-16

Aquest Treball de final de Grau ha sigut possible gràcies al recolzament de la meua família que ha cregut en mi en tot moment.

Gràcies als meus companys, que després de quatre anys d'aprendre junts ja són amics.

I no puc oblidar-me de la persona que ha aguantat els meus moments de nervis i ha aprés com funciona un robot per poder-me ajudar, gràcies Javi.

RESUM DOCUMENTAL

El següent document va descriure com s'ha desenvolupat una aplicació per controlar el moviment d'un robot de configuració cartesiana, per satisfer una necessitat molt present a la industria actual, com puga ser la classificació de productes.

L'aplicació implementada farà ús d'un sensor de força per saber quina alçada fa el producte a classificar i així portar-lo a la posició establerta.

Podrem veure el plantejament del problema, i com s'implementarà la solució per a aquest. El document, a més, inclou documentació històrica sobre l'evolució de la robòtica, el control de robots i els sensors, així com la descripció detallada dels elements utilitzats per al desenvolupament de la solució. Finalment s'analitzen els resultats obtinguts, fent una proposta de les millores que es podrien dur a terme.

RESUMEN DOCUMENTAL

El siguiente documento va a describir como se ha desarrollado una aplicación para controlar el movimiento de un robot de configuración cartesiana, para satisfacer una necesidad muy presente en la industria actual como pueda ser la clasificación de productos.

La aplicación implementada hará uso de un sensor de fuerza para saber cuál es el tamaño del producto que ha de clasificar y así poderlo llevar a la posición correcta.

Podremos ver el planteamiento del problema y como se implementará la solución de este, El documento además incluirá documentación histórica sobre la evolución de la robótica, el control de robots y los sensores, así como la descripción detallada de los elementos utilizados para el desarrollo de la solución. Finalmente de analizan los resultados obtenidos haciendo propuestas de las mejoras que se podrían llevar a término.

ABSTRACT

The following document is going to describe since an application has developed to control the movement of a robot of Cartesian configuration, to satisfy a very present need in the current industry since could be the classification of products.

The implemented application will use a sensor of force to know which is the size of the product that has to classify and this way to be able to take it to the correct position.

We will be able to see the exposition of the problem and since the solution will be implemented of this one, the document in addition will include historical documentation on the evolution of the robotics, the control of robots and the sensors, as well as the detailed description of the elements used for the development of the solution. Finally we analyze the results obtained, and do offers of the improvements that might be applied.

ÍNDEX

DOCUMENTS CONTINGUTS EN EL TFG

1. **Memòria descriptiva.**
2. **Pressupost.**
3. **Annexes.**

ÍNDEX MEMÒRIA DESCRIPTIVA

ÍNDEX PRESSUPOST

ÍNDEX ANNEXES

ÍNDEX D'IMATGES

ÍNDEX DE TAULES

MEMÒRIA DESCRIPTIVA

CAPÍTOL 1. INTRODUCCIÓ

1. INTRODUCCIÓ

1.1 DESCRIPCIÓ DEL PROJECTE

Aquest projecte de final de carrera es basa en la implementació d'una aplicació que permeta la comunicació d'un PC, un sensor de força JR3 i un robot manipulador de tipus cartesià.

En concret aquesta aplicació ha de permetre que, introduint l'usuari la posició de l'objecte a manipular, molt fàcil de obtenir manualment, el robot siga capaç de, classificar l'objecte segons la seua mida, sense tindre unes coordenades prèvies sobre aquesta programades. Per a aconseguir açò el robot es basarà en les dades que li proporcione el sensor de força. Els objectes seran dipositats finalment en columna, ja que d'aquesta forma es pot donar un ús més interessant al sensor de força.

Per a la realització d'aquesta comunicació entre el PC i el robot manipulador s'han utilitzat targetes d'adquisició de dades, aquestes es comuniquen amb el computador per facilitar el coneixement de la posició i velocitat de cada eix del robot. Aquestes targetes d'adquisició de dades fan ús dels encoders dels que disposa el robot industrial emprat per a obtenir aquesta informació.

Aquesta aplicació s'ha desenvolupat mitjançant llenguatge C en Microsoft Visual Studio 2005. Encara que és una versió obsoleta és la versió del programa de la que disposava el PC connectat al robot cartesià.

1.2 OBJECTIUS

L'objectiu principal d'aquest Projecte de Final de Grau és el disseny d'una aplicació la qual proporcione al robot porticat comunicació amb el PC per poder generar una ruta que, basada en els valors de força proporcionats per un sensor, classifique i organitze l'objecte desitjat segons la seua grandaria, en lloc de programar unes coordenades fixes per a què el robot les execute.

Per a aconseguir aquest objectiu principal podem subdividir aquest objectiu en els següents subojectius:

- **Comunicació del robot amb el PC** mitjançant targetes d'adquisició de dades, que permetran el coneixement dels diferents paràmetres del robot.
- **Incorporació del sensor de força JR3** a l'extrem de l'eix Z del robot, que és aquest el que estarà en contacte amb els objectes a manipular, per tant, el que necessita emprar un sensor de força per no aplicar una pressió excessiva.
- **Disseny del software Controlador** del robot encarregat de generar el moviment dels eixos del robot, controlant l'error en la velocitat i posició fent ús de la informació aportada pels sensors.
- Generació del **Software encarregat de la Modificació de la Trajectòria**. Aquest s'encarregarà d'anar generant la referència que haurà de seguir el controlador del robot.
- Implementació del programa capaç **d'interpretar les dades enviades pel sensor de força** i modificar els moviments i les trajectòries del robot fent ús d'aquestes.

1.3 ESTRUCTURA DEL PROJECTE

El document recollirà 8 capítols que es descriuen a continuació.

El primer Capítol, que és l'actual, es realitza una descripció dels temes a tractar en aquest treball de fina de grau, així com una breu descripció de les parts que el componen.

El Capítol 2 es fa una introducció històrica i teòrica sobre els elements del que tracta aquest projecte per poder entendre millor el seu funcionament. S'expliquen a més les característiques del robot porticat utilitzat i del sensor de força JR3.

Durant el Capítol 3 es descriu amb deteniment les targetes d'adquisició de dades utilitzades, així com el software del qual s'ha fet ús per a la implementació del codi, i del sensor de força.

En el Capítol 4 es desenvolupa la solució al problema plantejat. Es descriu la implementació de l'aplicació amb codi C fent ús del programa Visual Studio 2005. Es descriuen les parts més importants del codi, les que han estat dissenyades des de zero.

El Capítol 5 conté el pressupost, amb els quadres de preu, les unitats d'obra i el pressupost base de licitació final del projecte.

El Capítol 6 conté les conclusions del treball, exposant els problemes que s'han trobat durant la realització, així com les possibles modificacions que es podrien realitzar en el sistema per aconseguir un resultat més òptim i precís.

En el Capítol 7 es troba la bibliografia, on es poden veure els documents consultats al llarg de la realització del projecte.

El Capítol 8 recull els annexes, on podem trobar el codi complet de l'aplicació.

CAPÍTOL 2. FONAMENTS DEL PROJECTE

2. FONAMENTS DEL PROJECTE.

2.1 ROBOT MANIPULADOR.

2.1.1 DEFINICIÓ DE ROBOT INDUSTRIAL.

No hi ha una clara i mundialment acceptada definició de robot industrial, pel que es poden trobar diferencies a l'hora de definir el que és un robot industrial en el mercat occidental i en el mercat oriental. Per a aquest últim, un robot industrial és qualsevol dispositiu mecànic dotat d'articulacions mòbils destinat a la manipulació. La definició occidental és més restrictiva, i té en compte el control del robot. Així i tot no hi ha una definició concreta, i cada organització ha adoptat una pareguda però a la vegada diferent.

L'Associació de Industries de Robòtica (RIA, *Robotic Industry Association*) defineix el concepte de robot industrial com:

"Manipulador multifuncional reprogramable, capaç de moure materials, peces, ferramentes o dispositius especials segons trajectòries variables programades per a realitzar diverses tasques"

L'Organització Industrial d'Estàndards (ISO) ho defineix lleugerament diferent:

"Manipulador multifuncional reprogramable amb diversos graus de llibertat, capaç de manipular materials, peces, ferramentes o dispositius especials segons trajectòries variables programades per a realitzar tasques diverses"

IMATGE 2.1 DIBUIX BRAÇ ROBÒTIC

Per últim la Federació Internacional de Robòtica (IFR, *International Federation of Robotics)* fa una distinció entre robot industrial de manipulació i altres robots.

"Per robot industrial de manipulació s'entén una màquina de manipulació automàtica, reprogramable y multifunció amb tres o més eixos que poden posicionar i orientar matèries, peces, ferramentes o dispositius especials per a l'execució de diversos treballs en les diferents etapes de la producció industrial, siga una posició fixa o en moviment"

En totes les definicions anteriors s'accepta com a **robot industrial** un braç mecànic amb capacitat de manipulació i que fa ús d'un control més o menys complex.

Però un sistema robotitzat és, en canvi, un concepte molt més ampli. Engloba tots aquells dispositius que realitzen de forma automàtica tasques en substitució d'un ser humà que poden incloure un o més robots.

2.1.2 HISTÒRIA DE LA ROBÒTICA

En l'actualitat es porta a terme una producció massiva en l'industria, una producció que engloba quantitats mil·lenàries de productes fets en un dia. Per a aquesta s'empren sistemes robotitzats que treballen amb alta autonomia, amb gran flexibilitat d'horaris i una altíssima precisió que aporta una gran qualitat als productes fabricats. Aquests avantatges que ens aporten els sistemes robotitzats no serien executables amb la mà d'obra convencional, ja que encaririen el preu final del producte.

Però la robòtica que avui coneguem té el seu origen milers d'anys enrere. Durant molt de temps les persones hem construït màquines que imitaven les parts o moviments del cos humà. Fins i tot els grecs construïen estàtues de Deus amb sistemes hidràulics per a què es moguessen soles per fascinar als creients, com puga ser el cas de l'estàtua Memnon d'Etiòpia que emetia sorolls quan els rajos del sol incidien en ella.

IMATGE 2.2 ESTATUA DE MEMNON EN L'ACTUALITAT

A finals del segle XVIII i principis del XIX, durant la Revolució Industrial, es van desenvolupar una gran quantitat d'enginys mecànics utilitzats en l'industria tèxtil. Més tard s'incorporen els automatismes també a la industria minera i metal·lúrgica.

Però sense lloc a dubtes, l'autòmat que va ser més important durant la Revolució industrial va ser el de les vàlvules de la màquina de vapor atmosfèrica, creada per Thomas Newcomen.

La paraula robot no va aparèixer fins molts anys després. El que abans s'anomenava autòmat va passar a nomenar-se robot gràcies a un dramaturg txec, Karel Capek. Aquest va utilitzar la paraula *robota* per primera vegada l'any 1920. Aquesta significa treball, i en la major part de les llengües escandinaves significa figuradament "treball dur".

Encara que es a l'escriptor Isaac Asimov al qual se li atribueix el terme robòtica. A més, aquest escriptor de ciència ficció també va redactar el que avui en dia és conegut com a les tres lleis de la robòtica, que són:

1. Un robot no pot actuar contra un ser humà, o per mitjà de la no-acció, que un ser humà sofresca danys.

2. Un robot ha de seguir les ordres donades pels sers humans, excepte si per seguir-les entra en conflicte amb la primera llei.
3. Un robot ha de protegir la seua pròpia existència, a no ser que estiga en conflicte amb les dues primeres lleis.

Actualment la Real Acadèmia Espanyola de la Llengua defineix la paraula robot com: "*Màquina o ingeni electrònic programable, capaç de manipular objectes i realitzar operacions abans sols reservades per a les persones".* Per altre costat, el Robot Institute of America defineix un robot com a "*Un manipulador reprogramable i funcional dissenyat per a moure materials, peces o dispositius especialitzats, a través de moviments programats variables per a la realització de una diversitat de feines".*

Amb aquesta definició, un robot ha de tindre 'intel·ligència', que es deu normalment als algoritmes de computador associats al seu sistema de control i sensorial. A causa de la política de fabricants de I+D, en quasi la totalitat del casos de robots industrials estan basats en una arquitectura tancada, és a dir, el sistema operatiu utilitzat, el sistema de comunicació i el llenguatge de programació del robot són propis del sistema. Aquest fet té coma resultat una dependència al fabricant del robot. A més, l'intent de modificar o canviar l'algoritme de control resulta pràcticament impossible.

2.1.3 HISTÒRIA DEL CONTROL DE ROBOTS.

Per altre costat, per a la realització del control d'aquests sistemes robotitzats s'utilitzen els denominats algoritmes de control. Un algoritme de control descriu formalment l'estratègia de control. L'algoritme a partir d'unes variables confiables per al sistema, i a partir d'una seqüència finita d'instruccions, genera les variables d'eixida que es desitgen obtenir.

Els algoritmes de control més utilitzats actualment es basen en aproximacions lineals a la dinàmica del manipulador. Encara que és important considerar la dinàmica no lineal completa del manipulador quan s'està sintetitzant l'algoritme de control.

Els robots que utilitzen aquestes estratègies de control, és a dir, les lineals, són els anomenats robots de **primera generació**, aquells que encara són poc capaços d'adaptar-se a l'entorn. Per al bon funcionament d'aquests són necessaris dispositius addicionals, que faran que el preu d'aquest s'eleve.

S'està desenvolupant, a més, el control òptim o control no-lineal, basat en les equacions diferencials del model, que intenta conéixer el model matemàtic exacte del sistema, però en moltes ocasions açò ho impedeix la incertesa. A més tots els models físics a la llarga sofreixen desgasts provocats per l'envelliment o pas del temps, que fan que varien els seus paràmetres característics. Tot açò fa que trobar la solució d'aquest problema de control òptim siga una tasca complexa i llarga, i que a més en moltes ocasions ens siga impossible de determinar.

En la dècada de 1960 i 1970 va aparéixer el concepte de robustesa d'un sistema de control adaptatiu, és a dir, els paràmetres no es deuen veure afectats per la possible variació dels paràmetres característics del sistema, ja siga pel pas del temps o per envelliment, i tampoc s'han de veure afectats per les pertorbacions. En aquestes dècades no existien mètodes per a determinar el sistema robust adequat per a cada sistema, però de manera experimental es comprova que el control PID és un control robust, a més de senzill, davant les variacions dels paràmetres i les pertorbacions.

Per a açò cal que els robots tinguen sensors acoblats, per a rebre certa informació del seu entorn mitjançant els sensors i a més posseeixen un control de la seua execució. Aquests són nomenats robots de **segona generació.**

Però el clàssic bucle de control amb realimentació de guany constant tampoc pareix la millor opció per al concepte de robustesa d'un sistema de control. És necessària una nova filosofia de control, capaç de modificar els seus paràmetres de la forma més adequada per compensar les variacions en el sistema.

Entre les tècniques més utilitzades per a la implementació d'aquest tipus de controladors destaquem quatre grups: tècniques basades en el concepte de passivitat, tècniques basades en la dinàmica inversa, tècniques adaptatives i finalment controladores-observadores.

- **Controladors basats en la idea de passivitat.** Aquests afronten el problema de control de robots explotant l'estructura física del robot. La idea d'aquesta filosofia de disseny de controladors és modificar i modelar l'energia natural del sistema del robot de manera que l'objectiu de control siga aconseguida.

- **Controladors basats en la dinàmica inversa.** Les estratègies de control basades en la dinàmica inversa es poden incloure en la tècnica de control de la linealització per realimentació de l'entrada. Per a què un sistema puga ser linealitzable (com en el cas dels sistemes robotitzats) es deu complir que existeixen una transformació de l'espai de l'estat de realimentació estàtic regular que transforme el sistema no lineal en un lineal.

- **Controladors adaptatius.** Aquests apareixen pel fet que en la pràctica es donen una quantitat considerable d'incerteses en els models dinàmics dels sistemes robotitzats. Aquests no necessiten un coneixement tan exacte del sistema dinàmic a controlar. Apareixen en la realitat molts sistemes que tenen paràmetres desconeguts que varien de manera lenta, o que són constants. Són aquests els que es poden controlar amb el controlador adaptatiu. La idea bàsica d'aquest és estimar els paràmetres del controlador corresponent basant-se en les senyals del sistema de mesures, i utilitzant paràmetres estimats en el càlcul de l'entrada de control.

- **Controlador-observador.** Els controladors-observadors són vàlids quan es disposa únicament del coneixement parcial de les variables d'estat, considerant la posició. Generalment, a l'hora d'establir un control de seguiment de trajectòria de qualsevol procés necessitem tindre un coneixement clar de les variables d'estat. Però açò no

sempre és possible, donat que no sempre es disposa dels sensors necessaris. En aquestes situacions, si volem implementar un control, hem de construir un sistema auxiliar dinàmic que ens facilite les variables del sistema no accessibles, establint amb aquestes el control.

Aquestes no apleguen al seu punt àlgid fins a la dècada dels 90. En aquesta dècada es va desenvolupar i ampliar la teoria de control no lineal basada en tècniques de control adaptatiu.

Apareixen així els robots de **tercera generació**, aquestos no sols interactuen amb l'entorn, sinó que s'adapten a ell i poden generar els seus propis plans d'acció.

La ferramenta que sense lloc a dubtes ha ajudat a la seua investigació són els microprocessadors. Els microprocessadors són cada vegada més potents i més barats, el que permet la seua utilització en quasi totes les àrees científiques.

Des del punt de vista de l'aplicació tecnològica, aquestes tècniques adaptatives o de control no lineal són complexes i proporcionen lleis de control d'alt cost computacional en la seua implementació. Algunes s'han implementat des del punt de vista experimental, però la seua aplicació industrial encara no està explotada.

2.1.4 CLASSIFICACIÓ DELS ROBOTS.

Respecte a l'anatomia dels robots no tots els que podem trobar hui en dia són del mateix tipus. Per una part tenim els robots mòbils, aquests disposen de sistemes per a la transmissió de moviment, el que permet el seu desplaçament.

Aquests troben el seu lloc a l'industria per al transport de materials en, per exemple, terrenys abruptes o perillosos per a les persones. En la imatge següent podem veure un robot mòbil de la marca lego, exactament el model lego NXT, del que hem fet ús a l'assignatura optativa de quart del grau d'Enginyeria de Tecnologies Industrials nomenada Laboratori d'Automatització i Control.

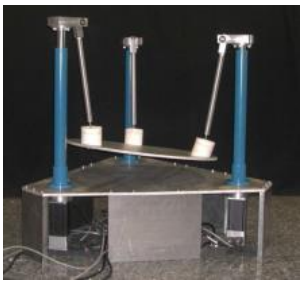
Els robots mòbils són un focus d'investigació important i quasi cada universitat té un laboratori centrat en la investigació d'aquests.

IMATGE 2.3 ROBOT LEGO NXT

Per altre costat tenim els robots manipuladors o robots industrials, que poden tindre formes antropomòrfiques, com puguen ser els robots de braç, o poden ser cartesians. Aquestos estan ancorats en un dels seus extrems, i troben el seu lloc en la industria per a sistemes estructurats.

IMATGE 2.4 ROBOT DE BRAÇ

IMATGE 2.6 ROBOT PORTICAT



IMATGE 2.5 ROBOT 3 EIXOS

2.1.5 FUTUR DE LA ROBÒTICA.

La robòtica entra en la era moderna amb la mateixa força que ho va fer la màquina de vapor en l'època preindustrial. Les instal·lacions industrials del futur seran modulars i molt més flexibles que les fàbriques actuals. Açò serà possible gràcies a microprocessadors, unitats d'emmagatzemament, sensors i transmissors integrats en quasi totes les màquines, productes i materials.

Encara que ja existeixen alguns elements de fàbrica "intel·ligent", els experts creuen que encara queden alguns anys per a aconseguir l'automatització quasi completa, que farà que els operaris tinguen un paper diferent dins del procés, i que sols s'encarreguen de supervisar i controlar els autòmats.

A més la intel·ligència artificial és un tema d'investigació actual, cada vegada més avançat. Pot tindre un gran futur en camps com la medicina, encara que també diuen els experts que mai substituirà per complet al ésser humà, ja que no podrà tindre la mateixa creativitat ni la mateixa

2.2 SENSORS

2.2.1 DEFINICIÓ DE SENSOR.

Un sensor és un dispositiu que està capacitat per a detectar accions o estímuls externs i respondre en conseqüència. Un exemple molt clar d'aquest es aquell que detecta quan el nivell de combustible d'un vehicle és baix i emet una llum per a avisar al conductor.

Altre tipus de sensor és el que es col·loca en la porta dels centres comercials, o locals d'oci, on, quan detecta la presència d'una persona s'emet un senyal elèctrica que fa que aquestes s'obrin. La utilització d'aquestes és per a evitar la presencià d'una persona encarregada d'obrir, com succeïa en el passat, a més que és una forma més eficient d'obrir les portes i s'evita la formació de cues o taps.

El termòmetre també és un sensor ja que aprofita la capacitat del mercuri per a reaccionar davant la temperatura i d'aquesta manera detectar si una persona té febre.

Els sensors són, en definitiva, dispositius que permetran obtindre informació del entorn i interactuar amb ella.

2.2.2 HISTORIA DELS SENSORS.

Els sensors han existit des de sempre, ja que l'ésser humà els té inclosos en el seu cos. I aquests no són sols d'un tipus, hi ha de diferents tipus. A l'igual que les persones fem ús d'aquests per a obtenir informació del nostre voltant, els robots requereixen els sensors per a la interacció amb el medi.

IMATGE 2.7 ICONES 5 SENTITS

Actualment les àrees d'aplicació dels sensors són: La industria del automòbil, la robòtica, la industria aeroespacial, la medicina, etc.

2.2.3 CARACTERÍSTIQUES DELS SENSORS.

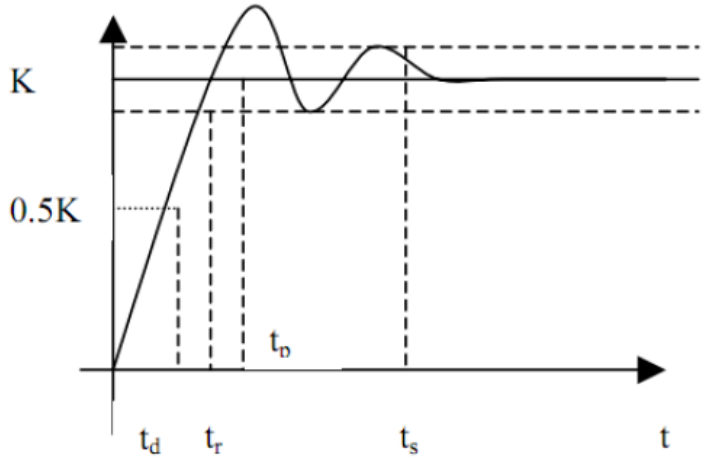
2.2.3.1 Característiques estàtiques de un sensor.

Aquestes són les característiques que defineixen el comportament en règim permanent del sensor.

- Rang: Valors màxims i mínims per a les variables d'entrada i eixida d'un sensor.
- Repetibilitat: La capacitat de reproduir una lectura amb una precisió donada.
- Exactitud: La desviació de la lectura del sensor d'un sistema respecte a una entrada coneguda. El error més gran esperant entre els senyals de mesura i les ideals.
- Resolució: La mesura més menuda que es pot detectar.
- Sensibilitat: La raó de canvis en l'eixida respecte a l'entrada.
- Soroll: Es tracta del senyal no desitjat que es mescla amb l'útil.

2.2.3.2 Característiques dinàmiques de un sensor.

Les característiques dinàmiques defineixen el comportament del règim transitori del sensor:



IMATGE 2.8 CARACTERÍSTIQUES DINÀMIQUES

D'aquesta imatge es poden definir les característiques dinàmiques més importants:

- t_d o Temps de retard: És el temps que tarda l'eixida del sensor en aplegar al 50% del seu valor final
- t_r o Temps de pujada: És el temps que tarda l'eixida del sensor en aplegar al seu valor final, aquest està molt relacionat amb la velocitat a la que respon davant d'una entrada.
- t_p o Temps pic: El temps que tarda l'eixida del sensor en aplegar al seu pic màxim.
- t_s o Temps d'establiment: Temps que tarda l'eixida del sensor en entrar en la banda del % del valor final i no torna a eixir de ella.

2.2.3.3 Classificació dels sensors utilitzats en robots.

Com s'ha mencionat abans, el robot, a l'igual que un ésser humà, per a poder interactuar de manera correcta amb el seu entorn ha de fer ús dels sensors. Aquests asseguraran que el robot faça la seua tasca amb la precisió adequada. La funció d'aquests sensors es pot dividir en dos:

- Sensors interns: Els sensors interns estan integrats en l'estructura del robot i donen la informació de l'estat d'aquest (posició, velocitat i acceleració de cada articulació).
- Sensors externs: Els sensors externs donen la informació de l'entorn del robot (alcans, proximitat, força, contacte...), aquest serveixen per a la manipulació d'objectes.

2.2.3.4 Sensors interns

Com s'ha explicat abans aquestos sensors seran els que proporcionen la informació sobre la posició, velocitat i acceleració del robot.

2.2.3.4.1 Sensors de posició

Es poden definir dos tipus de sensors de posició, aquests seran els encarregats de definir la posició del robot o de les seues articulacions.

- Analògics: L'eixida d'aquests serà variable dins d'un determinant rang. Alguns exemples d'aquest tipus de sensors de posició són els potenciòmetres. Els potenciòmetres s'utilitzen per a mesurar desplaçaments lineals o angulars, aquests poden disposar d'un sistema de calibratge per obtenir una relació entre la mesura desplaçada i el potencial obtingut.
- Digitals: són aquells que tenen com a eixida un senyal de tipus discret. És a dir, encara que la seua eixida també varia dins d'un rang determinat ho fa a xicotetes passes preestablertes. Els **encoders** serien un exemple d'aquest tipus de sensors, i són els que s'han utilitzat per al desenvolupament d'aquest projecte.

2.2.3.4.2 Sensors de velocitat

Tindre un coneixement de la velocitat del sistema robotitzat és necessària per a millorar el comportament dinàmic del robot. La informació de la velocitat de cada actuador ve normalment realimentada amb un bucle de control implementat en el propi accionador. Encara que hi ha vegades que el sistema de control de robot exigeix que la velocitat de gir de cada actuador siga portada a la unitat de control del robot.

Per a aquests casos s'utilitza un tacogenerador, donat que el bucle de control de la velocitat és analògic, aquest proporcionarà una tensió proporcional a la velocitat de gir de l'eix en el que estiga instal·lat.

Per a generar el senyal elèctric a partir del gir que proporcionarà posteriorment la velocitat del eix, una espira situada dins d'un camp magnètic fixe s'acobla al motor o a l'eix que es va a mesurar. En girar el motor l'espira girarà en l'interior del camp magnètic, fent així que es genere un corrent elèctric.

2.2.3.4.3 Sensors d'acceleració

Els acceleròmetres s'encarregaran de transformar la magnitud física de l'acceleració en altra magnitud elèctrica que serà la que posteriorment empren els equips d'automatització. Hi ha una gran varietat d'acceleròmetres, amb diferents rangs de mesura i de precisió.

2.2.3.5 Sensors externs

Els sensors externs poden classificar-se en sensors de contacte o no contacte. Aquestos són els que permeten al sistema robotitzat tindre coneixement del medi que els envolta, facilitant d'aquesta manera la implementació de programes més complexos, i donant la possibilitat que el robot realitze tasques amb major precisió. Seguint la classificació entre sensors de contacte o no contacte, els primers són aquells que responen al contacte físic, tant com tacte, torsió o lliscament. Els de no contacte es basen en la detecció de la variació d'un senyal electromagnètic o acústic. Aquests s'utilitzen per mesurar la proximitat o les propietats visuals d'un objecte.

Els sensors externs de no contacte proporcionen informació de guiat aproximat per a un manipulador, mentre que els sensors externs de contacte estan més associats als moments finals en el que el robot ja està en contacte amb l'objecte i ha d'agafar-lo i transportar-lo.

2.2.3.5.1 Sensors contacte

Aquests s'empren en robòtica per a obtenir informació associada amb el contacte de l'extrem del manipulador i els objectes del seu entorn de treball. La informació es pot utilitzar per al reconeixement de l'objecte, així com per a controlar la força exercida sobre un objecte donat. Aquests es poden subdividir en dues categories principals:

- Sensors binaris.

  Aquests són dispositius de contacte com micro-interruptors. Aquests s'utilitzaran per a enviar un senyal que confirme la presència de l'objecte en la posició desitjada. La seua eixida serà zero o u, u en cas que el sensor estiga activat.

- Sensors analògics.

  Els sensors de contacte analògics seran aquells que envien un senyal proporcional a la força local. Aquest és el cas del sensor utilitzat per a la realització d'aquest projecte.

2.3 PARTS D'UN ROBOT

En aquest punt es va a explicar els elements bàsics que conformen un robot, de manera que en els posteriors apartats es puga entendre la funció de cadascuna d'aquestes.

Es va a definir la estructura de forma general, i per a açò ho podem dividir en tres parts.

UNITATS DE PROGRAMACIÓ

Aquesta és la que s'utilitza per a programar i operar amb el robot. A través de la unitat de programació, els resultats dels programes executats i els comandaments poden ser analitzats. Normalment sol consistir en un teclat industrial, una pantalla i un control remot.

SISTEMA DE CONTROL

El sistema de control rep les ordres i instruccions de la unitat de programació. Després els interpreta. La seua tasca serà generar les accions de control adequades, és a dir, generar un senyal de referència que siga la que s'encarregue de dirigir els actuadors dels robots, aquesta referència es compararà amb la informació rebuda pels sensors, i el sistema de control ha d'intentar reduir la diferència entre la referència i l'eixida del robot. A més el sistema de control informarà l'usuari sobre el resultat de les accions de control a través de la unitat de programació.

Per a aquest es necessari disposar de:

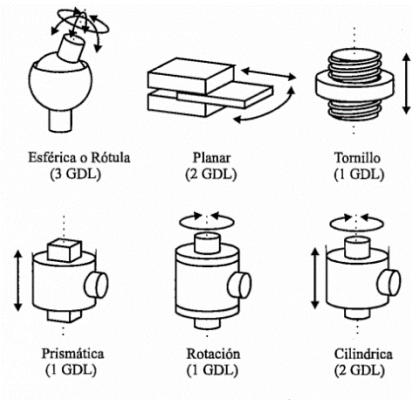
- Targetes d'adquisició de dades que ens permeten extraure informació dels sensors
- Convertidor D/A per proporcionar les accions de control.

MECÀNICA DEL ROBOT

La mecànica del robot és la que s'encarrega de convertir les accions de control en moviments, amb la finalitat d'interactuar amb l'entorn i realitzar una activitat determinada. El moviment del robot es du a cap amb el moviment de les seues articulacions que estarà produït pels actuadors.

Podem distingir entre tres tipus d'actuadors segons la font d'energia que els impulse. Aquests seran els actuadors pneumàtics, els actuadors hidràulics, i els motors elèctrics. En l'actualitat el que més s'utilitza és el motor elèctric, ja que té un major rendiment i una millor precisió.

També hi ha diferents tipus d'articulacions, que són l'element que proporciona moviment al robot. El moviment d'aquestes pot ser de rotació, de translació o combinar aquestes dues. Les més comuns són la cilíndrica, la de revolució, la prismàtica, i l'esfèrica. Encara que a hui en dia la major part dels robots industrials incloguen sols les de revolució i les prismàtiques, donat que són més econòmiques, robustes i senzilles.



IMATGE 2.9 ARTICULACIONS D'UN ROBOT

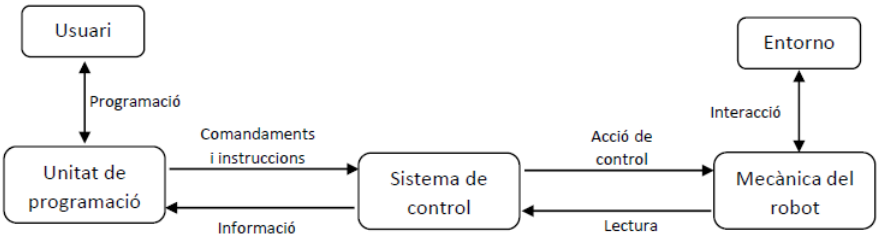
A més, hi ha components mecànics que s'encarreguen d'informar periòdicament al sistema de control de l'estat del robot. Aquests són tant els sensors, com els potenciòmetres o els encoders, que permetran conéixer a cada instant la posició del robot i l'estat.

Els **encoders** són un dispositiu mecànic que utilitza sensors òptics per traduir una sèrie de polsos que es poden traduir en moviment, posició o velocitat. Hi ha dos tipus bàsics d'encoders, els incrementals i els absoluts.

El encoder incremental funciona de la següent manera. Un disc de molt poc gruix està connectat a un dels eixos del motor, aquell del que desitgem conéixer la posició o velocitat. Un eix de llum estacionar enfoca contínuament aquest disc. A l'altra banda del disc hi ha un fotodetector que emetrà un pols cada vegada que detecte llum. Per poder conéixer la direcció del moviment es disposaran de dos encoders desfasats.

Els encoders absoluts estan dissenyats de tal manera que saben en cada moment en quina posició es troben, ja que generen un codi binari únic per a cadascuna de les posicions. Aquests s'utilitzen per a aplicacions en les quals la informació de la posició és més important que un canvi en aquesta.

En la següent il·lustració es pot veure un esquema resum de com és un robot, de manera genèrica. Per al treball que estem exposant en aquest document el sistema de control i la unitat de programació estan a **l'ordinador,** pel que es disposarà d'un sistema de control obert que fa que siga possible modificar les estratègies de control.



IMATGE 2.10 ESTRUCTURA D'UN ROBOT

Per a la realització d'aquest treball utilitzarem un robot porticat. És a dir, un robot de coordenades cartesianes limitat en el seu eix horitzontal i recolzat en els seus extrems. Els seus tres eixos de control seran lineals i formaran angles rectes els uns amb els altres. Aquesta configuració simplificarà les equacions de control dels braços robot. L'aplicació més estesa per a aquest tipus de robots és la de màquina de control numèric, com per a fresadores, on la ferramenta puja i baixa mentre es segueix un dibuix.

Utilitzem aquest ja que per poder modificar el control de força, és necessari que el robot permeta modificar trajectòries cartesianes a temps real. Els controladors dels robots tenen aquesta capacitat des de fa aproximadament deu anys, però hi ha robots de última generació que no permeten açò.

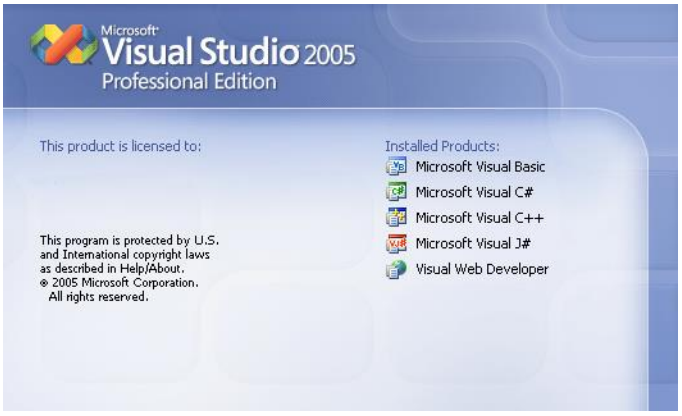
CAPÍTOL 3. DESCRIPCIÓ DEL SISTEMA

3. DESCRIPCIÓ DEL SISTEMA.

3.1 SOFTWARE UTILITZAT.

Per al desenvolupament del programa propòsit d'aquest treball, és a dir, un programa que ens permeta controlar el moviment i la força d'un robot per aconseguir l'objectiu de classificar i col·locar correctament certs elements seleccionats, s'ha utilitzat el codi de programació basat en el llenguatge C.

Per a la implementació del software s'ha utilitzat el programa Microsoft Visual Studio, en la versió del 2005, que encara que és una versió obsoleta és la que disposa l'ordinador del que hem fet ús per a desenvolupar la solució al problema presentat.



IMATGE 3.1 PANTALLA INICI VISUAL STUDIO

Aquest software és un entorn de desenvolupament integrat per a sistemes operatius de Windows. Suporta múltiples llenguatges de programació, com el C++, Java, Python, i també d'altres menys coneguts i menys utilitzats.

No sols s'utilitza per a implementació de programes a mode consola, sinó que a més es pot utilitzar per el desenvolupament web, creant llocs i aplicacions web així com serveis web en qualsevol plataforma que suporte la plataforma .NET.

Per poder treballar amb les targetes d'adquisició de dades, que posteriorment s'expliquen, han sigut necessari introduir llibreries que ens permeten aquesta interacció. Per a fer us de les targetes s'ha introduït la llibreria "ADSAPI32I.lib", per a fer ús del sensor de força hem inclòs la nomenada "jr3pci_ft.lib".

3.2 TARGETES D'ADQUISICIÓ DE DADES.

Com s'ha nomenat abans, no tots els robots en l'actualitat permeten modificar les coordenades cartesianes a temps real. Per al propòsit d'aquest treball que és fer el control de la força del robot açò és indispensable.

Dur a terme aquest control serà possible sempre que disposem d'un sistema de comunicació entre el robot i els seus sensors i l'ordinador. Aquesta comunicació es du a terme per mitjà d'unes targetes d'adquisició de dades.

Les targetes d'adquisició de dades de les que s'ha disposat per a aquest treball són: PCL 833, PCI 1711 i la PCI 1720.

Totes aquestes són de la marca Advantech, que és un fabricant Taiwanes reconegut en l'àmbit mundial. Des de 1983 desenvolupa hardware industrial dins de l'àrea de l'automatització, control i comunicacions industrials. Els seus productes estan enfocats a sistemes embedded, informàtica industrial, automatització industrial, comunicacions i xarxes.

IMATGE 3.2 LOGO EMPRESA ADVANTECH

3.2.1 TARGETA D'ADQUISICIÓ DE DADES PCL 833

La PCL 833 és un encoder de 3 eixos de quadratura, amb un comptador extern per al IBM PC/AT i compatibles. Aquesta targeta permet al PC realitzat el control de moviment de sistemes. Cada entrada inclou un circuit de descodificació per a l'increment de l'encoder. Les característiques principals d'aquesta targeta són:

- Tres comptadors de 24 bits ascendents/descendents.
- Entrades absolutes o diferencials
- Freqüència de entrada de senyal quadràtica màxima 1.0 MHz.
- Freqüència màxima de pols d'entrada 2.4 MHz.
- Entrades absolutes o diferencials.
- Aïllat òpticament fins a 2500V.

- Filtre digital de 4 estats amb freqüències de mostreig seleccionables.

El *encoder* genera polsos que indiquen la posició d'un eix. L'eixida de *l'encoder* inclou dos senyals, que nomenem canal A i canal B, que generen N polsos de revolució. Les dos senyals es troben desfasades un quart de cicle. Aquest desfasament de els senyals permet al controlador determinar la direcció de rotació, depenent de si aplega abans el senyal A o el senyal B.

IMATGE 3.3 PCI 833

Per a obtindre la informació sobre la posició del motor la targeta ofereix diferents maneres de contar tenint en compte per a cadascun d'ells una freqüència màxima per a el senyal d'entrada.

Mode	**Freqüència de entrada màxima**		
	8 MHz	**4 MHz**	**2 MHz**
Quadràtica X1, X2, X4	1 MHz	600 KHz	300 KHz
2-polsos	2.4 MHz	1.2 MHz	600 KHz
Pols/Direcció	2.4 MHz	1.2 MHz	600 KHz

TAULA 3.1 MODES DE FUNCIONAMENT

L'entrada quadràtica determina la direcció amb la comparació del primer senyal que aplegue. En aquest mode existeixen diferents mètodes de comptatge.

X1 El comptador incrementa o disminueix el compte sempre que aparega un flanc de pujada en el canal d'entrada A.

X2 El comptador incrementa o disminueix el compte sempre que aparega un flanc de pujada o baixada en el canal d'entrada A.

X4 El comptador incrementa o disminueix el compte sempre que aparega un flanc de pujada en el canal d'entrada B.

El mode 2-polsos utilitza dos polsos d'entrada com a mètode de comptatge, un per al sentit de les agulles del rellotge i l'altre per al sentit contrari. El comptador disminuirà el compte sempre que es produeixi un flanc de pujada al canal d'entrada A, i s'incrementarà sempre que aparega un flanc de pujada al canal d'entrada B.

Amb el mode pols/direcció la entrada A s'utilitzarà com entrada de pols, mentre que la entrada del canal B s'utilitzarà per a la direcció. Si el canal B està a nivell alt i es produeix un flanc de baixada en el canal A el comptador disminuirà el compte. Si el canal B està a nivell baix i el canal A experimenta un flanc de pujada el compte s'incrementarà.

Per al control de la posició hem utilitzat tant aquesta targeta com la PCI-1720. La targeta d'adquisició de dades PCL 833 s'ha utilitzat per portar el compte dels polsos del *encoder* que es generen amb el moviment de l'eix. Segons el valor d'aquesta operació sabrem si hem aplegat a la posició desitjada o no.

La targeta PCI-1720 actuarà en els motors en funció de la informació de la velocitat i posició que estem rebent de les altres dues targetes.

La característica més important d'aquesta targeta és que, a més d'encarregar-se de convertir els valors digitals calculats per l'ordinador a valors analògics que s'utilitzaran com alimentació per als motors, és que aquests poden ser tant positius com negatius. Així depenent de la posició actual, i de la posició final desitjada el motor podrà girar en una direcció o l'altra.

3.2.2 TARGETA D'ADQUISICIÓ DE DADES PCI 1711

La PCI 1711 es una targeta digital de entrada eixida (E/S) analògica multifuncional, que compta amb cinc de les funcions de control més desitjables per a un PC/AT i compatibles. Es una targeta d'alt rendiment, alta velocitat i d'adquisició de dades multifunció per a ordinador IBM PC/XT/AT i compatibles.

IMATGE 3.4 PCI 1711

La majoria de les targetes de interfície i perifèrics per a PC es controlen amb ports E/S, aquests es direccionen utilitzant el espai de direccionament de ports E/S.

Les característiques principals d'aquesta targeta són:

- 16 canals d'entrada analògics
- Un convertidor industrial estàndard de 12 bits per convertir les entrades analògiques.
- Cerca de canals automàtic/ganancia .

Capacitat per transferir dades convertides de analògic a digital mitjançant control per programa, interrupcions o per DMA.

- 16 TTL/DTL compatibles entrades digitals i 16 canals d'eixida digital.

. Aquesta targeta ha sigut utilitzada per a l'habilitació de bit que permet el funcionament del motor. Anteriorment es disposava de la PCL 812, correctament cablejada, aquesta, a més de per a habilitar el bit per a l'encesa dels motors utilitzava per fer les conversions analògiques-digitals dels senyals corresponents a la mesura de la velocitat dels motors.

Dels 16 canals d'entrada analògics utilitzaven tres per obtenir la velocitat de l'eix. Així que el canal 0 s'utilitzarà per llegir la velocitat de gir de l'eix X, el canal 1 per llegir la velocitat de gir de l'eix Y i per últim el canal 2 per llegir la de l'eix Z.

Per problemes tècnics, i per culpa de l'antiguitat de la targeta PCL 833 ha hagut de ser substituïda per aquesta, que encara que permet la mateixa funcionalitat que l'anterior, per falta de temps no s'han pogut fer les noves connexions per a que puga funcionar com la PCL 833. El canvi de targeta d'adquisició de dades ha suposat també un canvi en la implementació del controlador, ja que no es podia disposar de la velocitat directament, i s'ha hagut de fer una aproximació.

3.2.3 TARGETA D'ADQUISICIÓ DE DADES PCI 1720

La PCI 1720 es una targeta de conversió digital analògica D/A. Aquesta compta amb quatre canals de 12 bits per a la eixida analògica, a més de la possibilitat d'elegir entre 4 rangs de tensió:

- De 0 a 5V.
- De 0 a 10V.
- +/- 5V.
- +/- 10V.

IMATGE 3.5 PCI 1720

D'aquests tres canals hem utilitzat els tres primers per a l'aplicació de les accions de control. Per a aplicar l'acció de control per a l'eix X s'ha utilitat l'eixida 0, per a la de l'eix Y l'eixida 1, i finalment, per a l'eix Z s'ha utilitzat l'eixida 2.

3.3 DESCRIPCIÓ DEL SISTEMA ROBOTITZAT

Per a la realització pràctica del sistema proposat en el objecte del treball es disposa d'un robot porticat. A continuació es detallaran les característiques d'aquest, descomponent-lo en tres parts.

S'utilitza aquesta estructura robotitzada donat que en robots més modernes el software de controlador no es pot modificar, i es el més interesant en aquest treball.

3.3.1 EL ROBOT

El element més característic d'un robot son les seues articulacions, són el que el defineix, es a dir, el que delimita el tipus de moviment que es pot fer, la resolució d'aquest, les característiques espacials, etc.

El robot que s'ha utilitzat en aquest treball té tres articulacions prismàtiques que realitzen un moviment lineal al llarg d'un eix, és a dir, un robot porticat.

Com abans s'ha explicat, es tracta d'un robot de coordenades cartesianes, és a dir, els seus tres eixos formen un sistema cartesià de referència, el que facilitarà la conversió de coordenades reals a coordenades del robot, per a transmetre el moviment. Es pot parlar dels eixos X, Y, Z, corresponent cadascun a un dels eixos del sistema de coordenades.

Cada eix estarà dirigit per un motor de corrent contínu amb característiques similars. Però la manera de transmetre el moviment de l'eix Z serà diferent. En els eixos X i Y existeixen un joc de reductors de RPM (revolucions per minut) que tenen com a entrada l'eix de dit motor, i l'eixida és una corriola. Aquesta corriola actua sobre una roda dentada que transmet el moviment a l'eix per a què aquest tinga un moviment lineal.

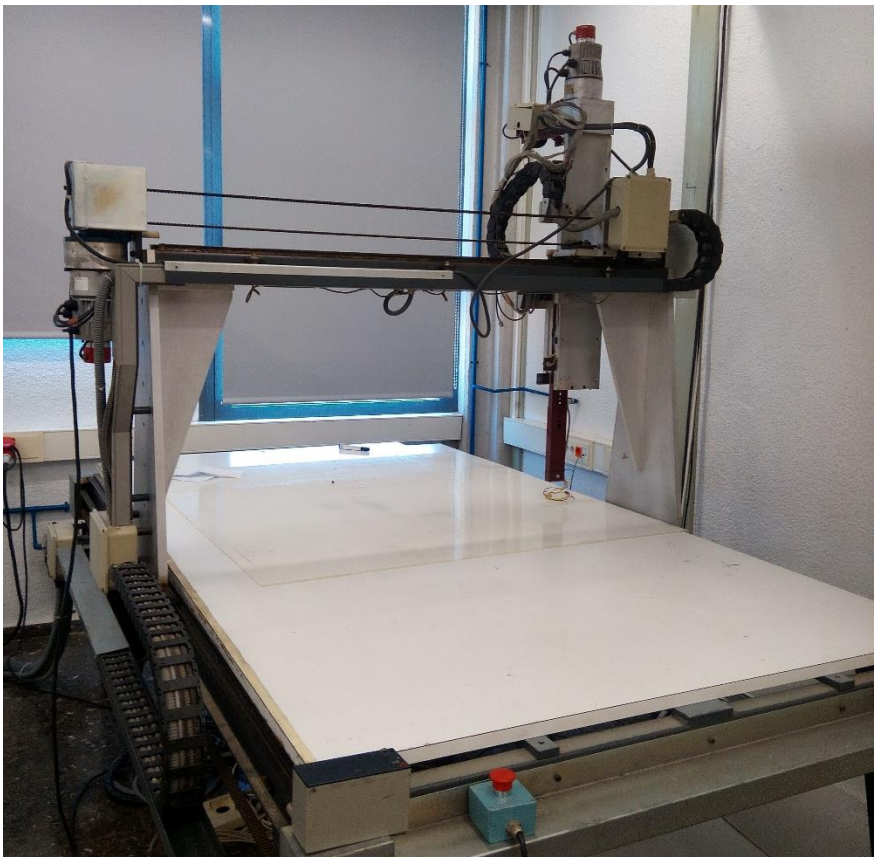
A l'eix Z, en canvi, se li transmet el moviment mitjançant un cargol. Aquest es troba solidari a l'eix de gir del motor, pel que per cada volta d'aquest es produeix un moviment lineal equivalent al pas d'una volta del caragol.

Cada eix disposa d'uns sensors, que seran els mateixos per als tres casos, aquests es poden classificar en dues categories:

- Sensors de final de carrera: Detectors de posició absoluta, s'activen quan la posició del robot està dins d'un rang determinat. Cada eix disposarà de tres finals de carrera, dos per a determinar la proximitat al final de la zona de treball i altre com a referència del punt 0 de l'eix. Son de tipus inductiu, el que vol dir que detecten la proximitat d'un cos metàl·lic, que en aquest cas correspon a l'eix del robot. Aquestos s'utilitzen com a senyals de seguretat, per a què en el cas que el robot estiga fóra de control no es faça massa malbé.

- Sensors de moviment: Cada eix disposa de dos tipus de sensor de moviment, una dinamo tacomètrica i un encoder diferencial. Els encoders mesuren el desplaçament contant els polsos generats per uns leds, utilitzant per a dur a terme açò un comptador extern. La dinamo tacomètrica produeix a la seua eixida una tensió proporcional a la velocitat angular de l'eix, utilitzant-se aquesta mesura per a saber la velocitat de l'eix.

En la figura següent podem vore l'aspecte del robot porticat que anteriorment s'ha explicat. Les mesures total d'aquest són: De llargària dos metres amb seixanta cinc centímetres, i d'amplada un metre quaranta cinc centímetres.



IMATGE 3.6 ROBOT UTILITZAT

3.3.2 EL ARMARI DE CONNEXIONS

Un armari de connexions és l'element encarregat de rebre tot el cablejat. Aquest serveix per a organitzar les connexions de la xarxa, per a què els elements relacionats de la xarxa d'àrea local LAN i els equips que necessiten connectivitat entre ells puguen ser fàcilment incorporats al sistema, i a més que els ports no sofresquen desperfectes pel constant treball d'incorporar i retirar els cables del seu port.

A causa de la gran quantitat de cables necessaris per al sistema robotitzat, s'ha utilitzat en aquest cas un armari pel que passen tots els senyals del sistema per tant de simplificar les connexions, açò és un sistema molt utilitzat quan es treballa amb una gran quantitat de connexions, permet una visió més clara i una actuació més ràpida en el cas d'algun incident.

Es disposa de senyals d'emergència, que són polsadors que inhabiliten els comptadors de velocitat, provocant en la realitat que el moviment dels motors es detinga, els senyals d'emergència són polsadors vistosos de color roig per a ser fàcilment identificats en cas de necessitat. No sols disposem de setes d'emergència en l'armari de connexions, el robot utilitzat per a aquest treball disposa d'altres sistemes de seguretat per poder detenir la seua activitat en qualsevol moment. Un d'ells està incorporat a les portes que separen el robot d'on es troba el computador on s'executa el programa. Si aquestes portes es troben tancades els motors podran fer el seu funcionament normal, ja que hi haurà corrent de llum, si aquestes s'obrin es deixa d'abastir elèctricament l'armari de connexions que fa que es detinga per complet el funcionament del robot porticat

Els elements encarregats de donar alimentació als motors quan es treballa en mode manual són reguladors PID continus, encarregats de regular la velocitat del motor, aquest permet fer posicionaments manuals del robot amb major seguretat y precisió. El PID es maneja mitjançant un senyal externa diferencial, de manera que el motor podrà girar en els dos sentits possibles. El valor d'aquest senyal variarà entre els valors de -12 i 12 volts.

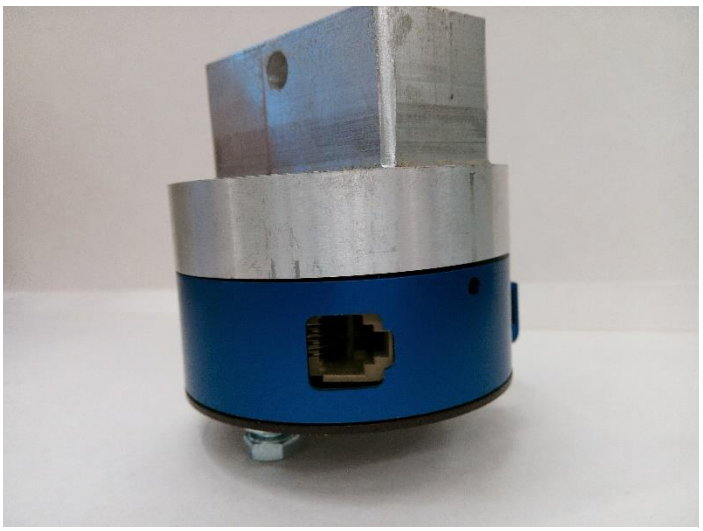
Quan treballem en mode automàtic, l'estratègia de control són els PID implementats que s'expliquen un poc més avant en aquest capítol.

IMATGE 3.7 INTERIOR ARMARI CONNEXIONAT

IMATGE 3.8 EXTERIOR ARMARI CONNEXIONAT

3.4 SENSOR DE FORÇA

El sensor que s'ha utilitzat per a la realització pràctica d'aquest treball és el sensor JR3 67M25A. Es tracta d'un sensor de força par fabricat principalment d'alumini encara que també es pot trobar d'acer inoxidable o titani. Conté sistemes analògics i digitals que formen la seua electrònica interna, així com les galgues encarregades de mesurar la deformació en cadascun dels eixos. Els senyals que llig son amplificats per a convertir-se en representacions analògiques, per a una millor interpretació d'aquesta es converteix en digital.



IMATGE 3.9 SENSOR DE FORÇA JR3

El sensor que s'utilitza en aquest projecte proporciona una eixida digital en un format síncron. Totes les senyals analògiques així com el transformador A/D estan incloses en l'interior del sensor, per poder-se protegir de les interferències electromagnètiques gràcies a la seua carcassa metàl·lica.

Tots aquests sensors tenen cert grau d'acoblament, és a dir, que un moment o una càrrega en un eix pot produeix una càrrega de menor valor en els altres eixos. Cada sensor JR3 està calibrat individualment amb carregues aplicades a cada eix.

En cas del sensor utilitzat, al ser d'eixida digital, guarda les dades en una eixida d'una memòria no volàtil, i aquestes són transferides automàticament descarregant-se en els primers segons de l'operació. Després d'açò el receptor aplica la matriu de calibració a les dades que es van rebent sense intervenció de l'usuari. Si s'haguera de resoldre aquesta matriu manualment seria molt costos i hauria de resoldre's cada una per separat.

3.5 ELECTROIMANT

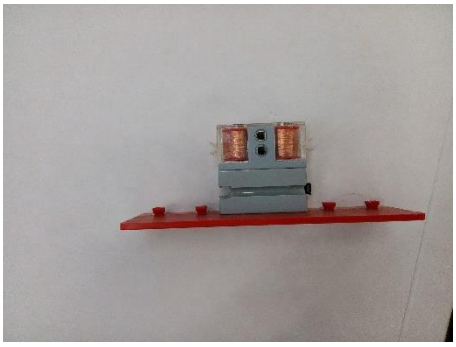
Per poder dur a terme el transport de les llaunes, que és l'objecte de treball del manipulador en aquest objecte s'ha fet servir un electroimant.

Un electroimant és un imant en el que el camp magnètic apareix pel flux de un corrent elèctric. El seu efecte ferromagnètic desapareix quan el corrent elèctric deixa de recórrer-lo.

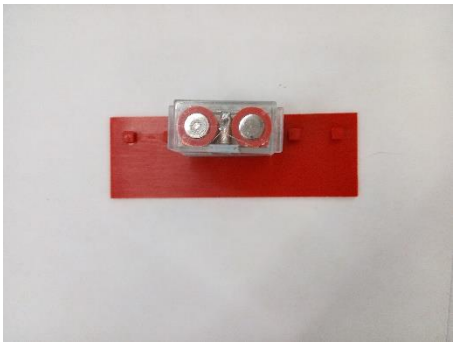
Per a poder resoldre el problema que presenta aquest projecte d'agafar una llauna en un moment donat i soltar-la quan siga pertinent la millor solució trobada es un electroimant. Aquest l'hem acoblat al sensor de força, per a que quan estiga en contacte en la peça transmeta la força al sensor.

L'electroimant està format per dues bobines molt properes per poder generar el camp magnètic. L'utilitzat en aquest projecte no té una gran capacitat, però a la industria s'utilitzen electroimants molt potents capaços d'alçar tonades de ferro.

En la imatge es pot veure l'electroimant utilitzat en aquesta aplicació:



IMATGE 3.11 PLANTA ELECTROIMANT



IMATGE 3.10 ALÇAT ELECTROIMANT

3.6 OBJECTE MANIPULAT

Com s'ha comentat anteriorment l'objecte manipulat són llàunes metàl·liques de diferents altures. Per a la realització del projecte hem comptat amb llaunes de tres altures diferents.

La més alta té una altura total de 20,5 centímetres, la llauna amb altura mitjana té una altura de 11,5 centímetres i la llauna més menuda té una altura de 6,5 centímetres.

S'han utilitzat llaunes lleugeres, amb l'interior buit, ja que l'electroimant del que s'ha disposat no té una gran potència, i per a aquest projecte s'ha assegurat que aquest siga capaç d'alçar i transportar les llaunes.

En la següent imatge es poden veure diferenciades clarament les altures de les llaunes disposades.

IMATGE 3.12 OBJECTE MANIPULAT

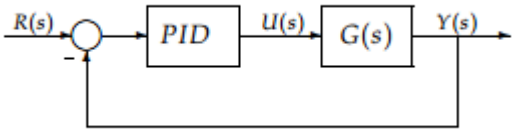
3.7 DESCRIPCIÓ DELS CONTROLADORS

A continuació es va a exposar quins han sigut els controladors seleccionats per a aquest proyecte. S'explicarà tant el seu funcionament, com la rao d'aquesta selecció.

Per últim es farà un anàlisis sobre la manera de traure els paràmetres del controlador.

3.7.1 CONTROLADOR PID

Com s'ha nomenat anteriorment, el controlador PID és un mecanisme de control per realimentació molt utilitzat en sistemes de control industrial. El control PID consta de tres parts. La proporcional, la derivativa i la integral.



IMATGE 3.13 DIAGRAMA DE BLOCS

Primerament, la part **proporcional** es un guany ajustable, aquest té un paper important però limitat a l'hora de seguir una referència, ja que té error en règim permanent. Per a compensar aquest error utilitzem l'acció **integral**, per a que aquesta funcione necesitarem un error distint de 0, és a dir, que la referència a seguir, i la referència que té no siga la mateixa en tot moment. Depenent del valor de l'error en aquesta funció, el control variarà de signe, per a així aconseguir un error en règim permanent sempre nul.

Per ultim l'acció **derivativa** fà més ràpida l'acció de control, és a dir, respon a la velocitat de canvi de l'error, encara que, com a inconvenient pot amplificar els senyals de soroll. Aquesta acció dona amortiguament al sistema, que resultarà en una millora del estat estable.

Aquest controlador s'ha utilitzat per als eixos on era necessaria una major precisió, que són els eixos X i Y, donat que el controlador PID té error zero, és a dir, és capaç de seguir de forma precisa una referència, que en aquest cas és el que era d'interés.

La formula implementada en aquest treball per a dur a terme el control PID ha sigut aquesta:

$$u_k = K_p e_k - K_d(dy_k - dy_{k-1}) + K_i\left(\frac{e_k + e_{k-1}}{2}\right)T_s$$

u_k és l'acció de control a aplicar en cada moment, e_k és l'error que estem cometent al intentar seguir la referència, y_k és la posició real en la que es troba l'eix del controlador i T es el període de temps amb el que obtenim les mostres de les senyals que ofereix el nostre sistema. Les variables K_p, K_i, K_d són les variables del controlador. La derivada de y_k en aquest cas es la velocitat.

Per a donar valors als paràmetres del controlador hem utilitzat el mètode experimental, és a dir, dins d'un rang de valors que puguen donar una bona resposta anar provant fins que es troben els paràmetres que ens donen un millor resultat.

En aquest cas s'ha començat amb la condició de que el valor del controlador no podia ser ni major de deu ni menor de menys deu, ja que són els valors màxims i mínims que podem donar als motors que donen moviment als eixos. Per acurar un poc més, restringim el valor entre ± 5.

Fent les operacions matemàtiques pertinents, suposant un error molt gran i una diferència de velocitats molt gran, tant en sentit positiu com en negatiu, s'han obtes els següents valors:

- K_p, és a dir, el paràmetre proporcional ha de tindre un valor contingut entre 0.001 i 0.006.
- K_d, és a dir, el paràmetre derivatiu ha de tindre un valor contingut entre -0.003 i 0.003.
- K_i o paràmetre integral ha de tindre un valor que oscile entre 0.01 i 0.09.

Dins d'aquest valors s'ha anat provant diversos, apleguent a la solució que s'ha considerat més òptima, que ha sigut:

- K_p= 0.004.
- K_d= 0.001
- K_i = 0.04

3.7.2 CONTROLADOR P

Per a l'eix Z no és necessari un seguiment de la referència tan exacte, per el que un control proporcional és suficient. Així el programa és més senzill de implementar.

L'equació per a aquest controlador serà:

$$u_k = K_p e_k$$

On K_p és el paràmetre proporcional del controlador, i e_k és el error actual del controlador.

Com s'ha realitzat anteriorment, el paràmetre s'ha tret de forma experimental. Com que aquest eix té un recorregut menor, en lloc de deixar un rang de ± 5, s'ha reduït a $\pm 2{,}5$. Suposant un valor d'error, tant en sentit positiu com en negatiu, el paràmetre proporcional havia de tindre un valor similar al 0.0006.

Després de comparar diversos valors similars a aquest s'ha trobat que el que millor resultat dona és:

- K_p = 0.0005

3.7.3 CONTROLADOR DE FORÇA

El control de força utilitzat té una estructura molt similar a la del controlador P del eix Z. Per al control de força nosaltres fiquem una referència de força a exercir. Quant el valor del sensor, és a dir, la força que està exercint realment el sensor de força contra l'objecte manipulat siga igual que el valor de referència que s'ha seleccionat, el valor del controlador ha de ser zero, ja que ha de parar, i no seguir descendint.

L'equació que utilitzarem per a aquest control serà:

$$u_k = K_f(ref_{força} - Val_{força})$$

Aquest control també és per a l'eix Z, que com s'ha comentat anteriorment té un recorregut més curt, per el que el rang d'actuació s'ha limitat a $\pm 2{,}5$, per a evitar que l'eix ixga dels carrils.

Per la similitud amb l'anterior, i com hem comprovat experimentalment que amb el valor que s'ha seleccionat anteriorment s'obté un bon resultat, el valor elegit per a K_f será 0.0005.

CAPÍTOL 4. DESENVOLUPAMENT DE LA SOLUCIÓ

4. DESENVOLUPAMENT DE LA SOLUCIÓ.

4.1 DESCRIPCIÓ DEL SOFTWARE

El seqüencial es la clàssica programació per accions. Aquestes s'ordenen de manera estricta com una seqüència temporal, el comportament del programa dependrà únicament dels efectes de les accions individuals i de l'ordre en les que aquestes tinguen lloc. En aquest cas el temps que cadascuna tarde en realitzar-se no es important .

El software que permetrà el moviment del robot i la classificació de les peces en este treball estarà conformat per una sèrie de funcions seqüencials que permetran al robot controlar el moviment i la força, simulant així a un autòmat.

Entre les accions de les que disposem, la del controlador es la més important a l'hora de fer moure el braç, aquesta li envia una tensió que serà la que active els motors que fan moure el robot . Apareixen també funcions que permetran variar la referencia que segueix el controlador per aconseguir un moviment més suau.

Altres funcions s'encarregaran d'extraure la informació de les targetes d'adquisició de dades per a que el robot puga treballar amb aquesta.

4.1.1 EL CONTROLADOR

La funció **Controlador** bàsicament s'encarrega de dur a terme el control del moviment del robot porticat, i per a açò s'encarregarà de tot allò que tinga a veure amb el moviment d'aquest.

Per a comprendre el funcionament d'aquest primer s'ha d'explicar com funciona el control de moviment. Existeixen dos tipus de controls: El control lineal, on totes les variables del sistema es comporten de manera lineal, es a dir, on seleccionant uns paràmetres arbitraris i aplicant-los a un sistema fixe fan que aquest es comporte de una manera determinada. I el control no lineal, que es durà a terme en els sistemes definits per variables no lineals. En gran part dels casos els sistemes no lineals estan lligats a comportaments no previsibles, o caòtics.

Els algoritmes de control més utilitzats actualment es basen en aproximacions lineals a la dinàmica del manipulador. Encara que es important considerar aquesta de manera completa quan s'estan sintetitzant alguns algoritmes de control. S'estan introduint, a més, en alguns robots algoritmes de control no lineals. Aquestes ofereixen unes millors prestacions , es poden classificar en 4 grups, que han estat exposats amb anterioritat.

Per al càlcul de l'acció de control s'han implementat dos controladors del tipus PID, un per a l'eix x i altre per a l'eix y, la equació en diferencies d'aquests correspondrà a la següent formula, que ha sigut explicada anteriorment:

$$u_k = K_p e_k - K_d(dy_k - dy_{k-1}) + K\_i\left(\frac{e_k + e_{k-1}}{2}\right)T_s$$

Per calcular el valor de l'error en cada moment necessitem calcular la diferència entre la referència que pretenem seguir i la posició real en la que es troba l'eix. La referència haurà sigut fixada per el fil de configuració de trajectòries, i anirà modificant el seu valor mentre s'execute l'aplicació. El valor de la posició de l'eix en cada moment, la targeta d'adquisició de dades 833 que és la que treballa amb la informació proporcionada per els encoders.

El valor de la derivada de la posició en cada moment, es a dir, la velocitat, no s'obtenia de forma calculada, sinó a traves de les targetes d'adquisició de dades, en el nostre cas de la PCL 812 ja que es disposava de la senyal de velocitat del eix, que correspon físicament a la derivada de la posició, per problemes tècnics la targeta PCL ha hagut de ser substituïda per la PCI 1711, que no ens proporciona la velocitat del eix, així que s'ha tingut que implementar. Per a aconseguir aquest propòsit s'ha utilitzat una aproximació senzilla, que es pot veure en la següent equació.

$$v_{eix} = \frac{posició - posició_{ant}}{T}$$

On la variable $posició_{ant}$es la posició immediatament anterior que ha tingut l'eix, es a dir, els polsos captats per el encoder. La variable $posició$ es la posició actual en la que es troba el eix, i per últim T es el període de mostreig, es a dir, el temps que ha passat des de que el programa ha rebut la posició immediatament anterior fins a que obté l'actual.

Aquesta solució és menys precisa que llegir directament la velocitat de la targeta d'adquisició de dades, per el que pot donar algun resultat erroni en algun cas.

S'ha implementat un controlador de posició comú per als tres eixos, per a aconseguir un eficiència major, gràcies a que els tres eixos es moguin a la vegada. Si el controlador fora per a cada eix per separat, a més de que el cost computacional seria més elevat, el robot en si seria menys eficient, ja que per a dur a terme el mateix moviment utilitzaria més temps.

En el cas de l'eix Z, a més del control de posició s'ha implementat un control de força. Quan es vol portar l'eix a la posició de transport aquest no va a exercicir ninguna força, per el que no cal fer un control de força, i sols cal fer un control de posició. Però en el moment en el que aquest eix ha de entrar en contacte amb un objecte s'ha de fer un control de la força que exercix. Per a aquest control el que es preté es reduir la velocitat a mesura que el sensor de força envie al PC un valor de força diferent a zero. Es a dir que, a mesura que l'eix Z comence a estar en contacte amb l'objecte, la velocitat del eix es redueixi fins que la força que està exercint siga la que s'ha marcat com a valor màxim que ha de exercir i llavors pare completament el moviment.

A continuació es pot veure una part del text del controlador de posició dels tres eixos. Podem veure com s'ha implementat el controlador PID i el valor que han pres cadascuna de les seues variables. Per a l'eix Z el controlador no es un PID, sinó un P, ja que no es necessària una gran presició a l'hora de seguir unes coordenades, ja que el seu moviment està controlat bàsicament per el sensor de força, i les dades que aquest envia al PC.

```
	void Controladorposicioxyz(void)
{
	float fTau_x, fTau_y, fTau_z;
	long fErrAnt_x, fErrAnt_y, fErrAnt_z;
	float fKp_x, fKp_y, fKp_z;
	float fKd_x, fKd_y, fKd_z;
	float fKi_x, fKi_y, fKi_z;
	float fKf;
	long fError_z=0;
	long fError_x=0;
	long fError_y=0;
	float fPos_z;
	int iPos_y, iPos_x;
	time_t inici, fi, t;


	fErrAnt_y=fError_y;
	fPosAnt_y=fPos_y;


	fPosAnt_x=fPos_x;
	fErrAnt_x=fError_x;


	t=4.0;
	fKp_y=0.004;
	fKd_y=0.001;
	fKi_y=0.04;
	fKp_x=0.004;
	fKd_x=0.001;
	fKi_x=0.04;
	fKf=0.0005;
	end=0;


	while (end==0){
		iPos_x=LligEncoder(1);
		iPos_y=LligEncoder(2);

		fPos_x=(float)iPos_x;
		fPos_y=(float)iPos_y;
		fPos_z=(float)LligEncoder(3);

		fError_x=refx-fPos_x;
		fError_y=refy-fPos_y;
		fError_z=refz-fPos_z;

	fTau_x=fKp_x*fError_x-fKd_x*((fPos_x-fPosAnt_x)/0.02)+fKi_x*(((fError_x+fErrAnt_x)/2)*0.01);
	fTau_y=fKp_y*fError_y-fKd_y*((fPos_y-fPosAnt_y)/0.02)+fKi_y*(((fError_y+fErrAnt_y)/2)*0.01);
	fTau_z=fKf*fError_z;

		fErrAnt_x=fError_x;
		fPosAnt_x=fPos_x;
```

```
                    fErrAnt_x=fError_x;
                    fPosAnt_x=fPos_x;

                    fErrAnt_y=fError_y;
                    fPosAnt_y=fPos_y;

                    if (fTau_x>=5)        fTau_x=5;
                    else if (fTau_x<=-5)          fTau_x=-5;

                    if (fTau_y>=5)        fTau_y=5;
                    else if (fTau_y<=-5)          fTau_y=-5;

                    EscriuTau(0,fTau_x);
                    EscriuTau(1,fTau_y);
                    EscriuTau(2,fTau_z);

                    if(fPos_x>=refx-5 && fPos_x<=refx+5)
                    {
                    EscriuTau(0,0);
                    }


                    if(fPos_y>=refy-5 && fPos_y<=refy+5)
                    {
                    EscriuTau(1,0);
                    }

                    if(fPos_z>=refz-5 && fPos_z<=refz+5)
                    {
                    EscriuTau(2,0);
                    }

                    if(fPos_x>=refx-5 && fPos_x<=refx+5 && fPos_y>=refy-5 && fPos_y<=refy+5)
                    {
                              end=1;
                    }

                    Sleep(10);


          }
          EscriuTau(0,0);
          EscriuTau(1,0);
          EscriuTau(2,0);


}
```

Aquest es part del codi que s'ha utilitzat per al controlador. Com es pot apreciar s'han utilitzat controls Anti-Windup tant per a l'eix x com per a l'eix y.

En les aplicacions industrials, l'acció de control, com en aquest cas, està acotada entre dos valors, el valor mínim que es pot aplicar a la variable manipulada, i el valor màxim que es pot aplicar a la variable manipulada. Si aquesta variable sen ix d'aquests paràmetres es possible que el comportament no siga el desitjat. Per a açò s'introdueix el control Anti-Windup, que farà que l'algoritme de control deixe d'integrar, es a dir, sumar l'error quan aquest arriba al seu punt de saturació.

La tensió màxima que se li pot transmetre al robot porticat es de valor 10 volts però per a evitar moviments excessivament bruscs del robot s'han saturat en 5V.

Per al control de força exposat abans del eix z s'ha utilitzat un codi diferent. En ell tu introdueixes la màxima força que vols que s'aplique sobre el sensor, a mesura que aquest vaja detectant valors de força anirà disminuint la velocitat, fins a que quan aplegue a aquest valor indicat es detindrà per complet. El codi el podem veure a continuació:

```
void ControladorForsaz(float refz)
{

        float fTau_x, fTau_y, fTau_z;
        float fKf;

        ft = read_ftdata(FILTER4,0);
        //El valor de referència ha de ser en negatiu
        fKf=0.0005;
        fTau_z=fKf*(refz-ft.fz);
        EscriuTau(2,fTau_z);
        if(ft.fz<=refz)
        {
        EscriuTau(2,0);
        }

        PolsosBaixats=fPos_trans_z-LligEncoder(3);


}
```

En aquest llegim les dades del sensor cada vegada que es repetix el bucle en el que està introduït. Se emmagatzema en una variable característica de la llibreria utilitzada, que s'ha nomenat ft. Aplica el controlador i envia la tensió a l'eix z. S'ha introduït també un comptador de polsos, per saber quina ha sigut la quantitat de polsos que s'ha desplaçat fins a ficarse en contacte amb l'objecte a manipular.

4.1.2 GENERACIÓ DE REFERENCIA I TRAJECTÒRIA

Podem definir el modelat de corbes com al procés mitjançant el qual el programador pot generar una família de corbes a partir de una sèrie de punts, de manera que aquesta corba es podrà aplicar a diversos camps de la enginyeria, com per al disseny i fabricació assistits per ordinador.

Aquestes corbes a més es poden utilitzar per a generar trajectòries de referència per a robots i demés maquines.

Algunes de les corbes que es poden generar per a poder utilitzar-les posteriorment en el control de robots seran les següents:

INTERPOLACIÓ POLINÒMICA

Siga $(x_1, y_1),(x_2, y_2)\ldots(x_3, y_3)$ una seqüència de punts en un pla, amb $x_i \neq x_j$ si $i \neq j$. Un polinomi de interpolació grau *n-1* serà:

$$P_n(x) = \sum_{i=1}^{n} y_i \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}$$

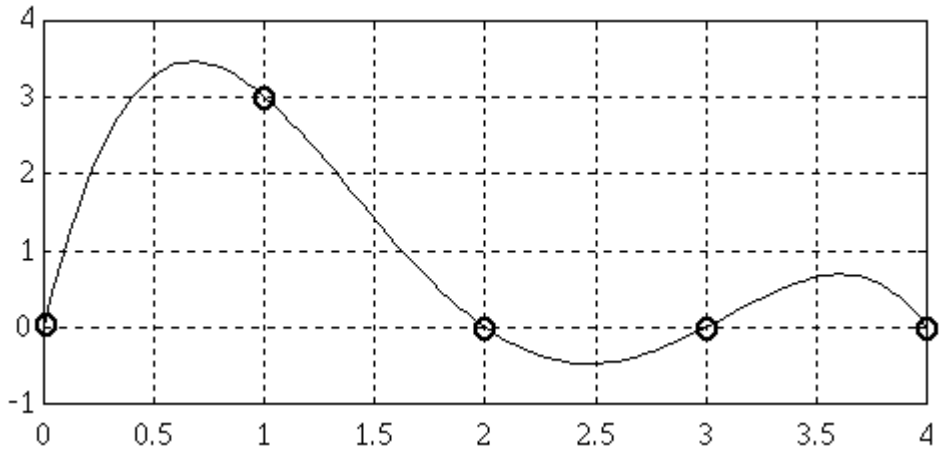
El major problema que aquest tipus d'interpolació planteja es que la corba pot oscil·lar significativament entre dos punts donats. Com en l'exemple següent:

Punts donats (0,0), (1,3), (2,0), (3,0), (4,0)

Polinomi resultat de la interpolació:

$$P_5(x) = -0.5x(x-2)(x-3)(x-4)$$

Representació gràfica d'aquesta:



Si ens fixem solament en els punts donats, es podria preveure una gràfica amb un màxim en el punt (1,3), i que fora nul·la, o pràcticament nul·la en la resta del interval. Però en canvi te dos màxims i un mínim.

INTERPOLACIÓ POLINÒMICA PER SEGMENTS

Com a variant de la interpolació polinòmica anterior es troba aquesta. S'introdueix un punt entremig, la manera de seleccionar-lo no es important per a aquest treball, que ens permet seccionar la recta fent que s'adapte de una manera més precisa a la forma que volem .

En aquesta es defineix la recta tram a tram, podent aconseguir un resultat més aproximat a l'esperat.

Aquesta sols resulta útil en intervals menuts, encara que s'utilitzen de base per a altres que son més apropiades per a definir la trajectòria del robot.

Seguint en l'exemple anterior, fem una interpolació quadràtica per segments:

$$P_a(x) = 6x(0.6 - 0.7x) \qquad 0 \le x \le 1.5$$

$$P_b(x) = 5.4x(x-2)^2 \qquad 1.5 \le x \le 2$$

$$P_c(x) = 0 \qquad 2 \le x \le 4$$

INTERPOLACIÓ D'HERMITE

La interpolació d'Hermite busca un polinomi que passe per tots els punts d'una mostra donada. El polinomi final serà el sumatori de diversos polinomis de cada interval. Cada una de les parts es un polinomi d'ordre 3 de manera que en cada subinterval $[x_{i-1}, x_i], 1 \le i \le n$ es compleixca que la

tangent siga igual en els punts finals de *i-1* i en els d'inici de *i*, es a dir, on les tangents o primeres derivades siguen iguals per les dos parts de la corba.

Si el que tenim es una corba llarga la interpolació d'Hermite pot ser realment complicada i tediosa, i també pot suposar un gran cost computacional.

SPLINE

Son un cas especial de la interpolació d'Hermite, on es poden elegir vectors de la primer a derivada per a que s'iguale la segona derivada també. Aquestes funcions nomenades de corbat o de suavitzat deuen complir certes propietats:

- Les funcions han de permetre que la corba interpole el primer i el últim punt de control, per a que l'usuari tinga control sobre on situar els punts extrems de la corba.
- Les tangents al extrem han de vindre donades per el punt anterior i per el següent spline.
- Les funcions han de ser simètriques respecte al temps, de manera que es puga prendre el sentit invers dels punts sense canviar la forma de la corba.

Bazier va trobar una família de funcions, els polinomis de Bernstein que satisfeien aquestes condicions simple i directament. Aquestes funcions dependran dels punts de control i seran:

$$B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

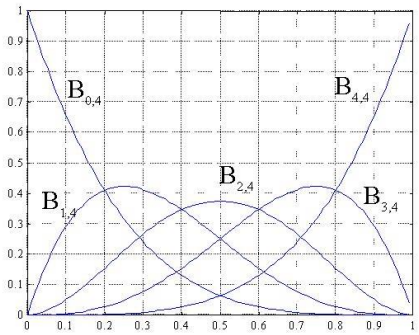
On:

$$\binom{n}{i} = \frac{n!}{i!(n-i)!} t^i$$

Per el que al final la corba de Bezier de splines quedarà:

$$r(t) = \sum_{i=0}^{n} P_i B_{i,n}(t) \qquad t \in [0,1]$$

La següent imatge ens mostra les corbes Bezier de un polinomi de n=4. Es pot apreciar que compleix les característiques anteriorment comentades, com puga ser la simetria en el temps.



n=4

Per a aquest treball la funció que s'encarrega de generar la trajectòria ho podrà fer de tres maneres diferents: escaló, rampa o spline cubica natural.

La trajectòria tipus escaló implica un canvi brusc en la referència, ja que directament la passa com a nova posició de referència. En la tipus rampa el canvi de referència es fa de manera progressiva. Es a dir, es va modificant la referència, des de la posició actual fins a la posició final mitjançant xicotets increments. En el cas del spline el procediment es similar al cas de la rampa, però modificant que la trajectòria es realitza mitjançant la descripció d'una spline cúbica natural.

Aquesta funció s'executarà inicialment. En ella l'usuari podrà elegir quin tipus de trajectòria es va a generar durant la totalitat del programa

Si l'usuari tria l'opció rampa la referència a seguir per el braç serà directament la posició final a la que vol que aplegue el robot. Per a açò no caldrà aplicar cap formula matemàtica. Si en canvi es tria generar trajectòria rampa, la referència ha seguir per el robot s'anirà incrementant amb xicotets increments prèviament calculats, fins a aconseguir la posició final desitjada. Per últim, l'opció restant es el Spline, que actuarà de maner que genere un creixement més lent, el que implicarà un moviment del robot més suau.

```
	void ConfiguraTrajectoria (int xf, int yf, int zf)
{
	int iLligEncoders_x, iLligEncoders_y, iLligEncoders_z;
	float fTs, fTemps_segur, t;
	float x0, y0, z0;

	fTs=0.05;
	fTemps_segur=0.5;
	t=4.0;

	x0=LligEncoder(1);
	y0=LligEncoder(2);
	z0=LligEncoder(3);

	fNomMostres=t/fTs;
	fNomMostres_segur=(t-fTemps_segur)/fTs;

	if(iTipusRef==2)
	{
		Increx=(xf-x0)/fNomMostres_segur;
		Increy=(yf-y0)/fNomMostres_segur;
		Increz=(zf-z0)/fNomMostres_segur;
	}
	else
		if(iTipusRef==3)
		{
			ax=x0;
			cx=3*(xf-x0);
			dx=2*(x0-xf);
```

```
                        ay=y0;
                        cy=3*(yf-y0);
                        dy=2*(y0-yf);

                        az=z0;
                        cz=3*(zf-z0);
                        dz=2*(z0-zf);

                }
        refx=x0;
        refy=y0;
        refz=z0;

        if(iTipusRef==1)
        {
                refx=xf;
                refy=yf;
                refz=zf;
        }
}
```

La funció Configura Trajectòria es l'encarregada de generar les referències que ha de seguir el robot, depenent del tipus de referència que haja triat l'usuari.

Com es pot veure si l'usuari ha triat l'opció 1, es a dir, opció escaló se li assigna directament a la referència a seguir el valor que s'ha introduït. Aquesta com s'ha explicat abans es la funció més senzilla, però la que fa que el moviment del robot siga més brusc.

Si l'usuari tecleja l'opció 2, es a dir, l'opció de generar una trajectòria en forma de rampa, esta funció s'encarrega de calcular el increment que anirà sumant a una referència inicialitzada en 0 fins a aplegar a la posició final que es desitjava per al robot. Aquest increment es calcula amb la diferència de la posició actual i la posició

final, dividit entre un nombre de mostres prèviament calculat, que ha sigut considerat el més convenient per al moviment del robot.

Per últim si la opció seleccionada ha sigut la trajectòria tipus Spline cúbic, aquesta funció s'encarregarà de generar els paràmetres que després s'utilitzarà en l'equació per a anar generant la referència.

```
void GeneraReferencia(int xf, int yf, int zf, int iteracio)
{
if (iteracio<fNomMostres_segur)
{
        if(iTipusRef==1)
        {
                refx=xf;
                refy=yf;
                refz=zf;
        }
        else if (iTipusRef==2)
        {
                refx=refx+Increx;
                refy=refy+Increy;
                refz=refz+Increz;
        }
        else
        {
                tg=iteracio/fNomMostres_segur;
                refx=ax+cx*pow(tg,2)+dx*pow(tg,3);
                refy=ay+cy*pow(tg,2)+dy*pow(tg,3);
                refz=az+cz*pow(tg,2)+dz*pow(tg,3);
        }
}
```

Aquesta finalment es la funció que va canviant la referència a seguir per el eix.

4.1.3 SENSOR DE FORÇA

El sensor de força forma part del controlador del eix Z, aquesta a diferència dels altres, i com ja s'ha explicat anteriorment basa el control del moviment en el valor del sensor de força. Es a dir, quan el sensor de força comence a augmentar els valors de la força que està exercint-se sobre aquest, la velocitat del eix Z s'anirà reduint.

Per a poder adquirir els valor de les forces que s'estan exercint sobre el sensor, primer haurem de inicialitzar-lo, i això es fa de la següent manera:

```
#include "jr3pci_ft.h"

force_array dadesforça, ft;
six_axis_array escalamin, escalamax, escala, sisa;
int axload, kk;


void main (void)
{

        init_jr3(0x1762,0x1111,1,4,1);
```

Com s'ha comentat en anterioritat es necessita una llibreria per poder fer us del dispositiu, per això fem ús del include que apareix en el tros de codi. Declarem variables especials d'aquesta llibreria que són les que ens permetran llegir els valors del sensor. Per a inicialitzar-lo hem de fer ús de unes dades que varien amb el sensor, es a dir, que son pròpies de cada sensor.

CAPÍTOL 5. CONCLUSIONS

5. CONCLUSIONS

5.1 CONCLUSIONS

Després de haver implementat el codi del controlador, i haver aconseguit la comunicació del sensor JR3 amb el PC, el robot manipulador ha aconseguit realitzar amb èxit l'objectiu del projecte, es a dir, la classificació de llaunes depenent del seu tamany.

Una vegada s'havia escrit un codi inicial per al controlador, en el que la lectura de la velocitat del eix era un paràmetre conegut proporcionat per la targeta d'adquisició de dades PCL 812, aquesta va deixar de funcionar i es va tindre que solucionar aquest problema fent una aproximació de la velocitat amb la posició i el temps de mostreig. Aquest ens dona un controlador vàlid, però que mostra un pitjor resultat que amb el valor exacte de la velocitat.

Respecte al moviment del robot, com s'ha explicat anteriorment els motors que feien possible el moviment del robot necessitaven de la entrada de una tensió per a que aquests es poguessin moure. Per a que el motor no es detingues-hi fins que no s'hagués-hi aplegat a la referència desitjada s'utilitza un bucle, on la referència es fracciona i el robot va aconseguint aplegar poc a poc. Un problema trobat ací es que l'eix X és molt pesat, i necessita un nivell mínim més alt del que oferia el controlador inicialment per a moure. S'ha solucionat afegint una restricció que fera que el bucle passara al següent nivell si la tensió que proporcionava el controlador era menor que la mínima exigida per el eix Z per a moure.

Quan es van començar a generar trajectòries, i per a poder facilitar la interacció amb l'usuari, es necessitava saber quants polsos d'encoder corresponien a un centímetre. Es va tindre que fer un programa independent que mesurés els polsos de encoder, i a la vegada es mesurés quant s'havia desplaçat, i així traure una recta. Encara que després sorgí el problema de que aquesta no era tan exacta, i que necessitava de un marge de error per a treballar dins de les mesures desitjades.

Una vegada muntat el sensor de força en l'extrem del eix Z s'havia d'acoblar el electroimant, i aquest no disposava de ningun mecanisme especial per a fer-ho de manera independent. Degut a la configuració del electroimant s'ha hagut de fer dos forats, un a cada costat, per poder mantenir-ho junt al sensor de força amb dos cargols.

S'ha de tenir en compte que degut al canvi de targeta d'adquisició de dades el sensors de final de recorregut tampoc estaven actius, encara que açò no ha sigut cap problema.

Una de les conclusions que es pot extraure d'aquest projecte és que el control intern d'un robot és molt complex, i que al tindre la capacitat de actuar amb el entorn, en aquest cas amb un sensor de força proporciona molts avantatges i facilitats a l'hora de treballar en la industria. Quants més coneixements se li proporcionen al robot sobre el que esta al seu voltant s'aconseguiran moviments més precisos, i per exemple es podrien evitar situacions on el robot no aconseguira trobar mai l'objecte manipulat degut a la acumulació d'error en la seua trajectòria.

5.2 TREBALL FUTUR

Després del desenvolupament d'aquest projecte sorgeixen una sèrie de possibles treballs futurs de manera que proporcionen un major interès a la aplicació, i s'introdueixin altres camps de investigació.

El més immediat podria ser la incorporació de visió al sistema, es a dir, que fora capaç de reconèixer un objecte i calcular la trajectòria que ha de seguir fins a aplegar a ell. Així asseguraríem que el robot sempre esta en la posició correcta per a agafar l'objecte i en la posició correcta per a deixar-lo de nou.

Es podria implementar una opció de funcionament automàtic que ell sols després de deixar una peça anessis a buscar la següent sense la opció de preguntar si hi ha altra peca disponible per a agafar. Per a açò es podria incorporar una cinta que anessis acostant les peces al lloc de recepció. Necessitaríem un sensor addicional que indicarà al robot en quin moment ha aplegat la peca a aquest lloc de recepció, per poder anar a buscar-la.

S'hauria de cablejar de manera correcta la targeta d'adquisició de dades PCI 1711 per a que permetessis rebre la velocitat com un paràmetre, per no tindre que fer la aproximació.

Podria ser interesant també substituir el electroimant actual per un de major potencia, per a que siga capaç de moure qualsevol objecte metàl·lic que necessitem, i no estiga limitat per el seu pes.

CAPÍTOL 6. BIBLIOGRAFÍA

1. Advantec Device Driver's Manual.
   - Informació sobre l'inicialització de les targetes d'adquisició de dades
   - Informació sobre el tractament de cada llibreria.

2. Apunts de l'assignatura 'Tecnologia Automàtica'.
   - Informació sobre el control Anti-Windup.
   - Implementació de controladors PID.
   - Implementació de controladors P.

3. Manual Sensor de Força JR3.
   - Informació sobre com tractar variables i inicialitzar el sensor.
   - Informació sobre com llegir els valors que envia el sensor.

4. Informació sobre els sensors. Disponible en:
   http://www.isa.cie.uva.es/~maria/sensores.pdf

   http://www.info-ab.uclm.es/labelec/Solar/Componentes/SPOSICION.htm

5. Història i teoria de la robòtica. Disponible en:
   http://www.esi2.us.es/~vivas/ayr2iaei/CIN_ROB.pdf

   https://es.wikipedia.org/wiki/Cinem%C3%A1tica_directa

   http://www.frc.utn.edu.ar/jar2006/docs/papers/008-jar06.pdf

6. Apunts de l'assignatura de 'Projectes'.
   - Estructura d'un pressupost.

7. Exemple codi de inicialització targetes d'adquisició de dades. Disponible en:

   https://www.experts-exchange.com/questions/22406155/conversion-LARGE-INTEGER-to-int64-or-operating-with-LARGE-INTEGER.html

PRESSUPOST

1. PRESSUPOST

Aquest capítol conté el pressupost aproximat de la realització i ficada en funcionament del projecte descrit en aquest document. El treball sols correspon al desenvolupament del software, per el que no es pressupostarà ni la instal·lació del robot, ni de les targetes d'adquisició de dades.

No estarà dins del pressupost tampoc l'adquisició del robot porticat ni de l'ordinador de taula, ja que es material del que es disposava en anterioritat.

1.1 QUADRE DE PREUS.

1.1.1 MATERIAL ELÈCTRIC I MECÀNIC.

DESCRICIÓ	PREU
Sensor de força-parell JR3. **Model 67M25A.**	956 €/u
Electroimant. **Electroimant de dues bobines.**	25€/u

TAULA 1.1 QUADRE DE PREUS MATERIAL ELÈCTRIC I MECÀNIC

1.1.2 SOFTWARE I HARDWARE.

DESCRICIÓ	PREU
Visual Studio 2005. **Inclou el programa i la llicencia corresponent**	2,86€/h
Targeta d'adquisició de dades PCI 1711. **Substituint a la PCI 812**	395€/u
Targeta d'adquisició de dades PCI 1720. **Conversor Digital/Analógic**	79,89€/u
Targeta d'adquisició de dades PCL 833. **Amb lector d'encoder**	178,11€/u

TAULA 1.2 QUADRE DE PREUS SOFTWARE I HARDWARE

1.1.3 MÀ D'OBRA

L'hora de treball de un Enginyer industrial ha sigut valorada en 30 euros per hora treballada, la de un tècnic qualificat, encarregat de l'assemblatge de les peces en 20 euros per hora treballada.

DESCRICIÓ	PREU
Enginyera Graduada en Enginyeria de Tecnologies industrials.	30€/h
Supervisora.	30€h
Tècnic de laboratori.	20€/h

TAULA 1.3 QUADRE DE PREUS DE LA MÀ D'OBRA

1.2 QUADRE DE PREUS DESCOMPOSTOS

Descripció unitat d'obra	Unitat	Rendiment	Preu	Import
Comunicació del robot amb el PC mitjançant targetes d'adquisició de dades PCI 1711, PCI 1720 i PCL 833				
Enginyera Graduada en Enginyeria de Tecnologies Industrials	h	10	30	300
Supervisora	h	4	30	120
Tècnic de laboratori	h	4	20	80
Llicència Visual Studio 2005	h	10	2,86	28,6
Targeta d'adquisició de dades PCI 1711	ud	1	395	395
Targeta d'adquisició de dades PCI 1720	ud	1	79,89	79,89
Targeta d'adquisició de dades PCL 833	ud	1	178,11	178,11
Costos directes				1181,6
Costos indirectes complementaris (2%)				23,64
COST TOTAL				1205,24

TAULA 1.4 PREUS DESCOMPOSTOS PRIMERA UNITAT D'OBRA

Descripció unitat d'obra	Unitat	Rendiment	Preu	Import
Incorporació del sensor de força JR3 a l'extrem del eix Z del robot porticat.				
Enginyera Graduada en Enginyeria de Tecnologies Industrials	h	3	30	90
Supervisora	h	1.5	30	45
Tècnic de laboratori	h	5	20	100
Sensor de força JR3	ud	1	956	956
Costos directes				1191
Costos indirectes complementaris (2%)				23,82

COST TOTAL	1214,82

TAULA 1.5 PREUS DESCOMPOSTOS SEGONA UNITAT D'OBRA

Descripció unitat d'obra	Unitat	Rendiment	Preu	Import
Disseny del software Controlador del robot , encarregat de generar el moviment dels eixos d'aquest				
Enginyera Graduada en Enginyeria de Tecnologies Industrials	h	60	30	1800
Supervisora	h	20	30	600
Tècnic de laboratori	h	15	20	300
Llicència Visual Studio 2005	h	60	2,86	171,6

Costos directes	2871,6
Costos indirectes complementaris (2%)	57,44
COST TOTAL	2929,04

TAULA 1.6 PREUS DESCOMPOSTOS TERCERA UNITAT D'OBRA

Descripció unitat d'obra	Unitat	Rendiment	Preu	Import
Disseny del software encarregat de la modificació de la Trajectòria				
Enginyera Graduada en Enginyeria de Tecnologies Industrials	h	15	30	450
Supervisora	h	3	30	90
Tècnic de laboratori	h	0,5	20	10
Llicència Visual Studio 2005	h	15	2,86	42,9

Costos directes	592,9
Costos indirectes complementaris (2%)	11,86
COST TOTAL	604,76

TAULA 1.7 PREUS DESCOMPOSTOS QUARTA UNITAT D'OBRA

Descripció unitat d'obra	Unitat	Rendiment	Preu	Import
Implementació del programa capaç de interpretar les dades enviades per el sensor de força i modificar el moviment				
Enginyera Graduada en Enginyeria de Tecnologies Industrials	h	15	30	450
Supervisora	h	5	30	150
Tècnic de laboratori	h	0,5	20	10
Llicència Visual Studio 2005	h	15	2,86	42,9

Costos directes	652,9
Costos indirectes complementaris (2%)	13,06
COST TOTAL	665,96

TAULA 1.8 PREUS DESCOMPOSTOS QUINTA UNITAT D'OBRA

Descripció unitat d'obra	Unitat	Rendiment	Preu	Import
Redacció dels documents de memòria i pressupost.				
Enginyera Graduada en Enginyeria de Tecnologies Industrials	h	50	30	1500
Supervisora	h	2	30	60
Costos directes				1560
Costos indirectes complementaris (2%)				31,2
COST TOTAL				1591,2

TAULA 1.9 PREUS DESCOMPOSTOS SEXTA UNITAT D'OBRA

1.3 QUADRE DE PREUS UNITARIS

DESCRICIÓ	PREU (€/ud)
Comunicació del robot amb el PC.	1205,24
Incorporació sensor de força JR3.	1214,82
Implementació del codi Controlador.	2929,04
Disseny de software de Trajectòria.	604,76
Desenvolupament programa interpretació dades.	665,96
Redacció de memòria i pressupost.	1591,2

TAULA 1.10 QUADRE DE PREUS DE LA MÀ D'OBRA

1.4 PRESSUPOST D'EXECUCIÓ MATERIAL

Descripció	Quantitat	Import
Comunicació del robot amb el PC	**1**	**1205,24**
Incorporació sensor de força JR3.	**1**	**1214,82**
Implementació del codi Controlador.	**1**	**2929,04**
Disseny de software de Trajectòria.	**1**	**604,76**
Desenvolupament programa interpretació dades.	**1**	**665,96**
Redacció de memòria i pressupost.	**1**	**1519,2**
PRESSUPOST D'EXECUCIÓ MATERIAL (PEM)		**8211,02**
Despeses generals (12% PEM)		**985,33**
Benefici industrial (6%PEM)		**492,67**
PRESSUPOST D'EXECUCIÓ PER CONTRACTA (PEC)		**9689,02**
IVA (21% PEC)		**2034,70**
PRESSUPOST BASE LICITACIÓ		**11723,72**

El pressupost per licitació del projecte plantejat suma un total de:

ONCE MIL SET-CENTS VINT I TRES EUROS AMB SETANTA DOS CÈNTIMS

ANNEXES

1. ANNEXES

A continuació es troba el codi complet de la aplicació desenvolupada, explicada pas a pas sobre el codi. Com ja s'ha exposat abans està escrita en llenguatge de programació C. No s'han inclòs les llibreries de les targetes d'adquisició de dades ni del sensor de força, ja que no han sigut modificades en aquest projecte.

APLICACIÓ DE CONTROL DE POSICIÓ I FORÇA

```
#include <Afxwin.h>

#include <stdio.h>

#include <conio.h>

#include <iostream>

#include <time.h>

#include <windows.h>

#include <math.h>

//Include per als drivers d'advantech

#include "driver.h"


//Include per al sensor de força

#include "jr3pci_ft.h"


using namespace std;




void ConfiguraTargetes();

void EscriuTau(int quin, float valor);
```

```
void TancaTargetes();

long LligEncoder (int quin);

void Sintonitzax(void);

void Sintonitzay(void);

void Sintonitzaz(void);

void GeneraReferencia(int xf,int yf, int zf, int iteracio);

void ConfiguraTrajectoria(int xf, int yf, int zf);

void Controladorposicioxyz(void);

void ControladorForsaz(float refz);

void ObtindrePolsos(int cx, int cy);

void ConfiguraDistancia(void);

void AgafemPeça(void);

void SoltemPeça(void);


//Variable que ens permetrà encendre els motors

PT_DioWriteBit enable;


//Variables globals per a la generació de referència

float refx, refy, refz, ppmms_x, ppmms_y, ppmms_z;

long repos_x, repos_y, repos_z;

int mostres, iMmtres,iTipusRef;

int statusBitEnable;

int condfi;

float ax,cx,dx,ay,cy,dy,az,cz,dz,xf,yf,zf;

float fNomMostres, fNomMostres_segur;

float Increx, Increy, Increz;
```

```
float tg;

float DisTotx, DisToty;

float Distanciax, Distanciay;

int i,k,q,w,l,n,b,a,s,d,f,g;

bool end, final;

long polsosx, polsosy;

int ipolsosx, ipolsosy, dpolsosx, dpolsosy, fpolsosx, fpolsosy;

int iPols_enc_x, iPols_enc_y, iPols_enc_z;

int PolsosBaixats, PolsosAPujar;

float fPos_x, fPosAnt_x;

float fPos_y, fPosAnt_y;

float fPos_z,fPos_trans_z;


//Variables globals per al sensor de força

force_array dadesforça, ft;

six_axis_array escalamin, escalamax, escala, sisa;

int axload, kk;



//Variables per a inicialitzar les targetes

long DeviceHandler833;

long DeviceHandler1720;

long DeviceHandler1711;
```

```
void main (void)
{
    char seguir,continuar;
    refx=0;
    refy=0;
    refz=0;
    time_t t;
    char cTipusRef;
    i=1;
    float PolsosBaixats,PolsosAPujar;

    final=0;
    k=1;
    q=1;
    w=1;
    l=1;
    n=1;
    b=1;
    a=1;
    s=1;
    d=1;
    f=1;
    g=1;
    fPos_x=0.0;
    fPos_y=0.0;
```

```
xf=0.0;

yf=0.0;

zf=0.0;




ConfiguraTargetes();


//Inicialitzem el sensor de força

init_jr3(0x1762,0x1111,1,4,1);


//Parem els motors abans de començar per si tenen algun moviment remanent

EscriuTau(0,0);

EscriuTau(1,0);

EscriuTau(2,0);


//Es dona a elegir el tipus de trajectoriaa seguir per el controlador

//Serà la mateixa durant tot el funcionament

        printf ("\n Tria el tipus de trajectoria que vols generar:\n");

        printf ("\t Escalo (1)\n");

        printf ("\t Rampa  (2)\n");

        printf ("\t Spline (3)\n");

        cTipusRef=getchar();

        iTipusRef=atoi(&cTipusRef);
```

```
do{

        ConfiguraTargetes();

        //Es demanen els moviments per portarlo a la posició de referència
        Sintonitzax();
        Sintonitzay();
        Sintonitzaz();
        final=0;
        fPos_z=(float)LligEncoder(3);

        ConfiguraTrajectoria(xf, yf, zf);



        while (final==0)
        {
                if(iTipusRef==1)
                {
            GeneraReferencia(xf,yf,zf,i);
            Controladorposicioxyz();
            fPos_trans_z=(float)LligEncoder(3);
            final=1;
            Sleep(10);

                }
                else
```

```
                        {


                                while(q!=fNomMostres)
                                {
                                        GeneraReferencia(xf,yf,zf,q);
                                        Controladorposicioxyz();
                                        q++;


                                }
                        fPos_trans_z=(float)LligEncoder(3);
                        final=1;




                }
        }


        printf("\n Desitja continuar modificant la posició?(s/n)");
        fflush(stdin);
        scanf("%c", &seguir);

} while (seguir != 'n');



printf("El braç esta en la posicio de referencia\n");
```

```
Sleep(1000);



 ConfiguraDistancia();


 Sleep(2000);


 ObtindrePolsos(10,20);
 ipolsosx=(int)polsosx-2000;
 ipolsosy=(int)polsosy;


 fpolsosx=(float)LligEncoder(1)+ipolsosx;
 fpolsosy=(float)LligEncoder(2)+ipolsosy;


 Sleep(3000);
 k=1;



 while(k!=fNomMostres)
                                    {
                                        GeneraReferencia(fpolsosx,fpolsosy,0,k);
                                        Controladorposicioxyz();
                                        k++;
```

```
                                        }



ObtindrePolsos(Distanciax,Distanciay/2);

ipolsosx=(int)polsosx-2000;

ipolsosy=(int)polsosy-1000;


dpolsosx=fpolsosx+ipolsosx;

dpolsosy=fpolsosy+ipolsosy;



Sleep(3000);

final=0;







do{


k=1;

q=1;

w=1;

l=1;

n=1;

b=1;

a=1;
```

```
s=1;
d=1;
f=1;
g=1;
                                        while(q!=fNomMostres)
                                        {
                                                GeneraReferencia(dpolsosx,dpolsosy,fPos_trans_z,q);
                                                Controladorposicioxyz();
                                                q++;
                                        }
                                        while(a!=fNomMostres)
                                        {
                                                GeneraReferencia(dpolsosx,dpolsosy,-600,a);
                                                ControladorForsaz(refz);
                                                a++;
                                        }
                                        AgafemPeça();
                                        while(s!=fNomMostres)
                                        {
                                                GeneraReferencia(dpolsosx,dpolsosy,fPos_trans_z,s);
                                                Controladorposicioxyz();
                                                s++;
                                        }
                                        q=1;

                                        if(PolsosBaixats>=21200 && PolsosBaixats<=22000)
```

```
                                        {
                                                while(q!=fNomMostres)
                                                {
GeneraReferencia(fpolsosx,dpolsosy+ipolsosy,fPos_trans_z,q);
                                                Controladorposicioxyz();
                                                q++;
                                                }
                                                while(f!=fNomMostres)
                                                {
                                                GeneraReferencia(fpolsosx,dpolsosy+ipolsosy,-600,f);
                                                ControladorForsaz(refz);
                                                f++;
                                                }
                                                        SoltemPeça();
                                                while(w!=fNomMostres)
                                                {
GeneraReferencia(fpolsosx,dpolsosy+ipolsosy,fPos_trans_z,w);
                                                Controladorposicioxyz();
                                                w++;
                                                }
                                                while(g!=fNomMostres)
                                                {
                                                GeneraReferencia(ipolsosx,ipolsosy,fPos_trans_z,g);
                                                Controladorposicioxyz();
                                                g++;
```

```
                }


        }
        else if (PolsosBaixats>=28500 && PolsosBaixats<=29200)
        {
                while(k!=fNomMostres)
                {
                GeneraReferencia(fpolsosx,dpolsosy,fPos_trans_z,k);
                Controladorposicioxyz();
                k++;
                }
                while(f!=fNomMostres)
                {
                GeneraReferencia(fpolsosx,dpolsosy,-600,f);
                ControladorForsaz(refz);
                f++;
                }
                        SoltemPeça();


                while(l!=fNomMostres)
                {
                GeneraReferencia(fpolsosx,dpolsosy,fPos_trans_z,l);
                Controladorposicioxyz();
                l++;
                }
                while(d!=fNomMostres)
```

```
                                        {
                                        GeneraReferencia(ipolsosx,ipolsosy,fPos_trans_z,d);
                                        Controladorposicioxyz();
                                        d++;
                                        }



                                }
                                else if(PolsosBaixats>=31000 && PolsosBaixats<=31500)
                                {
                                        while(n!=fNomMostres)
                                        {
                                        GeneraReferencia(fpolsosx,dpolsosy-
ipolsosy,fPos_trans_z,n);
                                        Controladorposicioxyz();
                                        n++;
                                        }
                                        while(k!=fNomMostres)
                                        {
                                        GeneraReferencia(fpolsosx,dpolsosy-ipolsosy,-600,k);
                                        ControladorForsaz(refz);
                                        k++;
                                        }
                                        SoltemPeça();
                                        while(b!=fNomMostres)
                                        {
```

```
                                        GeneraReferencia(fpolsosx,dpolsosy-
ipolsosy,fPos_trans_z,b);

                                        Controladorposicioxyz();

                                        b++;

                                        }

                                        while(d!=fNomMostres)

                                        {

                                        GeneraReferencia(ipolsosx,ipolsosy,fPos_trans_z,d);

                                        Controladorposicioxyz();

                                        d++;

                                        }


                                final=1;


                                }


            printf("\n Hi ha altra llauna per a classificat?(s/n)");

            fflush(stdin);

            scanf("%c", &continuar);


    } while (continuar != 'n');




    Sleep(5000);
```

```
        TancaTargetes();



}
void EscriuTau(int quin, float valor)
{
        PT_AOVoltageOut ptAOVoltageOut;


        ptAOVoltageOut.chan = quin;
        ptAOVoltageOut.OutputValue = valor;


        DRV_AOVoltageOut(DeviceHandler1720,&ptAOVoltageOut);
}


void ConfiguraTargetes()
{
        clock_t start, finish;
        double duration;
        PT_QCounterConfig contador1, contador2, contador3;
        PT_QCounterConfigSys sys1, sys2, sys3;
        PT_QCounterStart contador1start, contador2start, contador3start;
        PT_AIVoltageIn lecturavolt;
        int status726, status833, status812, status1720, status1711;
        long referencia, errorpos, incre;
        double acontrol, Kp;
        PT_AIConfig config812;
```

```
    PT_AOConfig config1720;



    USHORT matgan[2];


    //
    // Tarjetas:       0: PCI1720
    //                                 1: PCL833
    //                                 2: PCI1711



    status1720=DRV_DeviceOpen(0, &DeviceHandler1720);


    config1720.chan=0;
    config1720.RefSrc=0;
  config1720.MinValue=-10;
    config1720.MaxValue=10;
    status1720=DRV_AOConfig(DeviceHandler1720,&config1720);


    config1720.chan=1;
    config1720.RefSrc=0;
  config1720.MinValue=-10;
    config1720.MaxValue=10;
    DRV_AOConfig(DeviceHandler1720,&config1720);


    config1720.chan=2;
```

```
    config1720.RefSrc=0;
  config1720.MinValue=-10;
    config1720.MaxValue=10;
    DRV_AOConfig(DeviceHandler1720,&config1720);


    //obrim la PCL833
    status833=DRV_DeviceOpen(1,&DeviceHandler833);



    // Configuració canal encoder
    contador1.counter=0; // contador 0=eix X



    contador1.LatchSrc=0;
    contador1.LatchOverflow=0;
    contador1.ResetOnLatch=0;
    contador1.ResetValue=1;
    sys1.CascadeMode=0;
    sys1.SysClock=0;
    sys1.TimeBase=0;
    DRV_QCounterConfigSys(DeviceHandler833, &sys1);
    DRV_QCounterConfig(DeviceHandler833, &contador1);

    contador1start.counter=0;
    contador1start.InputMode=1;
    DRV_QCounterStart(DeviceHandler833, &contador1start);
```

```
contador1.counter=1; // contador 1=eix Y



contador1.LatchSrc=0;
contador1.LatchOverflow=0;
contador1.ResetOnLatch=0;
contador1.ResetValue=1;
sys1.CascadeMode=0;
sys1.SysClock=0;
sys1.TimeBase=0;
DRV_QCounterConfigSys(DeviceHandler833, &sys1);
DRV_QCounterConfig(DeviceHandler833, &contador1);

contador1start.counter=1;
contador1start.InputMode=1;
DRV_QCounterStart(DeviceHandler833, &contador1start);



contador1.counter=2; // contador 2=eix Z



contador1.LatchSrc=0;
contador1.LatchOverflow=0;
contador1.ResetOnLatch=0;
contador1.ResetValue=1;
```

```
        sys1.CascadeMode=0;
        sys1.SysClock=0;
        sys1.TimeBase=0;
        DRV_QCounterConfigSys(DeviceHandler833, &sys1);
        DRV_QCounterConfig(DeviceHandler833, &contador1);

        contador1start.counter=2;
        contador1start.InputMode=1;
        DRV_QCounterStart(DeviceHandler833, &contador1start);

        status1711=DRV_DeviceOpen(2, &DeviceHandler1711);
        enable.port=0;
        enable.bit=0;
        enable.state=1;

        statusBitEnable=DRV_DioWriteBit(DeviceHandler1711, &enable);




}

void TancaTargetes(void)
{

    for(int i=0;i<5;i++)
```

```
        {
                EscriuTau(i,0.0);
        }
        DRV_DeviceClose(&DeviceHandler833);
        DRV_DeviceClose(&DeviceHandler1720);
        enable.port=0;
        enable.bit=0;
        enable.state=0;
        statusBitEnable=DRV_DioWriteBit(DeviceHandler1711, &enable);
        DRV_DeviceClose(&DeviceHandler1711);

}
long LligEncoder(int quin)
{
        PT_QCounterRead lecturacont1;
        ULONG comptebaix,comptealt;

        if(quin==1){
                lecturacont1.counter=0;
                lecturacont1.LoCount=new ULONG;
                lecturacont1.HiCount=new ULONG;
                lecturacont1.overflow=new USHORT;
                DRV_QCounterRead(DeviceHandler833, &lecturacont1);
                comptebaix=(*lecturacont1.LoCount);
                comptealt=(*lecturacont1.HiCount);
                comptebaix=comptebaix & 0xffffffff;
```

```
            return comptebaix+(comptealt<<32)-8388608; // El que retorna, es a dir, el que llig
            }

else
  if (quin==2){
            lecturacont1.counter=1;
            lecturacont1.LoCount=new ULONG;
            lecturacont1.HiCount=new ULONG;
            lecturacont1.overflow=new USHORT;
            DRV_QCounterRead(DeviceHandler833, &lecturacont1);
            comptebaix=(*lecturacont1.LoCount);
            comptealt=(*lecturacont1.HiCount);
            comptebaix=comptebaix & 0xffffffff;

            return comptebaix+(comptealt<<32)-8388608;
}

else{
            lecturacont1.counter=2;
            lecturacont1.LoCount=new ULONG;
            lecturacont1.HiCount=new ULONG;
            lecturacont1.overflow=new USHORT;
            DRV_QCounterRead(DeviceHandler833, &lecturacont1);
            comptebaix=(*lecturacont1.LoCount);
            comptealt=(*lecturacont1.HiCount);
```

```
                comptebaix=comptebaix & 0xffffffff;


                return comptebaix+(comptealt<<32)-8388608;
        }
}

void Controladorposicioxyz(void)
{
        float fTau_x, fTau_y, fTau_z;
        long fErrAnt_x, fErrAnt_y, fErrAnt_z;
        float fKp_x, fKp_y, fKp_z;
        float fKd_x, fKd_y, fKd_z;
        float fKi_x, fKi_y, fKi_z;
        float fKf;
        long fError_z=0;
        long fError_x=0;
        long fError_y=0;
        float fPos_z;
        int iPos_y, iPos_x;
        time_t inici, fi, t;



        fErrAnt_y=fError_y;
        fPosAnt_y=fPos_y;
```

```
fPosAnt_x=fPos_x;

fErrAnt_x=fError_x;



t=4.0;


fKp_y=0.004;


fKd_y=0.001;


fKi_y=0.04;


fKp_x=0.004;


fKd_x=0.001;


fKi_x=0.04;



fKf=0.0005;


end=0;


                while (end==0)

                        {
```

```
                    iPos_x=LligEncoder(1);

                    iPos_y=LligEncoder(2);

                    ft = read_ftdata(FILTER4,0);


                    fPos_x=(float)iPos_x;

                    fPos_y=(float)iPos_y;

                    fPos_z=(float)LligEncoder(3);



                    fError_x=refx-fPos_x;

                    fError_y=refy-fPos_y;


                    fTau_x=fKp_x*fError_x-fKd_x*((fPos_x-
fPosAnt_x)/0.02)+fKi_x*(((fError_x+fErrAnt_x)/2)*0.01);

                    fTau_y=fKp_y*fError_y-fKd_y*((fPos_y-
fPosAnt_y)/0.02)+fKi_y*(((fError_y+fErrAnt_y)/2)*0.01);

                    fTau_z=fKf*refz;


                    fErrAnt_x=fError_x;

                    fPosAnt_x=fPos_x;


                    fErrAnt_y=fError_y;

                    fPosAnt_y=fPos_y;
```

```
if (fTau_x>=5)
{
        fTau_x=5;
}
else if (fTau_x<=-5)
{
        fTau_x=-5;
}



if (fTau_y>=5)
{
        fTau_y=5;
}
else if (fTau_y<=-5)
{
        fTau_y=-5;
}



EscriuTau(0,fTau_x);
EscriuTau(1,fTau_y);
EscriuTau(2,fTau_z);
```

```
if(fTau_x<=0.8 && fTau_x>-0.8)
{
        if(refx!=0)
        {
                end=1;
        }
        else
        {
                fPos_x=refx;
        }
}

if(fPos_x>=refx-5 && fPos_x<=refx+5)
{
EscriuTau(0,0);
}



if(fPos_y>=refy-5 && fPos_y<=refy+5)
{
EscriuTau(1,0);
}

if(fPos_z>=refz-5 && fPos_z<=refz+5)
{
```

```
                                EscriuTau(2,0);
                                }

                                if(fPos_x>=refx-5 && fPos_x<=refx+5 && fPos_y>=refy-5 &&
fPos_y<=refy+5)
                                {
                                        end=1;
                                }

                                Sleep(10);


                        }
                        EscriuTau(0,0);
                        EscriuTau(1,0);
                        EscriuTau(2,0);


    }

    void ControladorForsaz(float refz)
    {

        float fTau_x, fTau_y, fTau_z;
        float fKf;

        ft = read_ftdata(FILTER4,0);
```

```
        //El valor de referència ha de ser en negatiu
        fKf=0.0005;
        fTau_z=fKf*(refz-ft.fz);
        EscriuTau(2,fTau_z);
        if(ft.fz<=refz)
        {
        EscriuTau(2,0);
        }

        PolsosBaixats=fPos_trans_z-LligEncoder(3);


}



void ConfiguraTrajectoria (int xf, int yf, int zf)
{
        int iLligEncoders_x, iLligEncoders_y, iLligEncoders_z;
        float fTs, fTemps_segur, t;
        float x0, y0, z0;

        fTs=0.05;
        fTemps_segur=0.5;
        t=4.0;
```

```
x0=LligEncoder(1);
y0=LligEncoder(2);
z0=LligEncoder(3);



fNomMostres=t/fTs;
fNomMostres_segur=(t-fTemps_segur)/fTs;

if(iTipusRef==2)
{
	Increx=(xf-x0)/fNomMostres_segur;
	Increy=(yf-y0)/fNomMostres_segur;
	Increz=(zf-z0)/fNomMostres_segur;
}
else
	if(iTipusRef==3)
	{
		ax=x0;
		cx=3*(xf-x0);
		dx=2*(x0-xf);

		ay=y0;
		cy=3*(yf-y0);
		dy=2*(y0-yf);

		az=z0;
```

```
                        cz=3*(zf-z0);
                        dz=2*(z0-zf);

                }
        refx=x0;
        refy=y0;
        refz=z0;

        if(iTipusRef==1)
        {
                refx=xf;
                refy=yf;
                refz=zf;
        }
}
void GeneraReferencia(int xf, int yf, int zf, int iteracio)
{

if (iteracio<fNomMostres_segur)
{
        if(iTipusRef==1)
        {
                refx=xf;
                refy=yf;
                refz=zf;
        }
```

```
        else if (iTipusRef==2)
        {
                refx=refx+Increx;
                refy=refy+Increy;
                refz=refz+Increz;
        }
        else
        {
                tg=iteracio/fNomMostres_segur;
                refx=ax+cx*pow(tg,2)+dx*pow(tg,3);
                refy=ay+cy*pow(tg,2)+dy*pow(tg,3);
                refz=az+cz*pow(tg,2)+dz*pow(tg,3);
        }
    }
    else
    {
        refx=xf;
        refy=yf;
        refz=zf;

    }
    }


    void Sintonitzax(void){
```

```
char linea[100];
int polsos;
char coord, sentit;
PT_QCounterConfig contador1, contador2, contador3;
PT_QCounterConfigSys sys1, sys2, sys3;
PT_QCounterStart contador1start, contador2start, contador3start;
int iPols_enc_x;
coord='i';
printf("Cal dur el robot a l'oritge de l'eix x\n");
printf("Introdueix el moviment a realitzar (coordenada, sentit i pols, junt) i polsa 'return'\n");
printf("Quan siguen les coordenades definitives polsa 'f'\n");

while(coord!='f')
        {
                cin.getline(linea,100);
                coord=linea[0];
                sentit=linea[1];
                polsos=atoi(&linea[2]);
                iPols_enc_x=LligEncoder(1);

                if (coord=='x')
                {
                        if (sentit=='+')
                        {
                                xf=iPols_enc_x+polsos;
```

```
                                }


                                else
                                {
                                        xf=iPols_enc_x-polsos;


                                }
                        }



        }

}

void Sintonitzay(void){

        char linea[100];
        int polsos;
        char coord, sentit;
        int iPols_enc_y;
        PT_QCounterConfig contador1, contador2, contador3;
        PT_QCounterConfigSys sys1, sys2, sys3;
        PT_QCounterStart contador1start, contador2start, contador3start;


        coord='i';
        printf("Cal dur el robot a l'oritge de l'eix y\n");
```

```
printf("Introdueix el moviment a realitzar (coordenada, sentit i polsos, junt) i polsa 'return'\n");
printf("Quan siguen les coordenades definitives polsa 'f'\n");

while(coord!='f')
	{
		cin.getline(linea,100);
		coord=linea[0];
		sentit=linea[1];
		polsos=atoi(&linea[2]);
		iPols_enc_y=LligEncoder(2);

		if (coord=='y')
		{
			if (sentit=='+')
			{
				yf=iPols_enc_y+polsos;

			}

			else
			{
				yf=iPols_enc_y-polsos;

			}
		}
```

```
        }
    }
    void Sintonitzaz(void){

        char linea[100];
        int polsos;
        char coord, sentit;
        int iPols_enc_z;
        PT_QCounterConfig contador1, contador2, contador3;
        PT_QCounterConfigSys sys1, sys2, sys3;
        PT_QCounterStart contador1start, contador2start, contador3start;

        coord='i';
        printf("Cal dur el robot a l'oritge de l'eix z\n");
        printf("Introdueix el moviment a realitzar (coordenada, sentit i pols, junt) i polsa 'return'\n");
        printf("Quan siguen les coordenades definitives polsa 'f'\n");

        while(coord!='f')
                {
                        cin.getline(linea,100);
                        coord=linea[0];
                        sentit=linea[1];
                        polsos=atoi(&linea[2]);
                        iPols_enc_z=LligEncoder(3);
```

```
                    if (coord=='z')
                    {
                        if (sentit=='+')
                        {
                            zf=zf+polsos;

                        }

                        else
                        {
                            zf=zf-polsos;

                        }

                    }

    }
}

void ObtindrePolsos(int cx, int cy)
{

        polsosx=(long)853.691*cx+16,818;
        polsosy=(long)749.3*cy+106,17;
   }
```

```
void ConfiguraDistancia(void)
{

printf("Introdueix la separacio entre la llauna a agafar i la caixa on s'ha de dipositar en cm\n");
scanf("%f",&Distanciax);



printf("Introdueix la separació entre la caixa de les llaunes mes altes i la de les llaunes més baixes\n");
scanf("%f",&Distanciay);


}
void AgafemPeça(void)
{
    PT_AOVoltageOut ptAOVoltageOut;


    ptAOVoltageOut.chan = 3;
    ptAOVoltageOut.OutputValue = 3;


    DRV_AOVoltageOut(DeviceHandler1720,&ptAOVoltageOut);
}


void SoltemPeça(void)
{
    PT_AOVoltageOut ptAOVoltageOut;
```

```
        ptAOVoltageOut.chan =3;

        ptAOVoltageOut.OutputValue =0;



        DRV_AOVoltageOut(DeviceHandler1720,&ptAOVoltageOut);

    }
```