Escola Politècnica Superior
d'Enginyeria de Vilanova i la Geltrú
UNIVERSITAT POLITÈCNICA DE CATALUNYA

PROJECTE FI DE CARRERA

TÍTOL: Pantalla de producció per comunicar amb diversos robots KUKA

AUTOR: Patricia Espín Garcia

TITULACIÓ: Enginyeria en Automàtica i Electrònica Industrial

DIRECTOR: Pere Ponsa Asensio

DEPARTAMENT: Enginyeria de Sistemes, Automàtica i Informàtica Industrial

DATA: juny de 2011

PROJECTE FI DE CARRERA

RESUM (màxim 50 línies)

Avui dia és més comú presentar els productes de forma visual, atractiva i funcional. En el món de la robòtica industrial passa exactament el mateix. KUKA disposa de software per a la creació d'una interfície que pot funcionar per sobre de l'execució dels programes i pot facilitar la tasca del operari. No obstant aquestes pantalles se les ha de crear el propi client.

Després de les múltiples demandes per part dels clients, KUKA ha pensat que seria realment profitós crear un estàndard de pantalla que, arrancant els robots en automàtic extern, s'hi pugui visualitzar l'estat dels robots arrencats, carregar-los-hi el programa seleccionat i a més a més llegir i enviar dades al robot (com ara canviar la velocitat, llegir paràmetres a temps real, etc).

D'aquesta idea neix el projecte en que l'objectiu és crear una pantalla HMI en un ordinador extern, i arrencar 4 robots KR16 de KUKA en mode automàtic extern. Aquesta aplicació ens ha de permetre carregar els diversos programes associats a cada robot, indicant el número d'aquest (Tots els programes disponibles per a carregar als robots han estat programats per mi mateixa), així com la modificació d'alguns paràmetres de robot.

Però la tasca no acaba amb la programació de la interfície, cal posar-ho tot en marxa i que funcioni. Cal establi els enllaços de les comunicacions emprant OPC, cal arrancar els robots i passar-los-hi les comandes a través del ProConOS i Soft PLC (el PLC virtual que incorporen els robots i que ens permet fer la associació d'entrades i sortides sense necessitat d'un PLC físic).

Per a assolir aquests objectius cal conèixer els protocols d'enviament de senyals per a posar en marxa els robots de forma externa i aprendre a utilitzar també el programa HMI.

Cal a més a més, manipular els robots de manera que al seu arxiu cell.src tinguin els enllaços a les diferents tasques i que les reconeguin quan se'ls hi enviï a través de la interfície per OPC, així com programar les diferents tasques que aquests podran desenvolupar.

Paraules clau (màxim 10):

Robot	Industrial	HMI	KUKA
KR16	programació		

ÍNDEX DE CONTINGUTS:

1. INTRODUCCIÓ

Una de les empreses més importants avui en dia dins el sector de la robòtica industrial a nivell mundial, és sens dubte KUKA. Que, ara per ara, és la empresa més important a nivell europeu i la tercera al món. Les principals característiques dels robots KUKA són la seva potència i la seva versatilitat a l'hora de programar els seus robots, donat que, el seu llenguatge de programació s'acosta al C o a pascal i ens ofereix infinites possibilitats en quant a donar aplicació a un robot industrial.
Tenim la sort de tenir la seu peninsular a Vilanova i la Geltrú i això ens permet als alumnes de la EPSEVG, en la fase final de la carrera, tenir la oportunitat de conèixer, treballar i aprendre sobre un dels aspectes claus en els processos industrials i sobretot dins el camp de l'automàtica.
Trobo interessant, doncs, fer un petit repàs de la història de la marca i d'aquesta manera entendre una mica la seva mentalitat empresarial.

1.1 HISTÒRIA DE KUKA

1898 ANY DE FUNDACIÓ

En Johann Josef Keller i en JakobKnappich funden en Augsburg-Oberhausen (Alemanya) una fàbrica d'acetilè amb la finalitat produir, a un preu econòmic, dispositius d'il·luminació per a habitatges i espais urbans, així com aparells domèstics i fars per automòbils.
Les inicials de la raó social "Keller und Knappich Augsburg" van donar lloc posteriorment al nom de KUKA.

1900 FABRICACIÓ D'EQUIPAMENT

La elaboració d'equipament d'acetilè a la fàbrica d'en Keller&Knappich en Oberhausen creix i s'expandeix, empleant a 30 treballadors. La empresa aconsegueix obrir amb èxit nous mercats.

1918 KUKA REPRÈN L'ACTIVITAT EMPRESARIAL DESPRÈS DE LA PRIMERA GUERRA MUNDIAL

Keller und Knappich reprèn l'activitat empresarial amb set treballadors. Amb gran èxit, fabrica torns de seguritat, torns manuals i torns motoritzats amb regulador automàtic de frenada, sistema "Paul" (DRP).

1936 TÉCNICA DE SOLDADURA POR RESISTÈNCIAELÈCTRICA

La tècnica de soldadura autògena ha deixat de ser rentable i s'abandona en aquest any. KUKA es passa a la soldadura per resistència elèctrica i construeix la primera pinça de soldadura per punts d'Alemanya.

1939 MÉS DE 1.000 EMPLEATS

Els negocis de la fàbrica de maquinària d'Augsburg KUKA funcionen en tots els aspectes. La direcció de l'empresa s'enorgulleix de posseir una plantilla de 1.000 treballadors i registrar, per primera vegada, un volum total de vendes d' aproximadament 10 milions de marcs.

1945 UN NOU INICI AMB LA FABRICACIÓ DE MÀQUINES SOLDADORES I ALTRES EQUIPS PETITS

Després de certes negociacions complexes, KUKA obté permís pera tornar a construir màquines soldadores i altres equips petits. Les limitacions d'espai impedeixen reprendre la fabricació de vehicles municipals.

1948 KUKA ESTÉN LA SEVA ACTIVITAT A NOUS PRODUCTES

Per tal de dependre en menor mesura de les màquines soldadores i de la fabricació de vehicles municipals, la empresa busca altres camps d'activitat. Comença a construir tricotoses circulars de doble cilindre per a la firma Hilscher. També, inclouen la producció de màquines d'escriure planes i alimentadors per màquines impressores.

1966 LÍDER AL MERCAT EUROPEU DE VEHICLES MUNICIPALS

KUKA és el major fabricant de vehicles municipals d'Europa. En Alemanya, la empresa posseeix en aquest sector un 50 per cent del mercat. A nivell mundial, els vehicles de KUKA són els líders en tasques municipals de rebuig i neteja, però també en la indústria, el comerç i els negocis.

1969 KUKA ADQUIREIX ARO ALEMANYA

Gràcies a l'adquisició de un 50% de la companyia ARO Schweißmaschinen GmbH& Co. KG, ubicada en Neussi especialitzada en soldadura, la fàbrica de maquinària Keller&KnappichGmbHtorna a reforçar-se en aquesta àrea tecnològica.

1970 FUSIÓ DE KUKA I IWK

Keller & Knappich GmbH e Industrie-WerkeKarlsruhe AG, ambdues empreses pertinents al grup Quandt, es fusionen convertint-se en Industrie-Werke Karlsruhe Augsburg Aktiengesellschaft, coneguda de forma abreviada com a IWKA AG.

KUKA aporta a la construcció de vehicles municipals el seu lideratge dins del mercat i també com a desenvolupador i productor de màquines de soldadura. La nova IWKA AG també desenvoluparà activitats dins els sectors de les màquines embaladores, la tecnologia tèxtil, la conformació de metalls i les màquines eina.

1971 EL PRIMER TREN DE TRANSFERÈNCIA PER SOLDADURA D'EUROPA EQUIPATAMB ROBOTS

La instal·lació entregada a Daimler-Benz AG pera la elaboració de panells laterals marca, pel sector de la soldadura en KUKA, l'inici d'una nova etapa. En aquesta primera instal·lació robotitzada per a la producció de panells laterals dins d'una planta de Daimler-Benz, s'empren robots de cinc eixos de la marca nord-americana Unimation.

1973 EL PRIMER ROBOT DE KUKA

L'amplia experiència tecnològica de KUKA, unida a la necessitat de la indústria automotriu de disposar de robots potents i fiables, duu finalment a aquesta empresa a desenvolupar un robot industrial propi. El producte, presentat amb el nom de FAMULUS, és el primer robot amb sis eixos accionats electromagnèticament.

1976 EL IR 6/ 60

KUKA es decideix pel desenvolupament d'un tipus de robot completament nou: el IR 6/60. Aquest robot posseeix sis eixos accionats electromecànicament i està equipat amb una mà angular. La unitat de control està fabricada per Siemens.

1981 LA EXPANSIÓ MUNDIAL S'ACONSEGUEIX AMBÈXIT

Reorganització en la sucursal d'Augsburg.
Les activitats de KUKA s'agrupen en tres empreses independents:KUKA Schweissanlagen + Roboter GmbH, KUKA Umwelttechnik GmbHi KUKA Wehrtechnik GmbH. El número d'empleats ascén a aproximadament 2.000.
Per a les activitats internacionals de KUKA Schweißanlagen + Roboter GmbH, els punts de suport a l'estranger cobren una importància cada cop major. Per exemple a l'estat de Michigan (Estats Units), prop de la metròpoli del automòbil, Detroit, es funda la empresa KUKA Welding Systems + Robot Corp..

1983, es funden les empreses KUKA Automatisme + Robotique S.ä.r.l. en Francia i KUKA Sistemi di Saldatura + Robot s.r.l. en Italia.

1984, es funda KUKA Sistemas de Automatización S.A. en España (Actualment KUKA Robots Ibérica, S.A.)

1985, s'inauguren altres punts de suport: en Bèlgica, KUKA Automatisering + Robots N.V, i en 1987 a Suècia, com a KUKA Svetsanläggningar + Robotar AB.

1989 NOUS ROBOTS AL CAMÍ A L'ÈXIT

Una nova generació de robots industrials, presentada per primera vegada a la fira Hannover '89 produeix una gran expectació a nivell nacional i internacional. Equipats amb nous motors d'accionament sense escombretes, aquests productes destaquen per requerir molt poc manteniment i per la seva alta disponibilitat tècnica. Segons el seu tamany, poden manipular càrregues de entre 8 i 240 kg.

1995 KUKA CONQUISTA EL MERCAT D'ESTATS UNITS

KUKA Welding Systems + Robot Corporation, ubicada prop de la ciutat automotriu per excel·lència d'EEUU, Detroit, expandeix les seves activitats especialment cap a l'àmbit de la construcció de sistemes.

1996 KUKA ROBOTER GMBH ESDEVÉ UNA EMPRESA INDEPENDENT

L'1 de gener de 1996, KUKA Schweissanlagen + RoboterGmbH es subdivideix en dos societats que des d'aquell moment operaran de manera autònoma dins el mercat: KUKA RoboterGmbH y KUKA SchweissanlagenGmbH.
KUKA RoboterGmbH comença amb aproximadament 250 empleats i assoleix unes xifres de producció de 4.000 unitats.

LA REVOLUCIÓROBÒTICAPROCEDENT D'AUGSBURG

A la feria tecnològica anual de Hannover, KUKA RoboterGmbH presenta sensacionals novetats.
Una unitat de control basada en PC i una guia d'usuari per a Windows comprensible per qualsevol operador. El més destacat d'aquests avenços és un aparell de maneig per activitats de control i programació que duu integrat un ratolí 6D.

1997 DIN EN ISO 9001

KUKA RoboterGmbH rep la certificació DIN EN ISO 9001 sense cap tipus de restricció.

1998 PREMI DE DISSENY "IF"

Els robots de KUKA ocupen indiscutiblement la primera posició del mercat alemany, la segona del mercat europeu i la tercera del mercat mundial. En aquest any, KUKA fabrica 5.000 robots, la qual cosa implica una quintuplicació de la seva producció de robots des de l'any 1993.
Coincidint puntualment amb el centèsim aniversari de les companyies KUKA, els especialistes en robòtica d'Augsburg presenten nous robots industrials amb una capacitat i rendiment encara més grans.
A més a més, el seu disseny destaca també per una alta qualitat estètica, tal i com demostra el premio de disseny "if 1998" obtingut pel KR 60/100P.

1999 UNA SENSACIÓ EN TOT EL MÓN: EL PRIMER TELEDIAGNÒSTIC DE ROBOTS VIA INTERNET

KUKA RoboterGmbH presenta el primer telediagnòstic per a robots via Internet del món.

2002ELS ROBOTS DE KUKA CONQUISTEN NOUS SECTORS I APAREIXEN AMB JAMES BOND 007

KUKA conquista nous mercats gràcies als seus innovadors productes i sistemes. Amb nombroses solucions pels sectors més variats, KUKA demostra les amplies possibilitats d'aplicació dels seus robots.
Com a primer fabricant de robòtica, KUKA estableix un estret contacte entre l'ésser humà i el robot: Amb el Robocoaster, els passatgers donen voltes a l'aire amb el robot: un entreteniment extraordinari per a parcs d'atraccions i altres events.
En la pel·lícula de James Bond " Die anotherday – Otro día para morir" els robots de KUKA fan la seva aparició estel·lar al costat de Pierce Brosnan i Halle Berry.

2005 KUKA SAFE ROBOT

Quan cadascú fa allò que millor sap fer, coordinant-ho amb les aptituds de l'altre, s'arriba al millor resultat. Això devien pensar també els tècnics de desenvolupament de KUKA RoboterGmbH que creen el KUKA Safe Robot una tecnologia interdisciplinària.

2007 "TITAN": EL ROBOT MÉS FORT DEL MÓN

El robot "titan" de KUKA no tem batre's amb els més forts. Amb una capacitat de càrrega de 1000 kilograms i un abast de 3200 mm és el robot industrial de 6 eixos més grani fort a nivell mundial. Pera KUKA Roboter GmbH això suposa no només una mostra més de la seva capacitat innovadora, sinó a més a més, la entrada al Llibre Guinness dels Rècords

2009 GAMA DE ROBOTS DE PALETITZAT MÉS AMPLIA DEL MÓN

Amb l'aparició al mercat dels nostres tres robots de paletitzat KR 300 PA, KR 470 PA i KR 700 PA, KUKA completa la seva gama de productes convertint-la probablement en la més versàtil del món. Paletitzar, despaletitzar, elevar, apilar, embalar, transportar, classificar i etiquetar. Per a cada tasca i rang de càrrega disposem, a KUKA, del robot de paletitzat adient.

2010 LA NOVA GENERACIÓ DE PRODUCTES DE KUKA

La nova sèrie de robots QUANTEC és la primera i única gama de robots que cobreix completament el rang de capacitat de càrrega de 90 a 300 kg amb abasts de fins a 3.100 mm.
El nou sistema de control KR C4 s'integra també per primera vegada el control complet de la seguretat. Això permet solucionar totes les tasques alhora.
El nou dispositiu de maneigsmartPAD i la nova plataforma WorkVisual faciliten la manipulació la programació.

Ara que ja hem vist de forma resumida de on prové la empresa i quins són els passos que els han dut a ser el que són avui en dia, ens podem dedicar de ple a l'enteniment del projecte que tinc entre mans i el plantejament inicial que té aquest. En el proper subcapítol veurem quines són les pretensions del projecte.

1.2 PLANTEJAMENT DEL PROJECTE

La idea d'aquest projecte prové directament dels clients de KUKA. Com ja he comentat al inici d'aquest primer capítol, la programació del robot és versàtil, lliure i completament oberta. Això vol dir que els programes tant poden ser senzills com extremadament llargs i complexes. No obstant, sovint els programes no els crea l'operari que estarà pendent de controlar que tot marxi correctament, comprovarà els temps de cicle, posarà en marxa el procés o el canviarà per un altre quan sigui convenient.

Aquest operari, probablement no coneix com funciona la programació de Kuka i no és essencial que en conegui els detalls. Tot i així, la pantalla que té per defecte per a interactuar amb el robot mostra l'execució del programa línia a línia i per a fer canvis cal conèixer, si més no, com funciona la KCP (Kuka Control Panel, és la consola de manipulació, més endavant hi dedicarem tot un subcapítol).

Per aquest motiu, molts clients, demanen la possibilitat de disposar d'una pantalla que s'executi a la vegada que el programa, ocultant aquest i donant la possibilitat de conèixer més visualment l'estat del robot, poder modificar algun paràmetre, poder arrencar aquests robots i carregar-ne programes ja fets sense tenir cap coneixement de com funciona la KCP, ni tampoc ser cap expert en el tema, de forma intuïtiva.

Precisament, el fet que KUKA té aquesta versatilitat permet que qualsevol client es pugui crear aquesta interfície, amb programes com ara Visual Basic, Visual C, SCADAs en general... Però aquestes interfícies tenen un problema afegit: Els controladors i les trames de comunicació és possible que s'hagin de programar a part. KUKA, no obstant, té a la disposició de tothom que ho vulgui adquirir, el seu propi programa que fa la interfície HMI, i aquesta conté ja tots els links directes amb les dades i les accions del robot. El programa de KUKA s'anomena *HMI Studio* i es comunica mitjançant OPC.

Donat que els clients demanen la possibilitat que hi hagi programes que facin de base en HMI per a tapar l'execució del programa i alhora fer més senzilla la manipulació es va proposar des de l'empresa la creació d'un programa fent servir, òbviament, HMI Studio per a arrencar diversos robots KUKA, carregar-los-hi un programa, llegir-ne l'estat i enviar algunes senyals de control útils pels operaris inexperts.

Aquest projecte estarà format,doncs, per tres parts. Una primera part de disseny i programació de la interfície HMI amb el software de KUKA, una segona part de comunicació amb els robots i programació d'entrades/sortides i finalment una darrera part de programació dels robots (de forma que efectuïn la tasca preprogramada seleccionada). En paral·lel a aquestes parts cal haver realitzat l'ajustament dels robots, calculat els TCP de les eines i, quan convingui, determinar bases noves.

La part de disseny de la interfície es farà mitjançant HMI Studio. La part de comunicació fent servir OPC server de KUKA i Soft PLC i finalment, la darrera part de programació dels diferents robots anirà separada de les dues primeres parts (aquesta part és important de cara a tenir programes per a fer proves i aprendre més sobre el funcionament dels robots).

KUKA Robots Ibèrica disposa d'una sala robotitzada de formació (on imparteixen cursos de programació als clients que ho contractin). Aquesta sala està a la meva disposició per a que realitzi el projecte i incorpora 4robots KR16, amb els seus respectius armaris i les seves KCP.

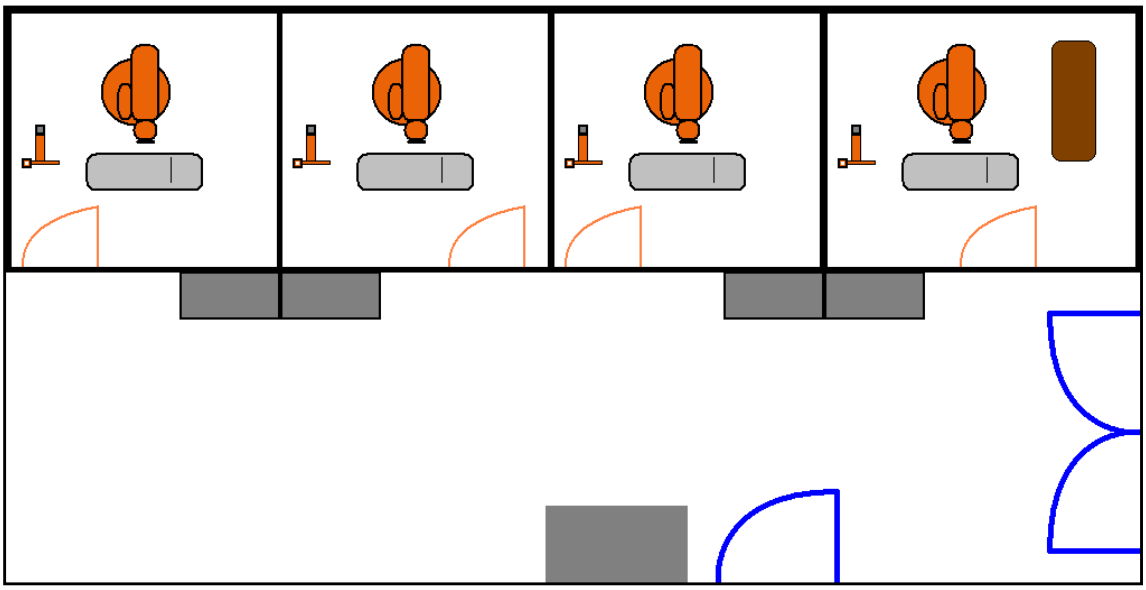


Fig. 1.1 – Sala de formació a KUKA Robots Ibérica, S.A.

Com podem veure a la figura 1.1 la sala de formació té tota la part de procés necessari per a la realització del projecte. A part, cal un ordinador extern on instal·lar el HMI Studio. Els robots tenen incorporada com a element terminal (ET) o eina, una pinça pneumàtica i manipulen peces principalment quadrades. Totes les celes inclouen un dispensador de peces quadrades i una taula de treball fixa.

1.3 PLANTEJAMENT DELS OBJECTIUS

Per a la realització d'aquest projecte s'han estipulat des de la empresa uns objectius principals a assolir:

1. Arrancar en Automàtic Extern diversos robots industrials mitjançant una interfície d'usuari en un PC.

2. Visualitzar per pantalla informació de l'estat dels robots com velocitat, posició actual, número de programa... (enviar i rebre diferents tipus de dades).

No obstant, a part dels dos objectius principals han aparegut altres objectius que també són importants de cara a la correcta realització del projecte i que cal esmenar:

A. Dissenyar la pantalla de forma intuïtiva, incloent detalls a nivell d'ajuda, que no carreguin la vista i que no saturin d'informació a l'operari.

B. Fer la programació dels robots de la sala amb, com a mínim, dos programes cadascun.

2. DESCRIPCIÓ DELS ELEMENTS EMPRATS

Com és d'esperar operar amb un robot industrial requereix d'una cela on el robot es trobi aïllat així com una sèrie de dispositius que ens permetin fer el control i la programació d'aquest robot i tot una sèrie d'elements que cal tenir en compte a l'hora de treballar.

Donat que és important conèixer els elements dels que disposem en la elaboració d'aquest treball i cal descriure alhora la configuració de la sala en que es troba el muntatge de les celes dels robots, tot seguit trobarem una sèrie de subcapítols amb la descripció dels dispositius necessaris en la implementació d'aquest projecte..

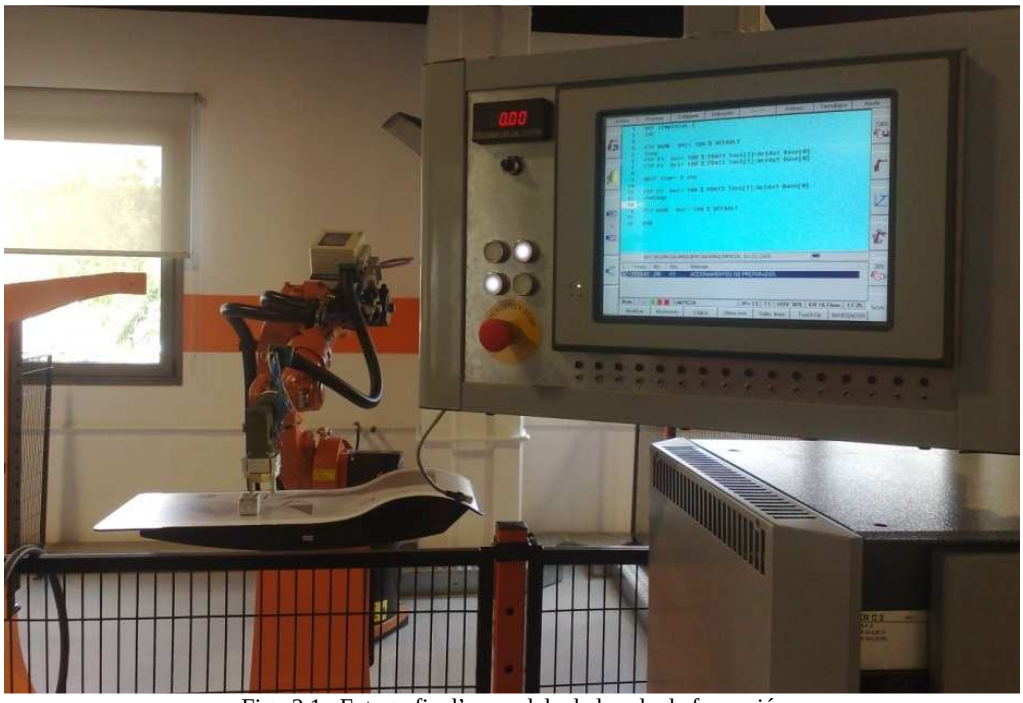


Fig. 2.1 –Fotografia d'una cel·la de la sala de formació

2.1 EL CONJUNT

Abans d'entrar en detall en les característiques de tots els elements, cal tenir una visió de conjunt del que és realment necessari dins d'una cel·la de treball. En aquest subcapítol veurem clarament quins són els elements i quines les possibilitats.
A continuació trobem la figura Fig. 2.2 en que visualitzem un conjunt d'elements que poden conformar una cel·la de treball d'un robot industrial. Alguns d'aquests elements són realment indispensables a l'hora de funcionar, mentre que, hi ha elements que no són sempre imprescindibles alhora de treballar amb múltiples aplicacions (Recordem que les aplicacions ens determinaran el tipus de cel·la que necessitem en cada cas així com l'element terminal/eina a utilitzar).

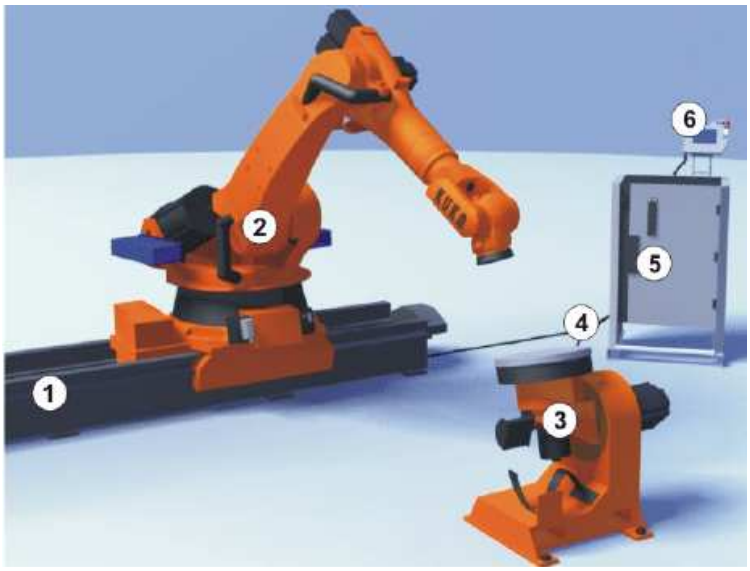


Fig. 2.2 – Elements que formen part d'una cel·la

Si mirem la figura Fig. 2.2 veurem que hi ha 6 elements numerats. En aquest muntatge encara faltaria un 7è element que no apareix a la figura (l'element terminal o eina) que aniria acoblat al plat del sisè eix (A6) del robot. D'aquests 6 elements tan sols quatre són imprescindibles per a qualsevol cela robotitzada de KUKA (els elements 2, 5, 4 i 6) i els altres dos elements (1 i 3) seran necessaris segons la aplicació i les necessitats de la tasca. Els elements són els següents:

1: Unitat linear (Eix extern al robot, no ha de ser necessàriament de KUKA, ni tampoc cal que sigui un eix alternatiu linear). Tanmateix qualsevol eix addicional pot ser emprat acoblat matemàticament al robot o de forma completament externa (en el segon cas, no es tindria en compte com a part del robot).

2: Robot (KUKA Robot → KR). Imprescindible disposar de robot en una cel·la robotitzada. Considerarem que el robot té com a mínim una eina de treball útil per a la realització de la tasca. Hi ha dedicat tot un capítol a parlar del Robot (apartat 2.2).

3: Posicionador (No cal que la taula o posicionador sigui de KUKA). En alguns casos em lloc de tenir un posicionador podem arribar a tenir una eina externa on el robot tractaria de portar la peça o producte. Com en el cas de la unitat linear es pot considerar com a eixos addicionals si s'acobla matemàticament al robot.

4: Cables de connexió, alimentació, etc. Per a qualsevol aplicació en que tingui lloc un robot és necessari tot un cablejat que ens alimenti el robot així com l'eina acoblada al sisè eix i que hi dugui el control i les dades de sensors, càlculs, etc. Les comunicacions són imprescindibles en qualsevol cel·la robotitzada i encara tenen lloc mitjançant cablejat.

5: Armari de control del robot (KUKA Robot Controller→ KRC). L'armari s'encarrega de la part del control dels eixos dels robots, inclou el PC, inclou l'alimentació, inclou els servos, etc. Sense l'armari de control el robot no es posa en marxa ni pot funcionar. Hi ha dedicat tot un capítol a parlar del armari (apartat 2.3).

6: Consola de interacció usuari-robot (KUKA Control Panel→KCP). Important per a tenir un control sobre l'Armari i sobre el robot. És la interfície que uneix amb l'operari. Hi ha dedicat tot un capítol a parlar de la KCP (apartat 2.4).

En el meu cas veurem que tan sols es fa servir el robot (KR 16), l'armari de control (KR C2 ed05), la consola d'interacció (KCP 2 Std ed05) i els cables de connexió (donat que així és com està preparada la sala de formació) com es mostra a la figura Fig. 2.3.

Si bé ja he definit que és allò que es necessita en una cela robotitzada, val a concretar a continuació com són les celes robotitzades amb les quals treballarem en aquest projecte en concret.

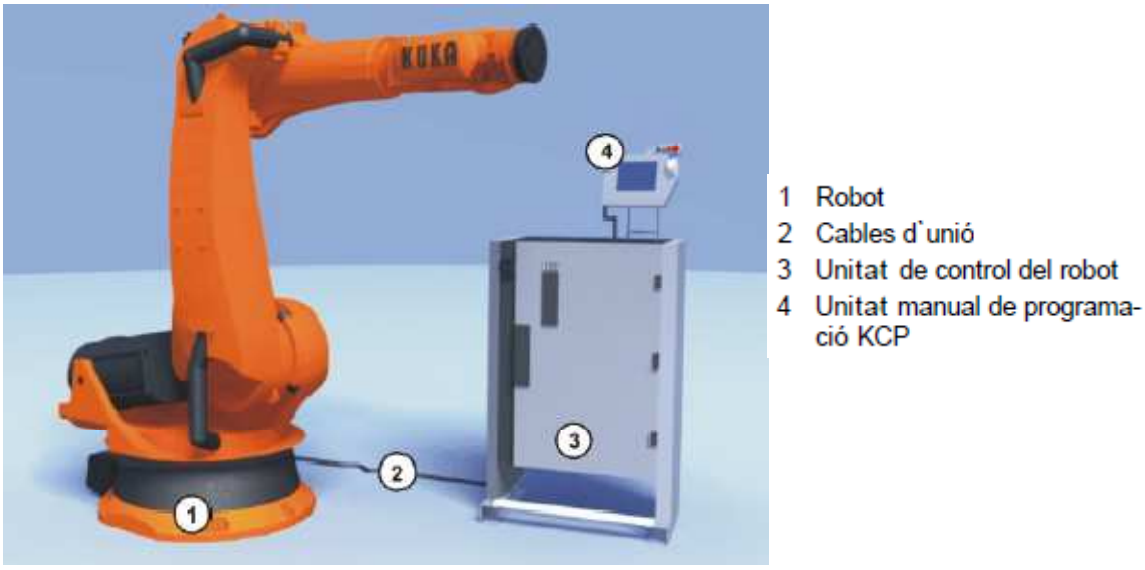


Fig. 2.3 - Sistema d'elements amb els que es treballarà.

Jo he treballat en les instal·lacions de la sala de formació robotitzada de KUKA Robots Ibérica, S.A., és a dir, que estic subjecte a les característiques concretes de les cel·les que s'hi troben. La sala compta amb 4 cel·les separades per les seves gàbies de seguretat, on en cada cel·la, es troba un robot. Els armaris de control queden fora del tancat de seguretat de forma que es pot arrancar l'armari sense haver d'envair l'espai de treball del robot (Per a més informació veure el subcapítol 1.2. Plantejament del projecte).

Com a elements bàsics disposarem del robot, l'armari i la KCP, però a més a més, també farem servir un ordinador a part que serà l'encarregat de comunicar-se amb els robots i carregar-hi els programes. Tot això estarà communicat per OPC mitjançant el OPC que incorpora HMI Studio i el ProConOS (OPC Server) amb Soft-PLC al robot.

A més a més, cal tenir els programes als robots i modificar el programa cell.src que es posa en marxa quan a un robot se li executa una posta en marxa mitjançant automàtic extern. D'aquesta manera es determinarà les tasques a realitzar per el robot en cada moment.

2.2 EL ROBOT (KR16)

Com ja he comentat anteriorment és disposen de 4 robots de tipus KR 16. Sempre que llegim les dades del robot val a dir que cal llegir-ho de la següent manera:

KR	XX	/	YY	Lletra/es
KUKA Robot	Càrrega màxima en el centre del plat del eix 6	(opcional)	(opcional) Elongació del braç -> abast	Aporta informació extra

Per exemple:

KR 250/120 HA → Robot KUKA amb càrrega màxima de 250Kg i amb elongació fins a 120mm, HA indica High Accuracy (alta precisió).

A la sala de formació pràctica trobem els KR 16, és a dir, Robots Kuka amb càrrega màxima de 16Kg en el centre del plat del eix 6 (A6).

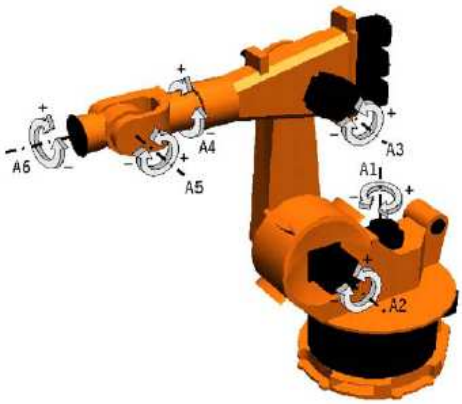


Fig. 2.4 – Eixos robot KR 16

El KR 16 és un robot de 6 eixos (A1, A2, A3, A4, A5 i A6) situats com especifica la figura Fig. 2.4. Tots ells treballen respecte incrementant-se/decrementant-se respecte del punt d'ajust. Un robot desajustat no funcionarà degudament (veure subapartat 3.2.1. per a més informació).Les característiques del robot KR 16 són:

KR 16

CÀRREGUES:	
Càrrega	16 Kg
Càrrega addicional	10 Kg
ZONA DE TREBALL:	
Abast màxim	1610 mm
ALTRES DADES:	
Número d'eixos	6
Repetibilitat	< ±0,05 mm
Pes	235 Kg
Posicions de muntatge	Terra, Sostre
Unitats de control	KR C2

Tots els robots tenen un número de sèrie que els identifica com a únics i que permet fer un seguiment de la vida que duu el robot. El número de sèrie del robot així com molts altres paràmetres que resulten útils es troben a la targeta, placa o adhesiu que els robots incorporen de fàbrica. En la figura Fig. 2.5 veurem on podem trobar la informació (indica tipus robot, número de sèrie, any de fabricació, número de planificació i pes):

Fig. 2.5 – Situació placa de robot

Tots els robots, no importa de quina casa siguin, tenen uns singularitats característiques. Aquestes singularitats s'han d'evitar donat que son posicions en les que el robot té la impossibilitat de fer el càlcul de la trajectòria, degut a les característiques intrínseques de la posició i la equació matemàtica que efectua el càlcul. KUKA per la seva banda té 3 tipus de singularitats diferents:

Singularitat 1: Singularitat superior (posició α1). Aquí el punt de trajectòria del canell (es localitza en la intersecció dels eixos A4, A5 i A6) queda posicionat directament a l'eix 1 (A1).

Singularitat 2: Singularitat extensió (posició α2). En aquesta posició la extensió dels eixos 2 i 3 (A2 i A3) intersecciona amb el punt de la trajectòria del canell.

Singularitat 3: Singularitat dels eixos del canell (posició α5). En aquest cas, els eixos 4 i 6 (A4 i A6) estan paral·lels. Amb aquesta configuració no és possible determinar les posicions de aquests dos eixos de manera no ambigua per mitjà d'una transformació inversa, ja que, existeixen un número infinit de valors per als eixos A4 i A6. Aquesta és, habitualment, la singularitat més comú i per tant, la que cal tenir present més sovint.

2.2.1 Ajust del robot

Tot robot necessita estar ajustat a una posició. Quan el robot està masteritzat o ajustat, els eixos es mouen cap a una posició mecànica definida, anomenada posició de zero mecànic.

Un cop que el robot es trobi en la posició de zero mecànic (figura 2.5.), el valor absolut de l'eix s'ha de emmagatzemar.

Un robot desajustat, tot i que només ho estigui en un sol eix no es podrà moure emprant coordenades cartesianes, tampoc es podrà moure executant un programa ni tampoc reconeixerà els límits de software. Així doncs caldrà tenir els robots ajustats de cara al correcte funcionament de tots ells.

La posició d'ajust o masteritzat que té un KR 16 és de A1=0°, A2=-90°, A3=+90°, A4=0°, A5=0° i A6=0°.

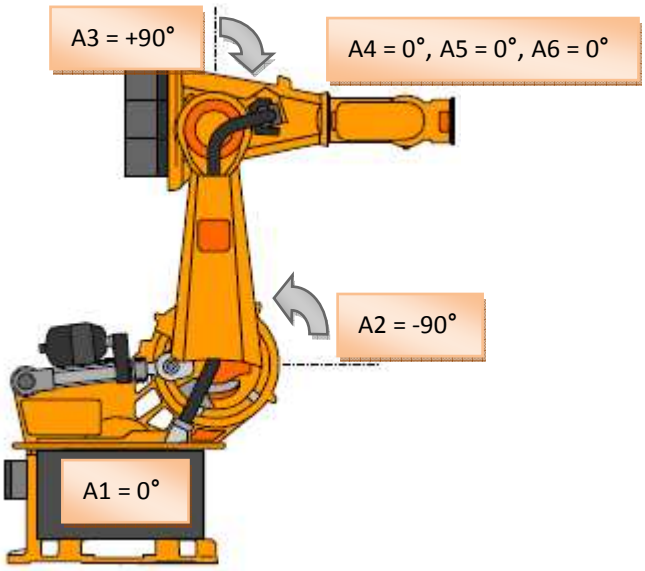


Fig. 2.6 – Posició zero mecànic

Per a ajustar els robots cal portar-los a la posició mecànica d'ajust com bé he comentat abans, i per a fer això cal efectuar l'ajust sempre sota les mateixes condicions de temperatura (per tal d'evitar inexactituds degudes a dilatacions). Això significa que la masterització s'ha de realitzar sempre amb el robot en un estat de servei fred o amb temperatura en règim.

Per a ser precisos cal dur el robot a la posició mecànica correcta. Per això cal que desplacem el robot primer a la posició de preajustament (com podem observar a la imatge de la figura Fig. 2.7 cal moure els eixos de manera que les marques blanques coincideixin) i després obrir el tap on es situa el patró d'ajustament (primera imatge de la figura Fig. 2.8).

Fig. 2.7 – posició de preajustament

Fig. 2.8 – Ajustament amb rellotge comparador

El que trobem és el patró d'ajust i caldrà un rellotge comparador o una unitat electrònica d'ajust (UEA) per a posar el robot al seu zero mecànic. Introduïm el rellotge comparador o bé la UEA pel forat com es mostra a la segona imatge de la figura Fig. 2.8.

Ara cal moure el robot cap a menys (és a dir, en direcció negativa axialment). Com podem observar a la figura següent (Fig. 2.9) aquest moviment ens permet buscar el punt exacte on hi ha la osca (tant manualment amb el rellotge comparador o automàticament endollant-hi la UEA).

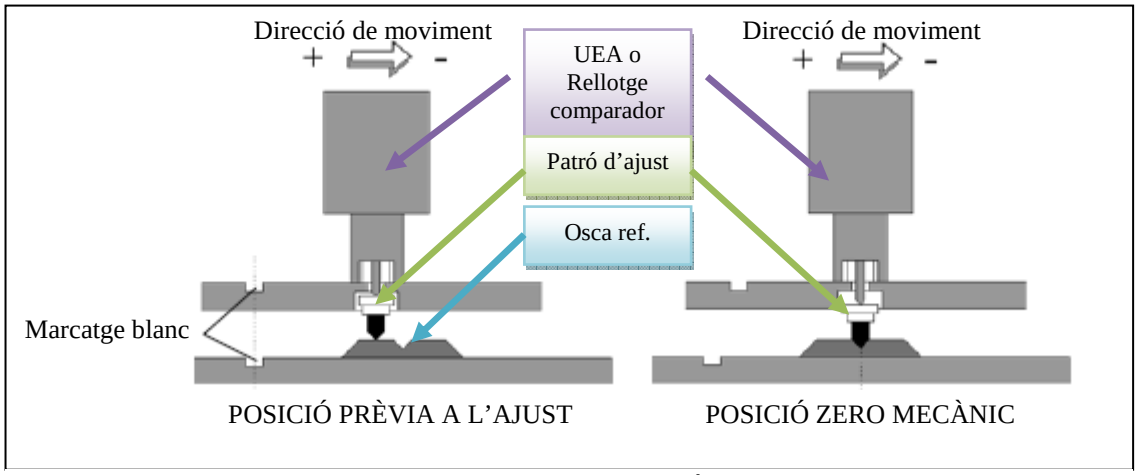


Fig. 2.9 – Ajust a zero mecànic

Un cop trobat el punt mecànic zero de l'eix es guarda al robot amb AJUSTAR. Ja tindrem l'eix ajustat i ara només cal repetir el procés per a cada eix de forma que tot quedi masteritzat.

2.2.2 Càlcul de TCP offset i bases

A cada eix del robot hi trobem integrat un "resolver" que capten en cada instant l'angle de la posició de l'eix. Conjuntament amb les distàncies conegudes existents entre els eixos del robot, la unitat de control pot, gràcies a la geometria, calcular la posició i la orientació en l'espai del punt central del plat del sisè eix (o brida).

La posició del punt central de la brida es descriu per la seva distància al punt d'origen del sistema de coordenades universals (a la figura Fig. 2.10 la línia de punts). Aquesta distància es descomposa en les seves components X, Y i Z.

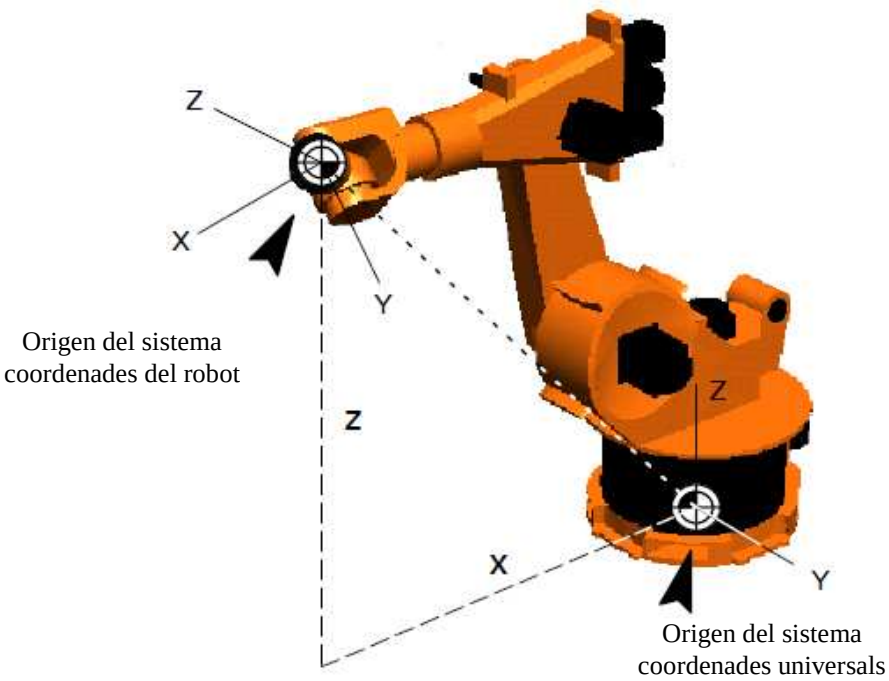
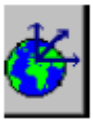


Fig. 2.10 coordenades del robot

La orientació del sistema de coordenades de la brida del robot, origen del qual es troba en el punt central de la brida, es descriu pel seu gir (en angles Euler de Z, Y i X) respecte al sistema de coordenades universals (Veure figura Fig. 2.10).

El robot es mou depenent del sistema de coordenades de referència seleccionat. Això té com a conseqüència que el robot es mogui, per exemple, diferent el sistema de coordenades de la eina (TOOL) que en amb sistema de coordenades base (BASE):

	**Sistema de coordenades específiques del eix** Cada eix del robot pot moure's de forma individual, en direcció positiva o negativa
	**Sistema de coordenades WORLD** Un sistema de coordenades rectangulars de localització fixa amb l'origen situat a la base del robot.
	**Sistema de coordenades BASE** Sistema de coordenades rectangulars en que l'origen es troba en una peça.
	**Sistema de coordenades TOOL** Sistema de coordenades rectangulars en que l'origen es troba en la eina (acoblada al eix 6).

Quan decidim moure'ns emprant el sistema de coordenades TOOL i no tenim definida cap eina, el robot interpreta que no té res instal·lat a la brida i es mou amb coordenades de brida. Alhora si el robot té una eina o element terminal incorporat a la brida és importantíssim indicar al robot que es mogui respecte a aquesta extensió. Tampoc són iguals tots els tipus d'eines com veurem en l'exemple de la figura Fig. 2.11. Com podem observar en la figura tenim els casos de tenir eines diferents o no tenir-ne cap. El sistema de coordenades ens convé tenir-ho distribuït de forma diferent segons el cas en que ens trobem.

Fig. 2.11 – Diferents tipus de sistemes de coordenades TOOL

Com podem observar quan no tenim eina l'eix Z de TOOL coincideix amb la Z de l'eix 6. Però quan indiquem un TCP Offset per a calcular l'eina, la X es considera en la orientació d'avanç.

Tampoc no és el mateix definir els punts emprant fent el càlcul segons el sistema de coordenades, caldrà fer el càlcul de la base sobre la peça:

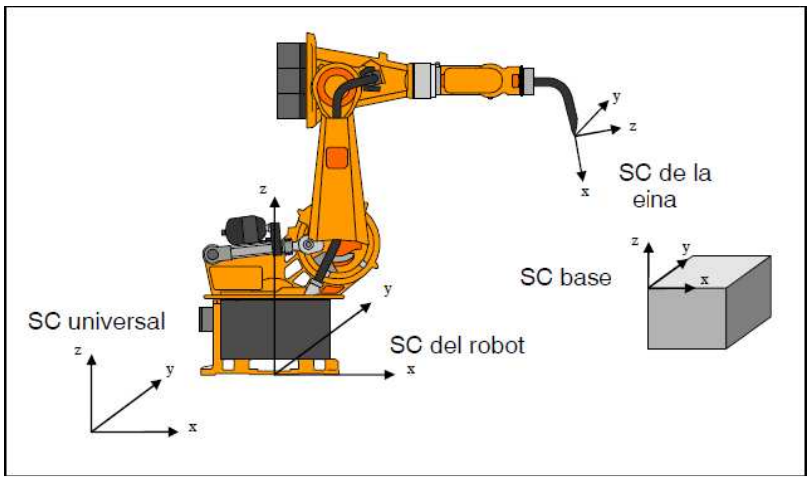


Fig. 2.12 - Representació dels sistemes de coordenades.

- Càlcul del nou TCP

Per a poder calcular la posició d'un punt de referència d'una eina muntada sobre la brida del robot o una peça, la unitat de control del robot ha de conèixer la posició i orientació d'aquests punts respecte al sistema de coordenades de la brida del robot (brida en buit→ TCP Tool Centre Point). Aquestes dades poden determinar-se per mitjà d'elements de mesura auxiliars externs. Si les dades es registren en una plantilla o formulari poden es poden indicar en tot moment a la unitat de control. Però després d'una col·lisió, aquestes dades són inservibles i s'han de determinar de nou (per exemple si mirem el cas de la figura Fig. 2.13).

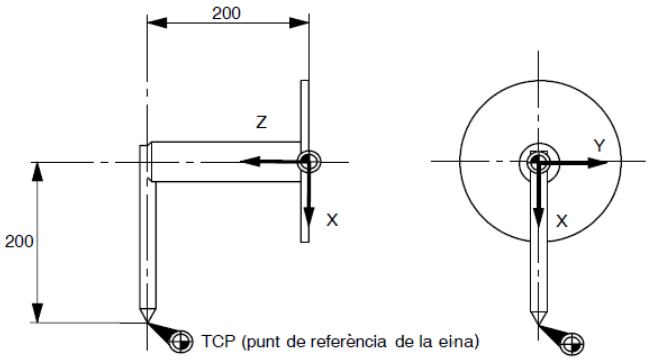


Fig. 2.13 – TCP mitjançant desplaçament

No obstant hi ha altres mètodes per a calcular el TCP Offset. Una altra possibilitat de determinar el TCP de l'eina és acotar aquesta eina mitjançant els sistemes de mesura i les funcions del propi robot. Per a emprar aquest sistema cal seguir una de les diverses opcions que el robot et permet (segons el cas i la eina, s'escull el mètode de càlcul del TCP Offset).

Per a la calibració del TCP de l'eina tenim 2 mètodes (X Y Z – 4 punts i/o X Y Z – Referència). En quant a la calibració de la orientació tenim 3 possibilitats (A B C - World 5D, A B C - World 6D i A B C - 2 punts).

Càlcul	Mètode	Descripció breu
TCP offset	X Y Z – 4 punts	Desplaçament a un punt de referència fix
TCP offset	X Y Z – Referència	Desplaçament amb eina coneguda de referència
Orientació	A B C –World 5D	Posicionament vertical en sistema de coordenades World.
Orientació	A B C –World6D	Posicionament vertical en sistema de coordenades World.
Orientació	A B C – 2 punts	Desplaçament a 2 punts amb dades d'orientació

Mètode	En quins casos està recomanat
A B C –World 5D	Quan només s'utilitza la direcció de treball de la eina per al seu posicionament i guiat (Soldadura Mig-Mag, aplicacions amb làser, tall...).
A B C –World 6D	Quan es necessita la orientació dels 3 eixos de l'eina per al posicionament guiat (pinces de soldar, pistoles d'aplicació de cola, etc).
A B C – 2 punts	Quan per el posicionament és necessari una orientació exacta dels tres eixos de la eina.

En el meu cas ha estat preferible emprar el mètode de *X Y Z – 4 punts* per trobar el TCP offset i després el *A B C – 2 punts* per a donar la orientació. A continuació descriurem breument de que tracten ambdós mètodes.

Si partim de que no tenim cap eina reconeguda en el robot partim en les condicions de la figura Fig. 2.14, encara que estigui instal·lada al plat del eix 6.

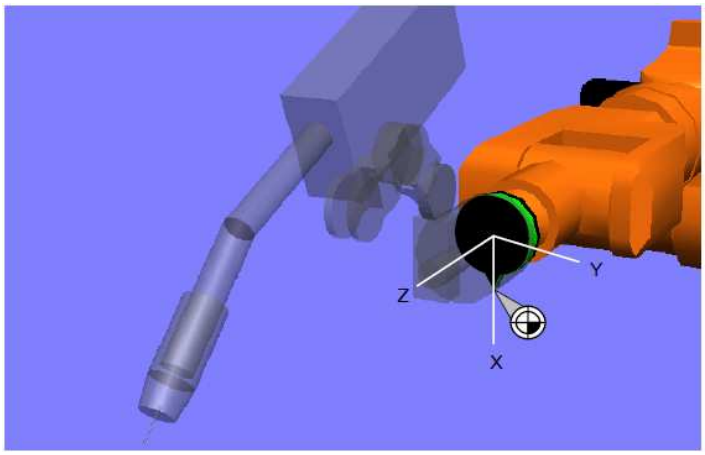


Fig. 2.14 - Robot sense TCP de la eina calculat

X Y Z - 4PUNTS:

En el mètode dels "4 punts" s'ha de desplaçar i posicionar el nou TCP (punt de referència de la eina) sobre un punt de referència, des de quatre posicions diferents (d'aquí li ve el nom). Com més exactes alhora d'arribar al mateix punt de amb diferents posicions més exacte serà el punt (veure figura Fig. 2.15).

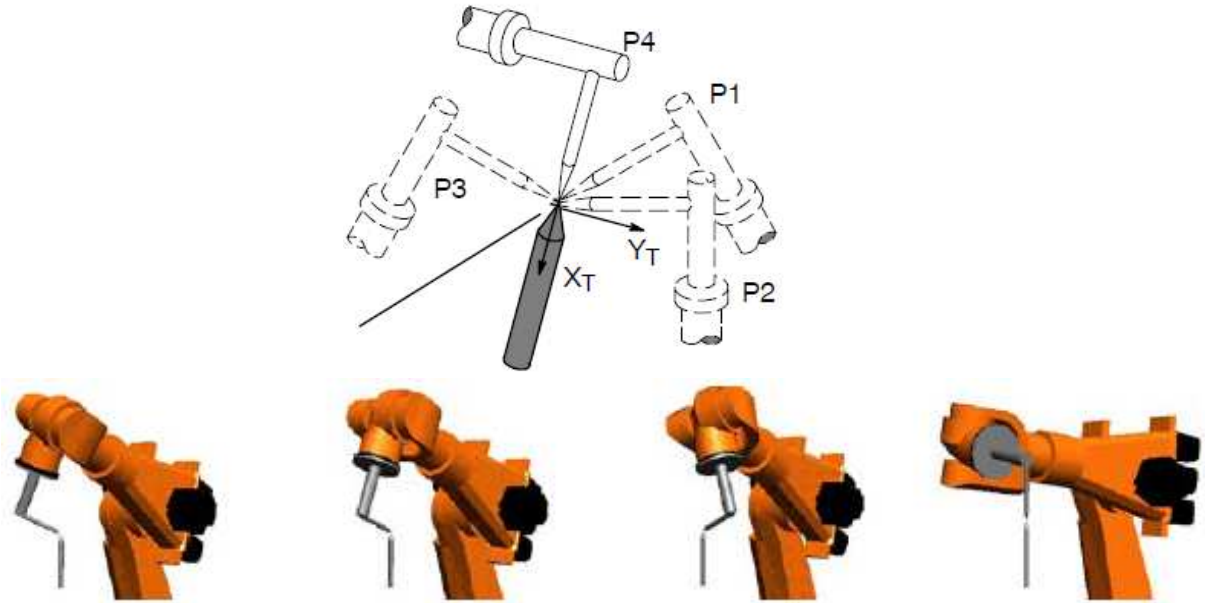


Fig. 2.15 – Mètode XY Z - 4 punts

El robot dóna el paràmetre de aproximació i un paràmetre d'error. Si aquest error supera els 5mm el propi robot et demana que es torni a calcular el punt.

A B C – 2 PUNTS:

Emprant aquest mètode podem orientar el sistema amb aquests passos simples:

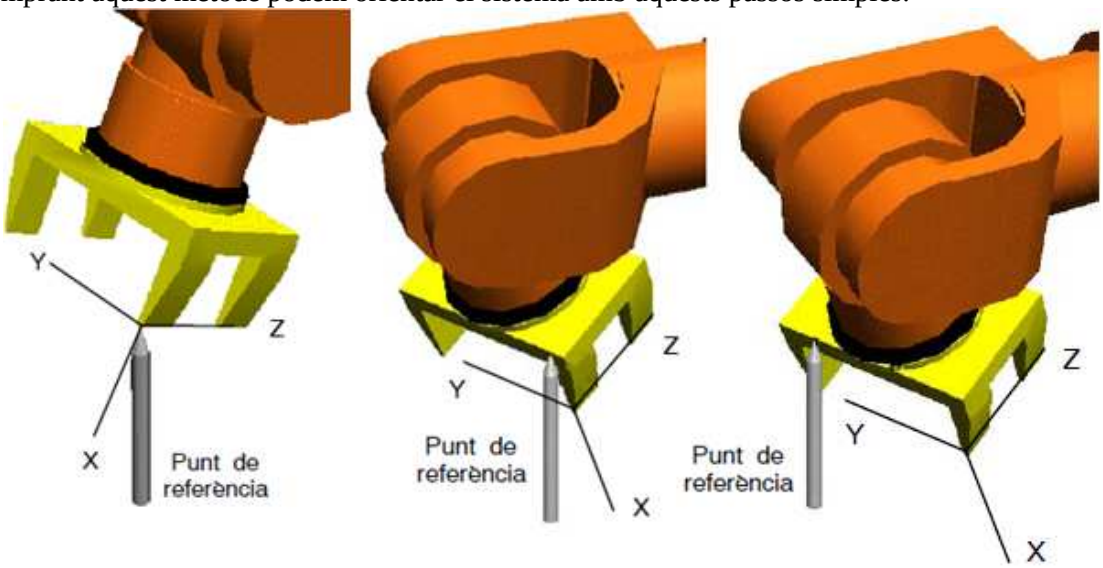


Fig. 2.16 – Mètode A B C - 2punts

1. Indiquem la direcció de treball de la eina. Per fer-ho desplacem la eina amb el seu TCP a un punt de referència qualsevol (primera imatge de la figura Fig. 2.16).

2. Ara cal desplaçar el robot al punt de referència amb un punt sobre l'eina que es trobi sobre el sentit oposat a la direcció de treball(segona imatge de la figura Fig. 2.16).

3. El pla YZ pot rotar encara lliurement al voltant de X (direcció de treball) i es determina amb aquest pas. Es necessari desplaçar la eina de tal manera que el punt de referència amb valor y positiu estigui situat al futur pla XY de la eina(última imatge de la figura Fig. 2.16).

Finalment ens queda la eina amb tots els eixos i la orientació adients (figura Fig. 2.17):

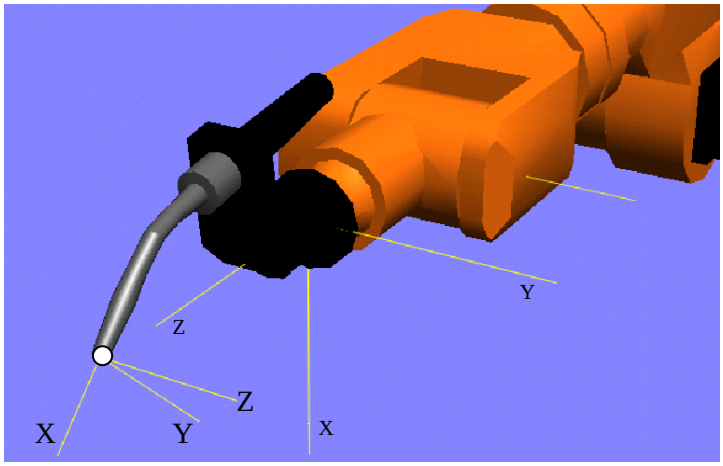


Fig. 2.17 – Robot amb TCP de la eina calculat

- Càlcul d'una BASE:

La base es pot calcular mitjançant qualsevol dels següents mètodes implementats al robot:

Mètode	Mesura a través de...
3 punts	Desplaçament al punt de referència d'una peça.
Indirecte	Indicació del punt de referència inabastable d'una peça.
Entrada numèrica	Introducció manual d'un punt de referència.

A cadascun d'aquests programes de mesura se li han assignat formularis que mitjançant diàlegs permeten la realització de la parametrització de la base corresponentment.

En el meu cas he trobat oportú utilitzar el mètode dels 3punts.

3 PUNTS:

Emprant aquest mètode es determina el punt de referència de una peça (BASE).

Això és produeix per desplaçament i memorització de tres punts específics amb una eina, les dimensions del qual són conegudes per la unitat de control (és a dir, té el TCP calculat). Aquests tres punts determinen tant la posició del origen com la orientació del sistema de coordenades de la peça.

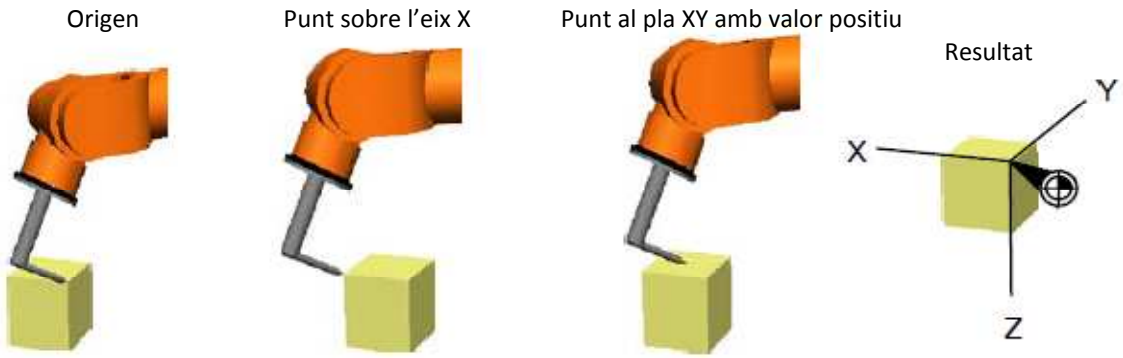


Fig. 2.18 - Càlcul de la BASE mètode 3 punts.

2.3 L'ARMARI (KR C2)

En la sala de formació tenim, per a cada robot, el seu armari de control (V)KR C2. Aquest ens permet fer el càlcul dels moviments i controla tots els eixos interns i fins a dos externs només en un armari (si es desitgés tenir encara més eixos externs per al mateix robot caldria emprar un armari addicional).

Dades bàsiques:

Tipus d'armari	KR C2 edition05
Quantitat d'eixos	Màx. 8
Pes	185 kg aprox.
Tipus de protecció	IP 54
Nivell de soroll segons DIN 45635-1	Valor mig 67 dB (A)
Separació amb altres armaris.	Lateralment, distància 50mm
Càrrega sobre el sostre amb distribució parella	1000N

Connexió a la xarxa:

Tensió nominal connectada segons DIN/IEC 38	AC 3x400V... AC 3x415V
Tolerància permesa de la tensió nominal	400V -10% ... 400V +10%
Freqüència de la xarxa	49 ... 61 Hz
Potència d'entrada nominal (estàndard)	7,3 kVA, veure placa caract.
Potència d'entrada nominal (Càrrega pesada, robots de paletitzat i interprenses)	13,5 kVA, veure placa caract.
Fusibles de la entrada d'alimentació	Mín. 3x25 A lent, màx 3x32 A lent (veure placa característica).
Fusible magnetotèrmic Diferència corrent de dispar	300mA per cada unitat de control de robot, sensible a corrent universal.
Equiparació de potència	Per als cables d'equipació de potencial i tots els cables de posta a terra, el punt de estrella comú és la barra de referència de la secció de potència.

Situació de la placa característica de l'armari (veure Fig. 2.19):

Fig. 2.19 - Placa característica de l'armari de control (V)KR C2

L’armari KR C2 permet el control dels 6 eixos des del robot KR 6 fins al KR 500.

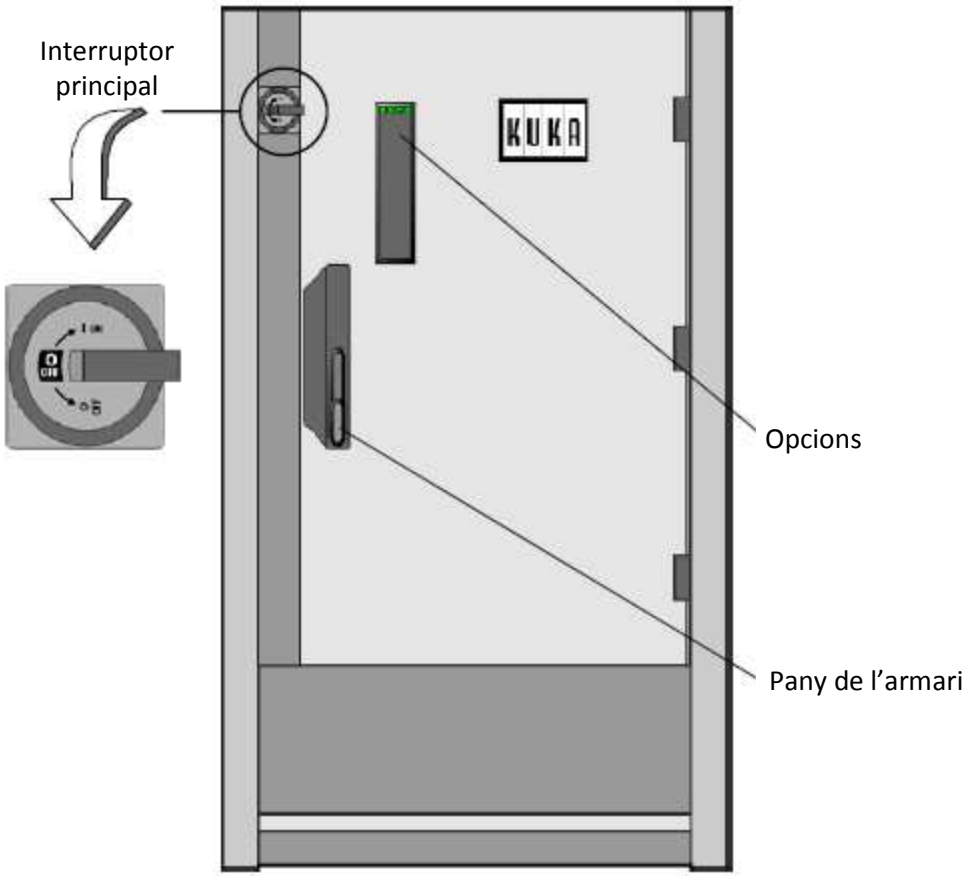
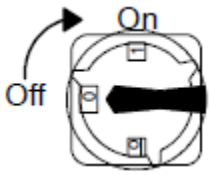
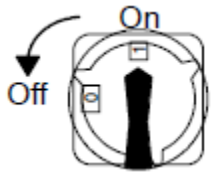


Fig. 2.20 – Esquema d’armari estàndard exterior.

Interruptor principal: Amb l’interruptor principal es connecta o desconnecta el sistema de robot i la unitat de control. Un cadenat col·locat a l’interruptor principal pot evitar, amb seguretat, una reconnexió no autoritzada (per exemple, en les tasques de manteniment en el sistema de robot).



Després de connectar el sistema de robot amb l’interruptor principal de l’armari de control, l’ordinador comença a posar en marxa (carregar) el sistema operatiu i el software de control. Aquest procés de càrrega dura uns minuts



Després de desconnectar el sistema de robot amb l’interruptor principal de l’armari de control la unitat de control tanca el seu propi software i desactiva el sistema operatiu. Durant aquest procediments es graven i asseguren certs fitxers de forma automàtica (funció Power Off). Això succeeix si la unitat de control ha estat arrencada amb anterioritat de forma correcta i completa.

Opcions: Si l’armari de la unitat de control està equipat amb condicions addicionals, les funcions de les mateixes es poden reconèixer mitjançant LED’s. (En el cas de la sala de formació els armaris no disposen de indicadors LED).

Pany de l’armari: El pany de l’armari es protegeix amb una tapa, que alhora, serveix de maneta.

Interiorment una unitat de control o “armari” està format per les seccions següents (veure Fig. 2.21):

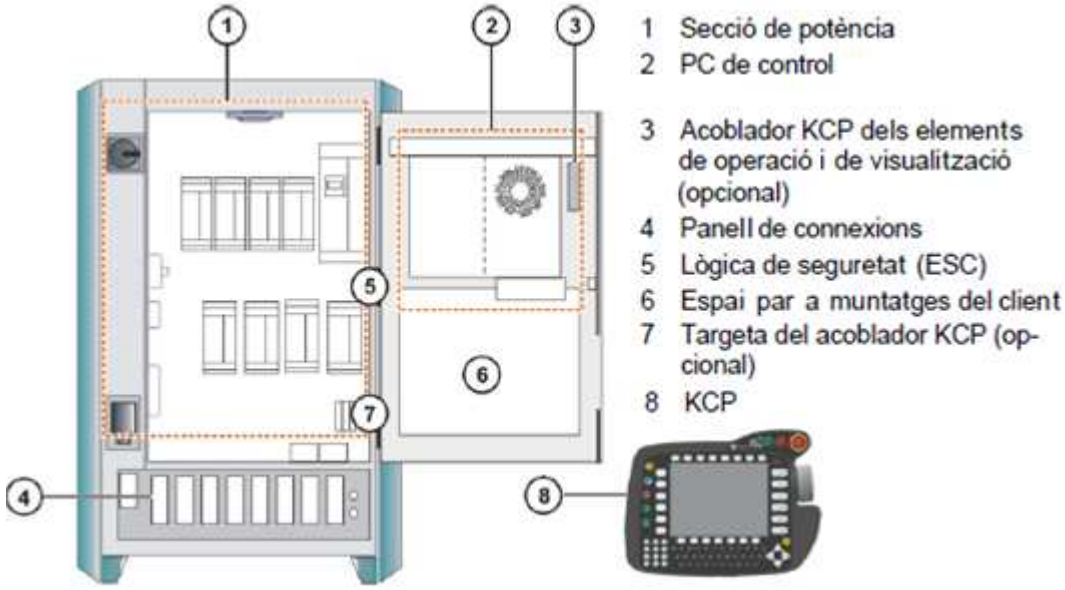


Fig. 2.21 - Esquema de l'interior d'un armari o unitat de control

A continuació veurem cadascuna d'aquestes seccions amb més deteniment de forma que s'entengui com funciona una unitat de control. Això és interessant de cara a comprendre com actuen les seguretats, qui gestiona

Dins de la secció de potència trobem bàsicament la KPS (que és el mòdul d'alimentació) i els servos dels eixos (veure Fig. 2.22).

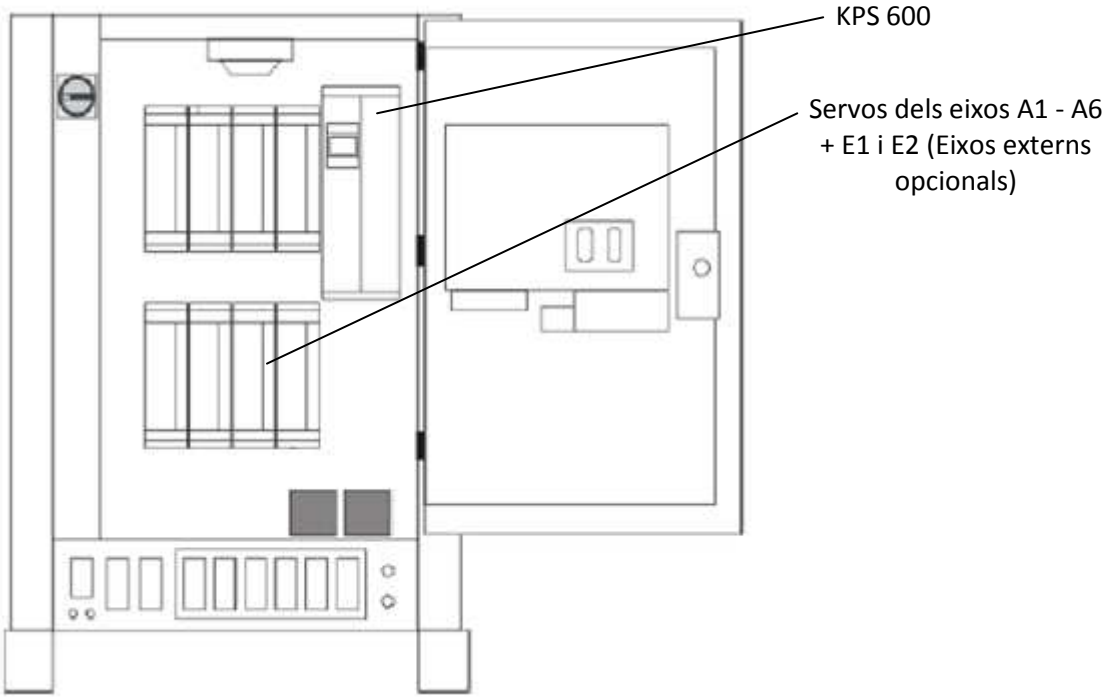


Fig. 2.22 – Esquema d'un armari estàndard interior

En quant al PC de control el trobem, com ja hem vist, a la porta de la unitat de control. Aquest assumeix la major part de les funcions de la unitat de control del robot gràcies al seus components endollables i funciona amb Windows Embedded. A continuació (Fig. 2.23) veurem amb més deteniment aquesta peça clau de l'armari de control:

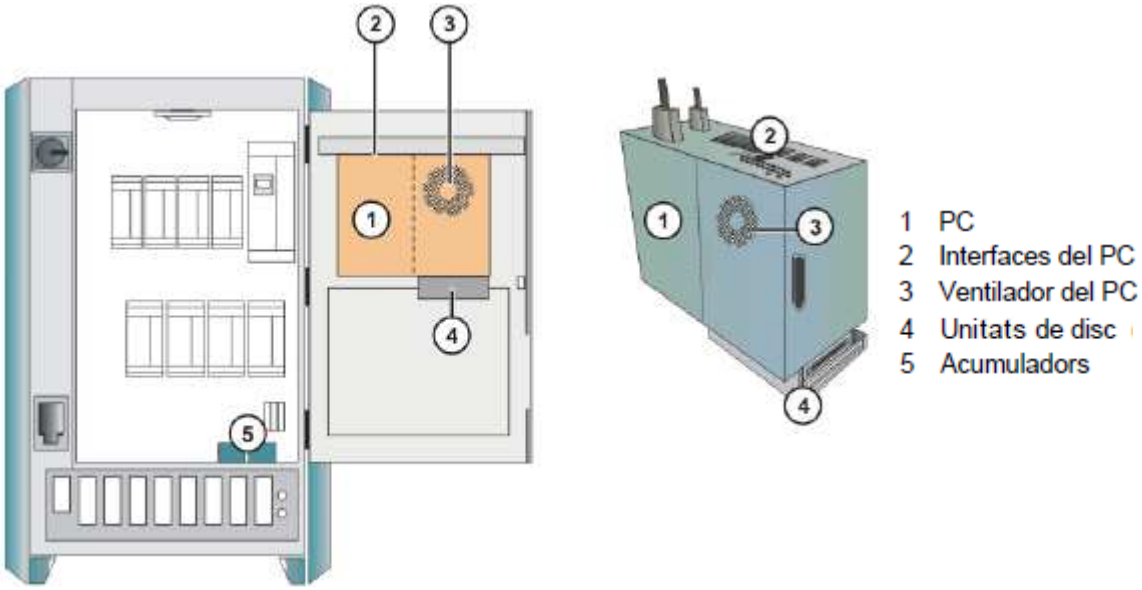


Fig. 2.23 - PC de control

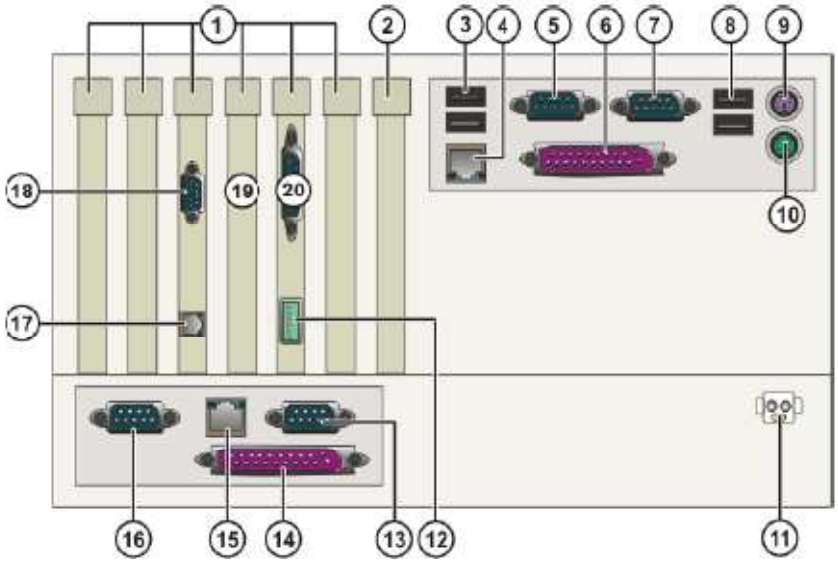


Fig. 2.24 - Interfaces del PC de control.

Slot	Interface	Slot	Interface
1	Endolls PCI (1 a 6)	11	X961 Alimentació de tensió 24V DC
2	Endoll AGP PRO	12	X801 DeviceNet (MFC3)
3	2 USB	13	ST5 connexió sèrie de temps real COM 3
4	X804 Ethernet	14	ST6 ESC/KCP i altres similars
5	COM 1 (sèrie)	15	ST3 Bus d'accionaments al KPS 600
6	LPT1 (paral·lel)	16	ST4 RDW sèrie X21
7	COM 2 (sèrie)	17	X805 KCP Display (KVGA)
8	2 USB	18	X821 Monitor Extern (KVGA)
9	Connexió de teclat	19	L'endoll 4 queda lliure. Si en la MFC3 s'endolla la segona DSE-IBS-C33 AUX, està ocupat el endoll 4.
10	Connexió de ratolí	20	X2 Entrades i sortides DC. SSB a la targeta CI3

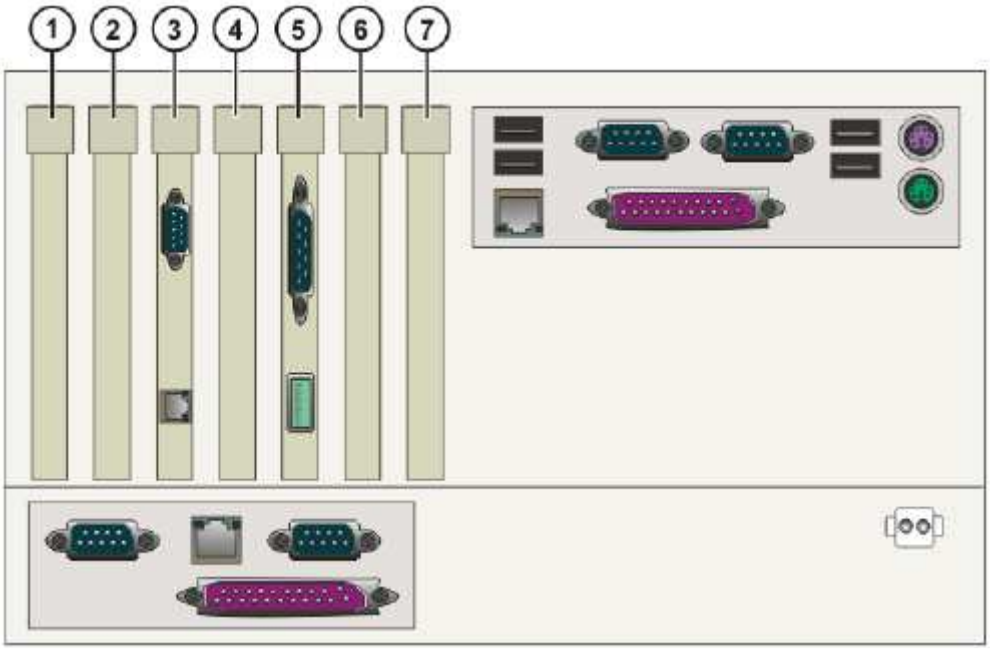


Fig. 2.25 - Slots del PCI.

Slot	Targeta (PCI)
1	Targeta Interbus (FOC) (Opcional) Targeta Interbus (coure) (Opcional) Targeta escàner LDPN (Opcional) Targeta Profibus mestre/esclau (Opcional) Targeta CN_EthernetIP (Opcional)
2	Targeta escàner LDPN (Opcional)
3	Targeta KVGA
4	Targeta DSE-IBS-C33 AUX (Opcional)
5	Targeta MFC3
6	Targeta de xarxa – Network (Opcional) Targeta escàner LDPN (Opcional) Targeta Profibus mestre/esclau (Opcional) Targeta LIBO-2PCI (Opcional) Targeta mòdem KUKA (Opcional)
7	Lliure

La Lògica de seguretat ESC (Electronic Safety Circuit) és un sistema de seguretat bicanal suportat per processador. Controla permanentment tots els components rellevants de seguretat connectats. En el cas de falles o interrupcions del circuit de seguretat, desconnecta la alimentació dels accionaments provocant una aturada del sistema del robot.

El sistema ESC consta de targeta CI3, KCP (master), KPS600 i MFC (node passiu). El sistema ESC amb perifèria de nodes reemplaça totes les interfaces d'un sistema clàssic de seguretat. Aquesta lògica controla les entrades següents:

- Aturada d'emergència local i externa.
- Protecció del operari.
- Polsador d'home mort.
- Accionaments DESC
- Accionaments CON.
- Modes de servei.
- Entrades qualificants.

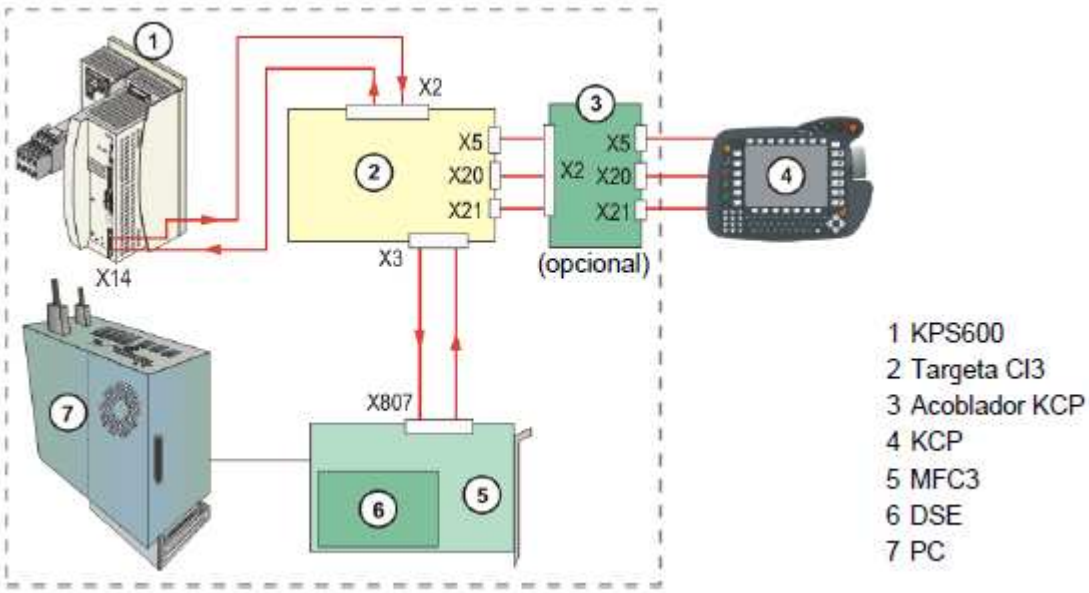


Fig. 2.26 - Estructura del circuit ESC

En quant al Panell de connexions (Fig. 2.27) hi ha una sèrie d'espais reservats segons el que cal endollar:

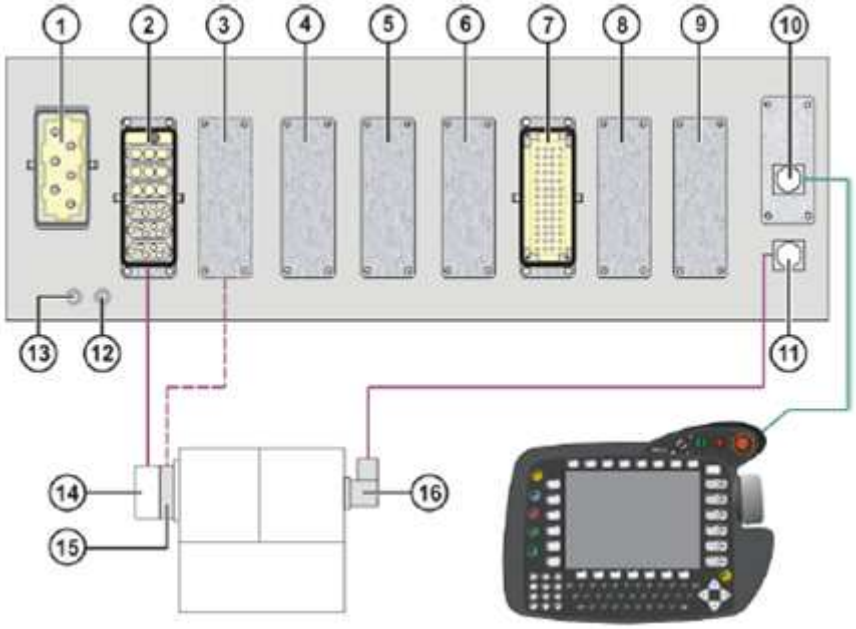


Fig. 2.27 - Connexions del panell de connexions.

Núm	Connexió	Núm	Connexió
1	Connexió xarxa X1/XS1	9	Opció
2	Connexió de motor X20	10	Connexió KCP X19
3	Connexió de motor X7	11	Connexió RDW X21
4	Opció	12	Cable posta terra al robot SL1
5	Opció	13	Cable posta terra escomesa principal SL2
6	Opció	14	Connexió de motor a caixa X30
7	X11	15	Connexió de motor a caixa X30.2
8	Opció	16	Connexió RDW a caixa connexions X31

Característiques del PC de control:

Processador principal	Segons la versió de subministrament.
Mòduls de memòria DIMM	Min. 256 MB
Disc dur, disquetera, unitat de discos CD-ROM	Segons la versió de subministrament.

Característiques del KUKA Control Panel:

Tensió d'alimentació	26,8 V DC
Mesures (ample x alt x prof)	Aprox. 33x26x8 cm^3
Resolució Display VGA	640x480 punts
Tamany del Display VGA	8"
Pes	1,4 kg
Longitud del cable	10 m

La unitat de control del robot s'utilitza per el control dels sistemes següents: Robots KUKA, KMC i Cinemàtica externa.

Un sistema de robot està format per: PC de contorl, secicó de potència, unitat manual de programació (KCP), lògica de seguretat ESC i acoplador KCP (opcional).

Les E/S es poden configurar amb els components següents:

- Devicenet
- Targetes de bus de camp opcionals
  - Interbus
  - Profibus
  - DeviceNet
- Profinet
- Interfaces específiques del client

Interconnexió X11: A través de la connexió X11 es poden connectar dispositus d'aturada d'emergència o concatenar la instal·lació amb unitats de control superiors (com podria ser un PLC).

Alhora de interconnectar la interface X11 s'ha de tenir en compte:

- La instal·lació
- Seguretats

Per a més informació veure el diagrama de la pàgina següent i mirar el punt 4 d'aquesta memòria. En funció de la targeta CI3 es disposaran de diferents senyals i funcions.

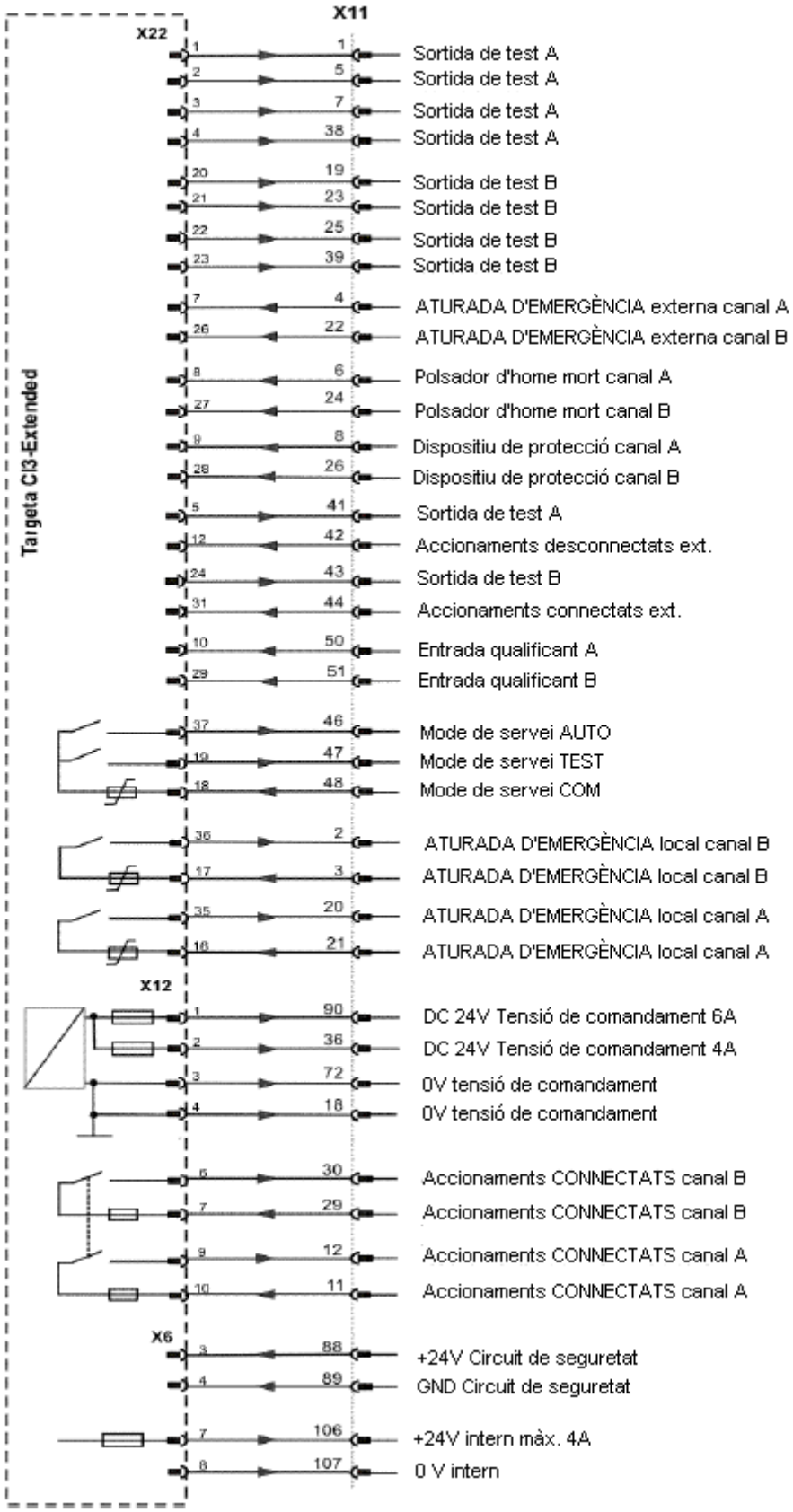


Fig. 2.28 - Assignació de contactes.

Ara que ja hem identificat els elements que conformen la unitat de control podem veure les fotos d'un armari d'una de les cel·les de la sala de formació. Com podeu observar a les figures Fig. 2.29 els servos dels eixos externs no hi són, donat que no en tenim cap. Quan s'afegeix cal introduir aquests servos dins l'armari, en aquest espai i connectar-los corresponentment.

Fig. 2.29 – Fotos interior d'un armari de la sala de formació.

Fig. 2.30 – Foto exterior d'un armari de la sala de formació.

Com podem observar a la part exterior de l'armari (Fig. 2.30) tenim un suport per a desar les KCP de manera que queden recollides i no hi pugui haver cap accident. Aquest suport es demana a part del armari.

2.4 KCP (KCP2 STDED05):

El KUKA Control Panel, que d'ara en endavant anomenarem "KCP", forma part de la interfície entre la persona i la màquina i serveix per una operació senzilla de la unitat de control (armari) del robot. Tots els elements de la programació i operació del sistema robot, amb excepció del interruptor principal, es troben situats directament al KCP. El KCP no només es pot emprar com a unitat de sobretaula sinó també com a aparell portàtil. Les empunyadures i els polsadors d'home mort s'han col·locat de forma que el KCP pugui ser manipulat amb facilitat tant per dretans com per esquerrans.

El display gràfic LCD VGA en color, serveix per a la visualització de les accions d'operació i programació. Hi ha molts elements que poden resultar familiars als que estiguin acostumats a treballar amb el sistema operatiu "Windows".

Tot seguit veurem una petita sinopsis sobre els elements i la superfície gràfica d'operació del KCP:

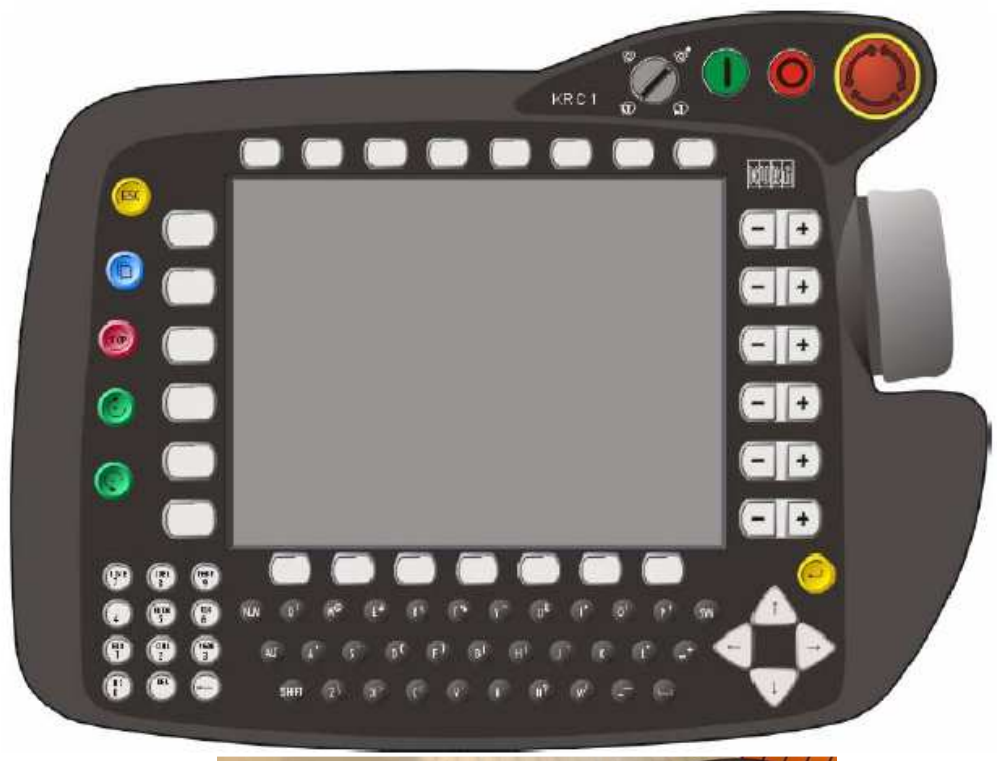
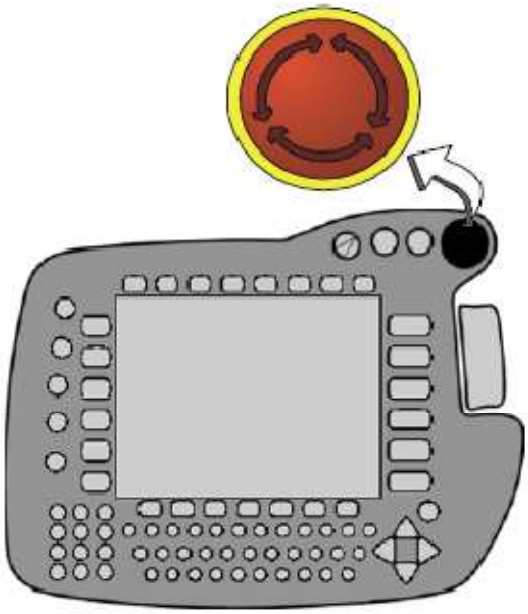
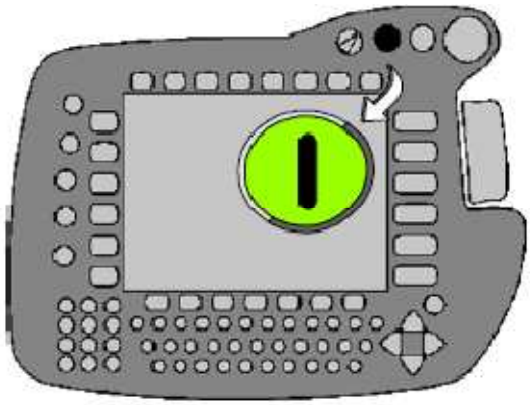


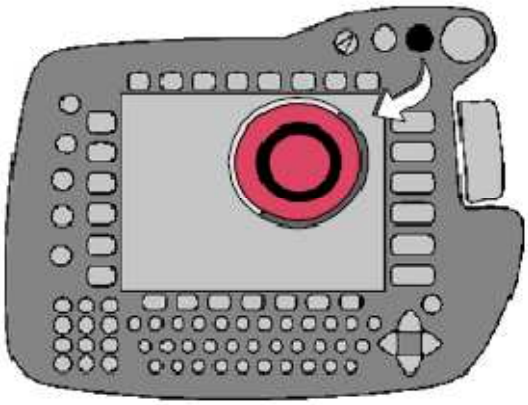
Fig. 2.31 Fotografia d'una KCP de la sala de formació



Polsador d'aturada d'emergència:El polsador d'aturada d'emergència representa el dispositiu de seguretat més important. Aquest polsador de color vermell s'acciona en casos de perill, amb la qual cosa, els accionaments del robot s'aturen immediatament.
Abans de que els accionaments es puguin tornar a connectar, cal desenclavar la tecla d'aturada d'emergència girant la part superior en sentit horari fins que el botó s'aixequi. Cal confirmar després el missatge d'emergència a la finestra de missatges.
Aquesta emergència actua creant un stop proper a la trajectòria del robot.

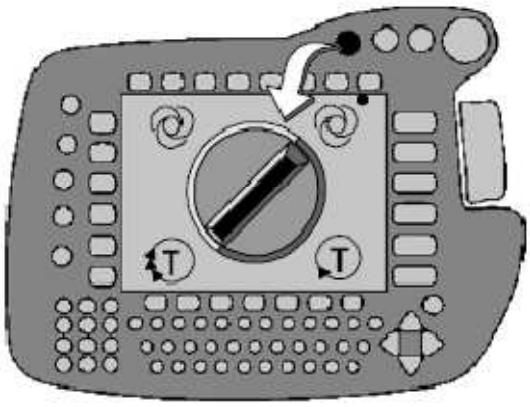
Accionaments connectats:Accionant aquest polsador es connecten els accionaments del robot.
Aquests només poden ser connectats sota condicions normals de servei (sense cap emergència activada, la porta de protecció tancada, etc.).
En la posició de selecció de mode de servei "manual" aquest polsador no té cap funcionalitat.





Accionaments desconnectats:Accionant aquest polsador es desconnecten els accionaments del robot. I així també es tanquen, amb cert retard, els frens dels motors, mantenint els eixos a la posició.
En la posició de selecció de mode de servei "manual" aquest polsador no té cap funcionalitat.
"Accionaments desconnectats" ocasiona una aturada per efecte generador.

Selector de modes de servei:Amb aquest interruptor de clau es pot commutar entre els modes de servei següents:



*Test 1:*El robot el mou només mentre es tingui polsat una de les tecles d'home mort (situades a la part del darrera de la KCP)

El moviments s'executen amb velocitat reduïda.

*Test 2:*El robot el mou només mentre es tingui polsat una de les tecles d'home mort (situades a la part del darrera de la KCP).

El moviments s'executen amb la velocitat programada.

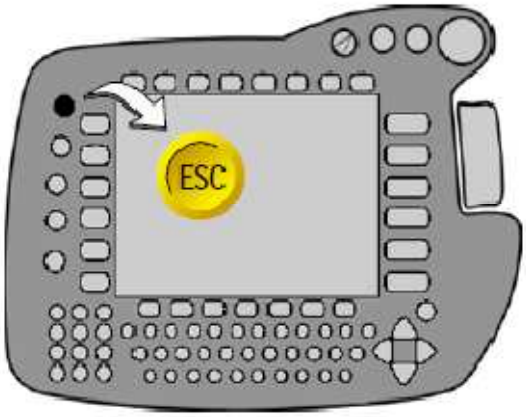
*Automàtic:*El robot processa automàticament el programa seleccionat i es controla a través de la KCP.

El moviment s'executa amb la velocitat programada.

*Automàtic extern:*El robot processa automàticament el programa seleccionat i es controlat per un ordinador de control superior o un PLC.

El moviment s'executa amb la velocitat programada.

Si amb el programa en execució es canvia el mode de servei, això ocasiona una parada per efecte generador.



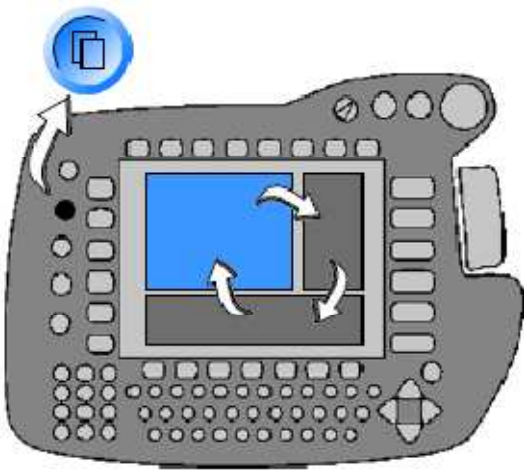
Tecla d'escapament (ESC):Qualsevol acció que ja ha començat pot ser interrompuda en qualsevol moment amb la tecla Escape. A aquestes accions compten formularis inline i finestres d'estat obertes.

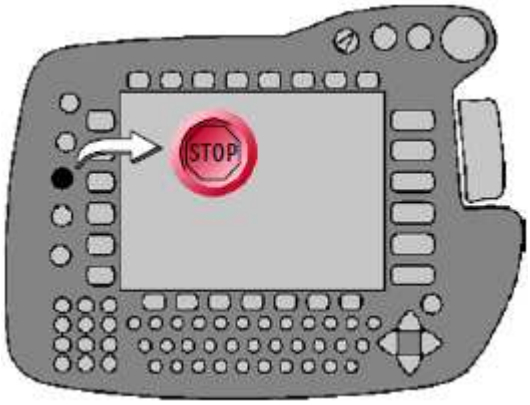
Així com també es poden tancar menús oberts de forma equivocada pas a pas emprant ESC.

Tecla de selecció de finestra:Mitjançant aquesta tecla es pot commutar entre les finestres del programa, quan estiguin disponibles.

El segon pla de la finestra seleccionada (activada) es ressalta en color.

Val a dir que els menús varien en funció de la finestra en que ens trobem en aquell moment!



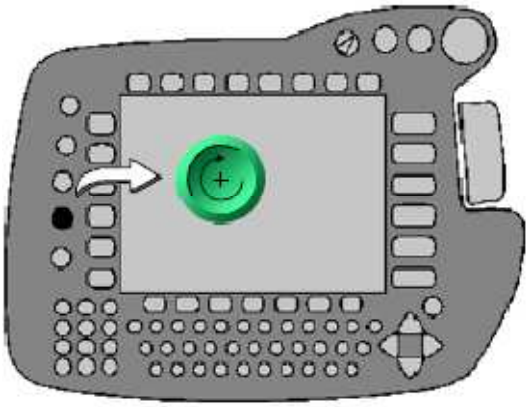


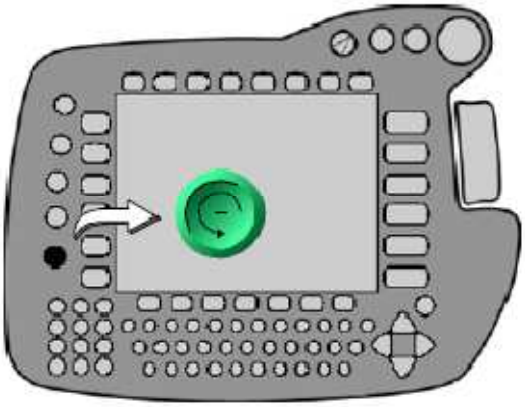
Stop (aturada) del programa:Polsant aquesta tecla es deté la execució d'un programa.
S'obté una aturada sobre la trajectòria, que pot ser confirmada en el mode de servei automàtic.

Per a continuar novament amb el programa seleccionat, accionar la tecla "Marxa del programa endavant".

Marxa del programa endavant:Polsant aquesta tecla es posa en marxa el programa seleccionat.
La posta en marxa només és possible si els accionaments es troben connectats i no hi ha situació d'aturada d'emergència.

En el mode Test 1 o Test 2, al deixar anar aquesta tecla es provoca una detenció del moviment sobre la trajectòria.





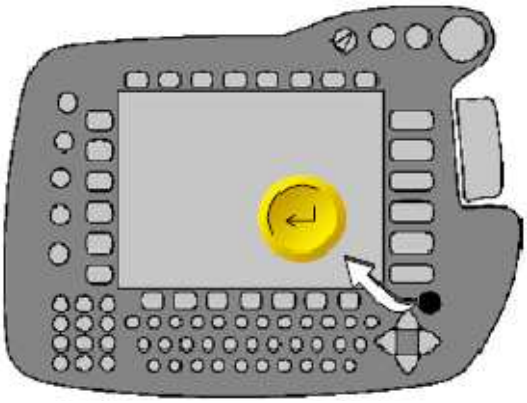
Marxa del programa enrere:Accionant aquesta tecla es processa pas a pas els registres de moviment del programa seleccionat en direcció a al inici del programa.

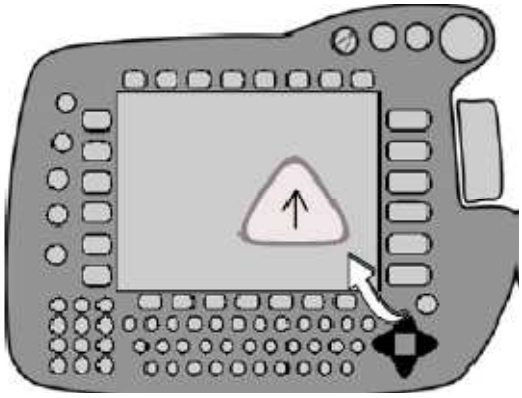
En aquest cas el robot pot moure's en sentit contrari a la trajectòria originalment programada.

Deixar anar la tecla en Test 1 i Test 2 causa una detenció del moviment sobre trajectòria.

Tecla d'entrada:Aquest element d'operació té la funció de la tecla "Enter", o bé "Return" del teclat d'un PC convencional.

Amb aquesta tecla es tanquen instruccions, es confirmen entrades en formularis, etc.





Tecles del cursor ←↑↓→:Les tecles del cursor serveixen per:

1. Modificar la posició del cursor d'edició.

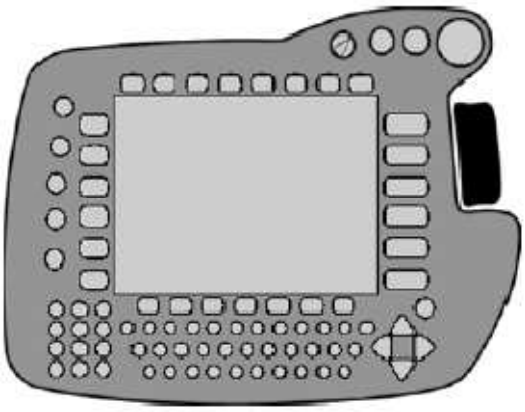
2. Canviar la posició entre els camps dels formularis inline y llistes de paràmetres.

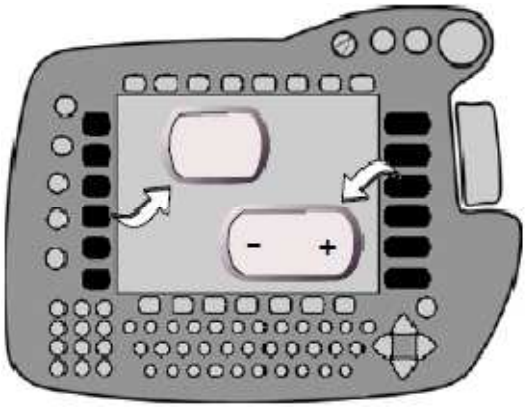
Per això prémer la fletxa corresponent. La funció és molt semblant a la del teclat d'un PC.

Space-Mouse:Aquest element d'operació serveix per el moviment en mode manual dels 6 eixos (graus de llibertat) del robot. La mesura de la desviació en el Space-Mouse influeix en la velocitat de desplaçament del robot.

Alternativament es poden emprar també les funcions d'estat +/- del costat dret del display.

Per a més informació veure subcapítol4.4.1.

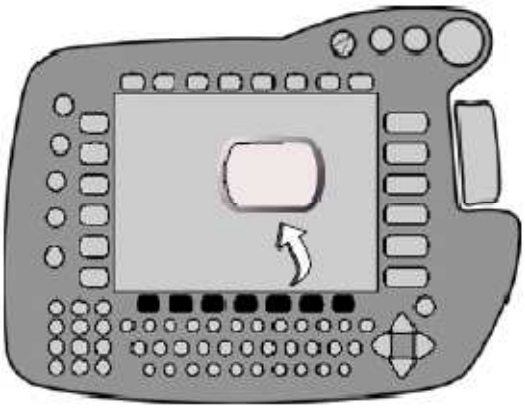




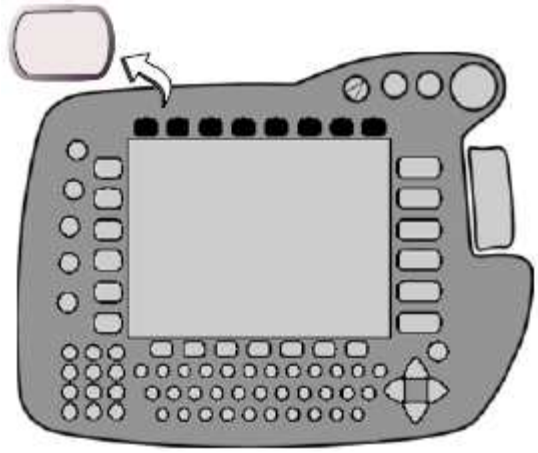
Tecles de funcions d'estat:Les tecles de funcions d'estat (costats esquerre i dret del display), serveixen per a la selecció d'opcions de servei, per a commutar funcions individuals i per a definir valors.

Cada funció es representa gràficament amb el símbol corresponent a la barra de funcions d'estat.

Softkeys:Amb aquests elements d'operació se seleccionen les funcions representades en la barra de softkeys (part inferior de la pantalla).

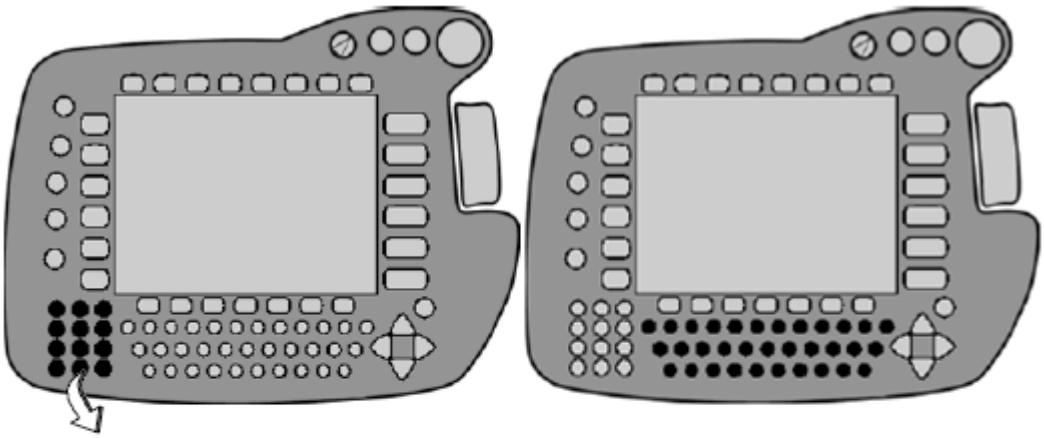


Les funcions que es trobin disponibles per la selecció s'adapten dinàmicament, això vol dir que aquesta barra va modificant la seva assignació.



Tecles dels grups de menús:Amb aquestes tecles s'obre un menú a la barra de menús (sobre el display).

Del menú que s'obre, la selecció es pot efectuar amb les tecles de cursor o per la entrada del número corresponent a la denominació del menú.



Teclat numèric i teclat:La funcionalitat de ambdós teclats és molt similar amb els ordinadors personals.

La commutació de teclat entre majúscules i minúscules s'efectua amb les tecles "Majúscules" (SHIFT). Per a seleccionar el símbol de cada tecla es selecciona amb "SYM".

La commutació entre els diferents nivells del camp numèric es produeix mitjançant un breu accionament de la tecla "NUM".

La combinació de tecles és possible (igual que amb un ordinador convencional).

Part del darrera de la KCP:

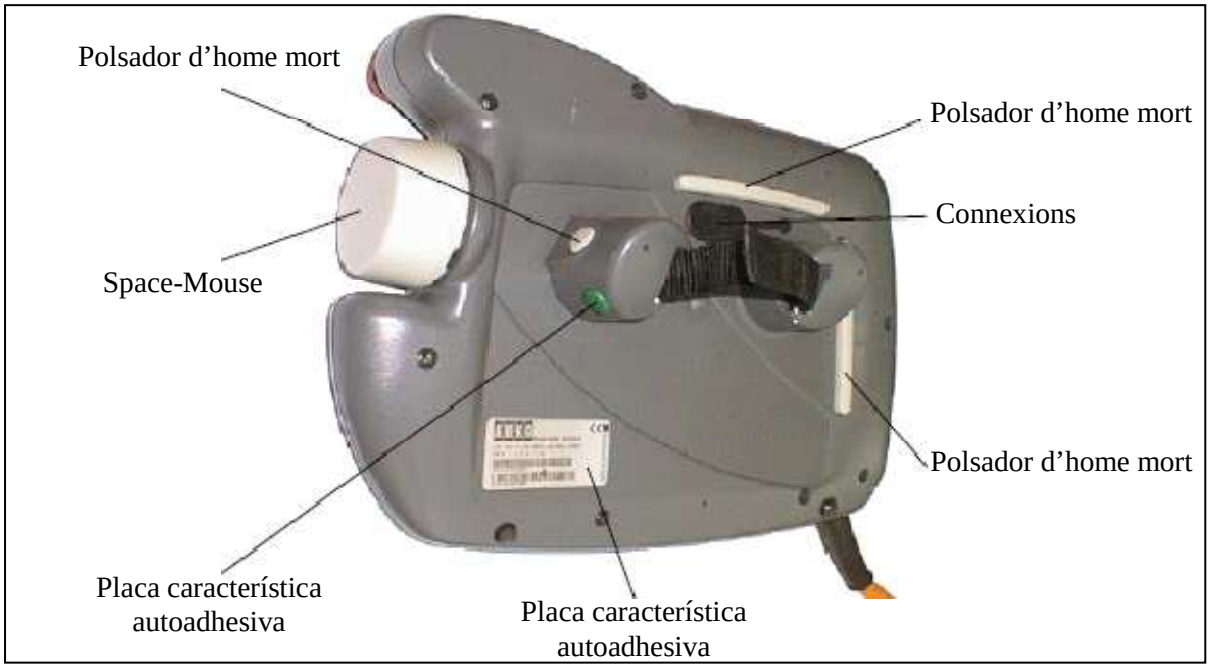


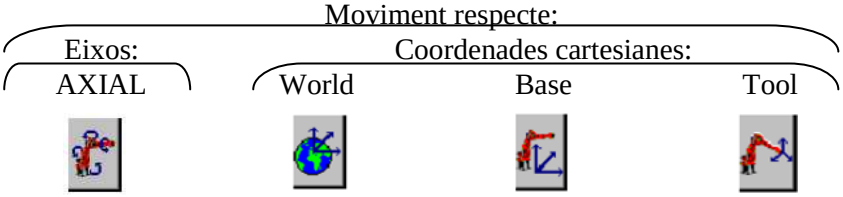
Fig. 2.32 – Part del darrera de la KCP

Fig. 2.33 –Ampliació connexions de la KCP

2.4.1 Space-Mouse:

Space-Mouse és un ratolí de 6 graus de llibertat instal·lat al costat dret del KCP i que permet al operari que estigui acostumat a l'ús del space-mouse a moure el robot de forma més ràpida i intuïtiva.

El primer que cal fer és habilitar el moviment amb el Space-Mouse: 



Determinar quin tipus de moviment volem realitzar:

6D ABC XYZ

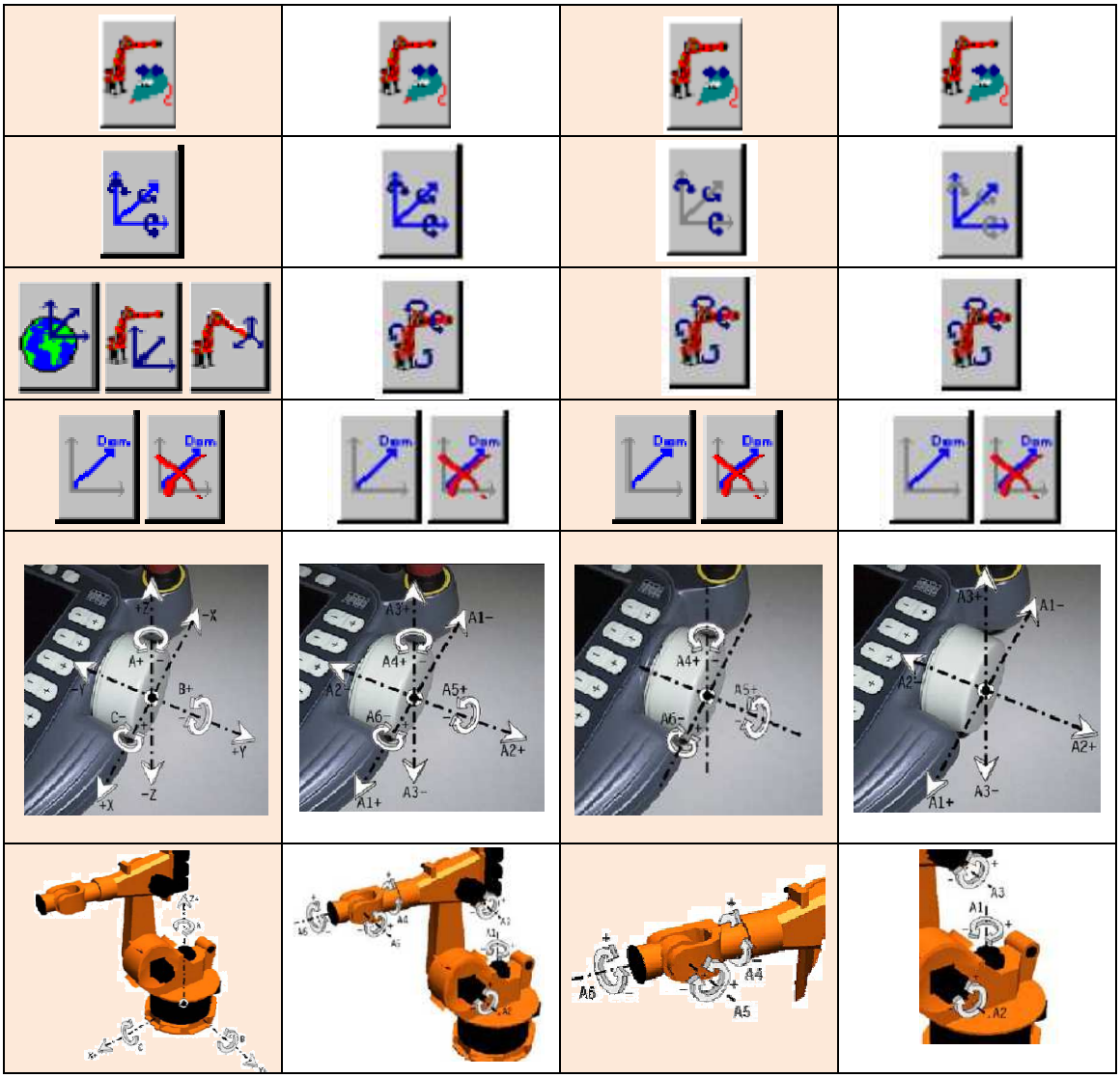


Sensibilitat del Space-Mouse:

Dominant No dominant


La dominància afecta en el desplaçament. Si hem seleccionant dominància el robot només es desplaçarà en l'eix que tingui la major desviació al Space-Mouse. En el cas de no determinar dominància el robot farà tots els moviments que rebi de les desviacions del Space-Mouse en combinació de moviments.

Aquesta taula resum el moviments que descriu el Space-Mouse segons les múltiples seleccions de paràmetres disponible.



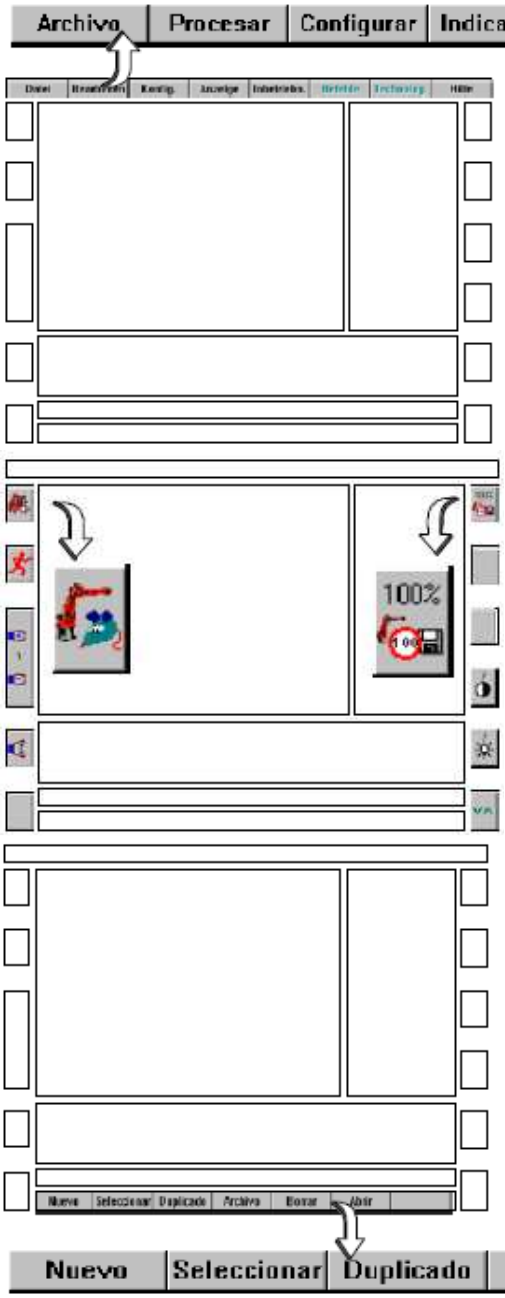
2.4.2 Superfície gràfica de operació (BOF):

El display del KCP està subdividit en diverses zones que s'encarreguen de diferents tasques. Aquests s'adapten automàticament durant el servei als diferents requeriments.

Entre els elements es troben les barres de menú, les barres de funcions d'estat, la barra de softkeys i la finestra de programa, els formularis inline, les finestres d'estat i de missatges així com una línia de funcions d'estat.

La assignació de les barres de menú, de estat i de softkeys depenen de la aplicació instalada.

Tecles de funcions:



*Barra de menús:*En la barra de menú s'han reunit en grups, funcions de control del robot.

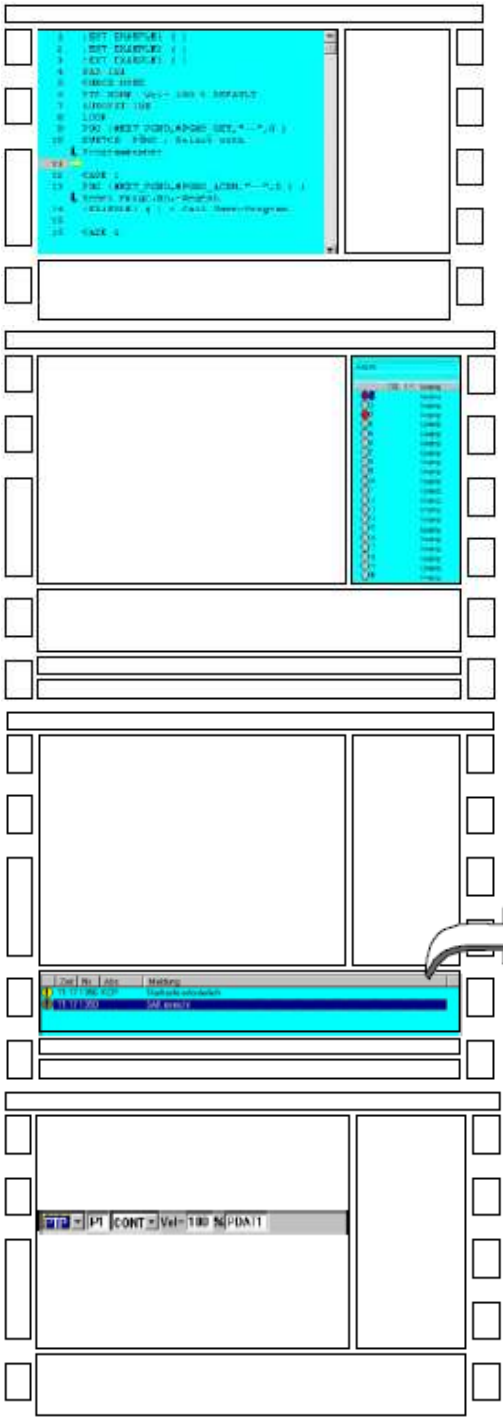
Aquests grups (punts del menú) han d'obrir-se a través de les tecles dels grups de menús (en la barra sobre el display) per a tenir accés a altres seleccions.

*Barra de funcions d'estat:*Les barres de funcions d'estat indiquen les funcions variables de les tecles de funcions d'estat que es troben a la esquerra i a la dreta del display.

L'aspecte i les funcions de les tecles de funcions van variant durant el desenvolupament del programa.

*Barra de softkeys:*A través d'una barra de softkeys d'adaptació dinàmica s'ofereixen funcions que poden ser seleccionades amb les tecles del softkey (sota el display).

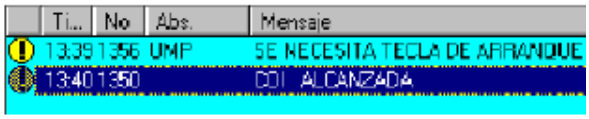
Finestres de entrada i sortida:



*Finestra de programes:*En aquesta finestra es representa el contingut del programa seleccionat (si no hi ha cap programa, al seu lloc es veu el llistat dels programes disponibles).

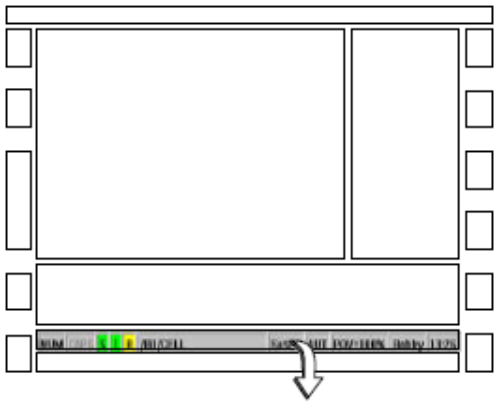
*Finestra d'estat:*En cas de necessitat, es visualitza addicionalment en pantalla, la finestra d'estats (com ara en assignació de sortides), o bé, de entrades (per exemple en la mesura d'eines).

*Finestra de missatges:*La unitat de control es comunica amb l'operari a través d'aquesta finestra. Aquí s'indiquen els missatges estats, d'observacions, confirmacions, esperes i diàlegs.



*Formulari inline:*Una part de les funcions del programa requereix la introducció o declaració de valors. Aquests valors es poden entrar mitjançant una màscara d'entrada (formulari inline) per tal d'assegurar-se que les instruccions programades tenen el format correcte.

Estat del sistema:



*Línia/Barra d'estat:*En la línia o barra d'estat es visualitza informació sobre l'estat dels serveis importants (com ara l'estat del programa).

2.5 SOFTWARE

HMI Studio és una eina que ens permet crear una interfície HMI. Utilitza OPC per a agafar el valors dels TAG i permet emprar variables locals. Com en tots els sistemes de desenvolupament de interfícies el software compta amb 3 modes d'operació (mode CONFIGURE i mode RUNTIME). Amb el primer mode es dissenya, parametritza, s'adjunten, es dona valors, es configuren, etc. els elements disponibles a cada pantalla. Dins del mode Runtime, per contra, no podem modificar la construcció de les pantalles però sí executar l'aplicació per a comprovar el seu funcionament en la versió definitiva.

Human Machine Intergace Graphics Software (Interfície Home Màquina programari gràfic):

- 32 bits, Multi-processador, Disseny multi-plataforma:

Per a Windows® 98/95, Windows NT® i sistemas operatius en general a partir de l'any 2000.

- OPC HMI & SCADA:

Connectar als equipaments de camp a través d' OLE estàndard per al control de processos industrials.

- Controls ActiveX Potents:

Allotjar qualsevol de Microsoft, ICONICS o la gran majoria dels controls ActiveX.

- Visual Basic per a aplicacions de seqüències de comandes:

Crear funcions personalitzades utilitzant el llenguatge estàndard de Microsoft.

- Potents eines de creació i visualització:

Disseny de creació i visualització emprant 'estat-del-art' dels controls ActiveX.

- Animació ràpida i dinàmica:

El canvi de color de gràfics, canviar el tamany, moure, rotar a través de dades en temps real tan ràpid que pot arribar a 20 veces por segon.

- Capes, degradats, expressions i càlculs:

Estil de gradients Autocad, realitza càlculs de gran abast en los objectes gràfics.

- Plantilles gràfiques:

Crear plantilles mestre de creació i visualització que contenen els elements usats més freqüentment, los logotips i les capes d'objectes.

En aquest apartat veurem com funciona el software en general HMI Studio a través d'exemples. En cas de voler saber com s'ha realitzat exactament el disseny de les pantalles per a aquest projecte específic cal veure el punt 3.2 d'aquesta memòria.

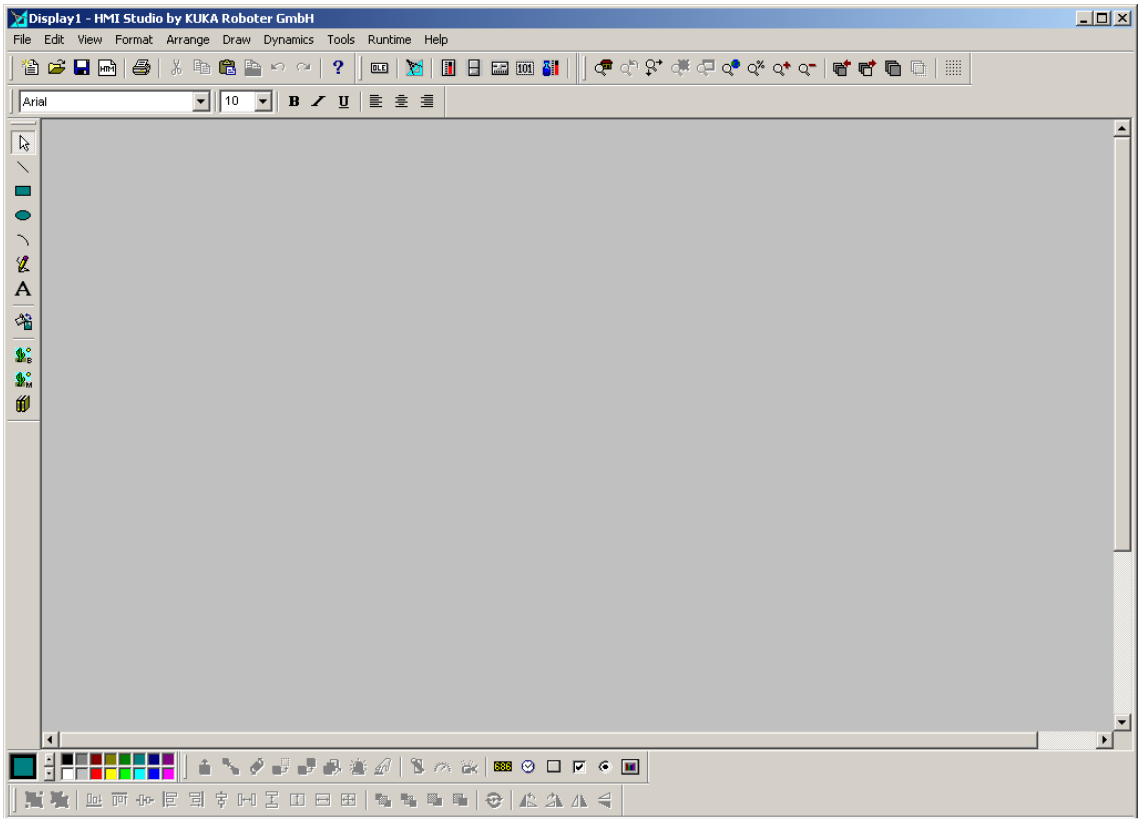
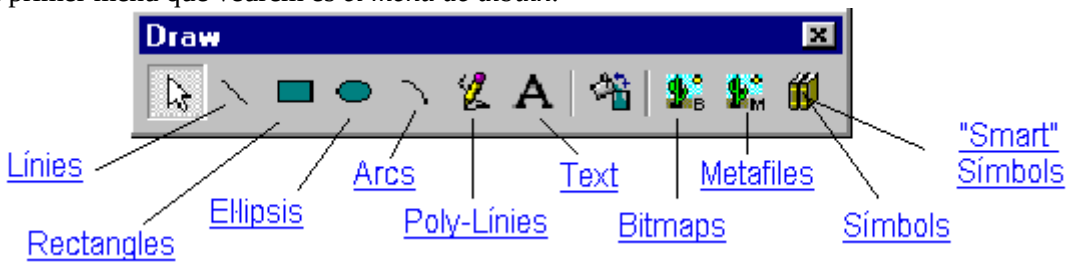


Fig. 2.34 - Pantalla general de HMI Studio

A continuació veurem de quins menús d'icones disposa el HMI Studio i funcionalitats té cada menú:
El primer menú que veurem és *el menú de dibuix*:



Com podem observar es disposa dels elements bàsics de dibuix (dibuix de línies, rectangles, el·lipsis, arcs, dibuix a mà alçada, omplir de color…) i text. Incorporen parametritzacions més potents dels elements de dibuix comparats amb altres programes(com ara el Word o el Paint de Windows®).

En quant als Bitmaps són imatges basades en píxels. Aquestes permeten un toc més real si per exemple se fotografia el procés i s'afegeix la animació per sobre.

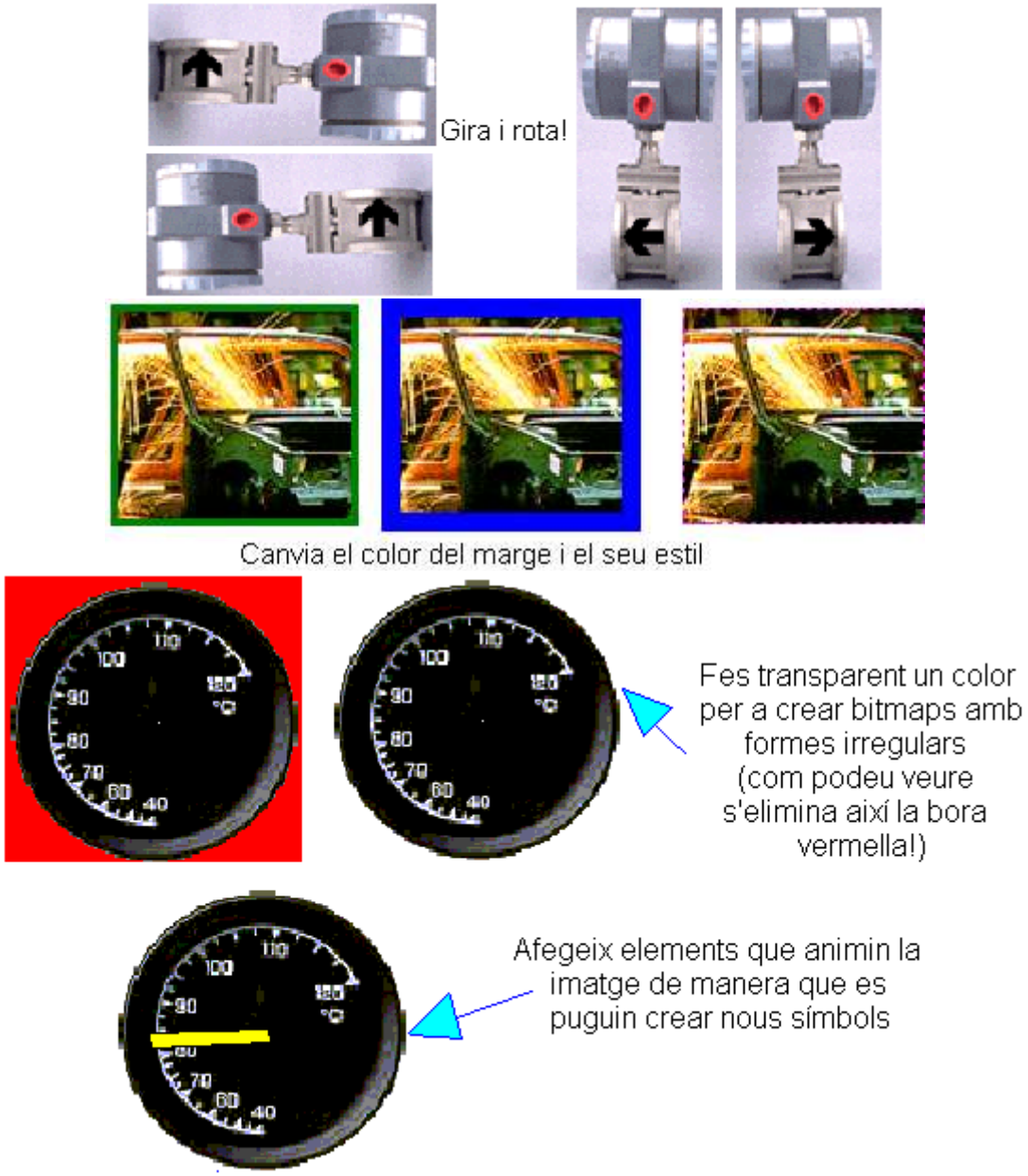


Fig. 2.35 - Algunes de les opcions per als BITMAPS

Els Metafiles són objectes basats en vectors, com els dibuixos de MS-Office. Dibuixats de forma professional i elaborada, objectes de parametrització fàcil disponibles amb el propòsit d'accelerar el procés de desenvolupament i millorar el disseny de les pantalles.

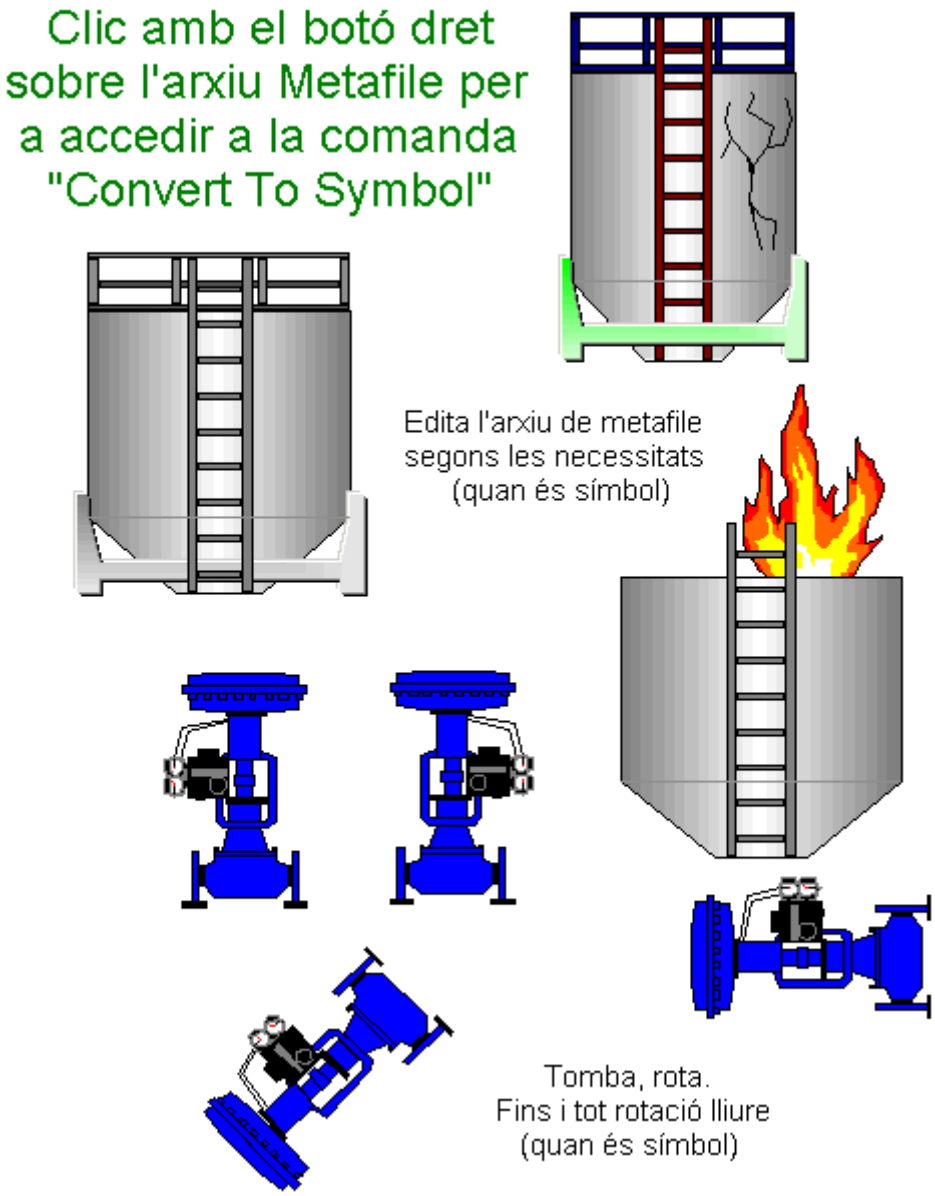
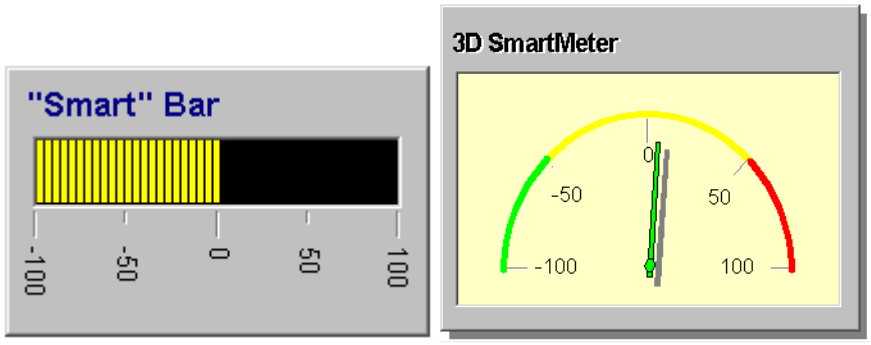


Fig. 2.36 - Algunes de les opcions per els Metafiles

Els símbols o els smart símbols estan pre-fets (tant els dinàmics com els estàtics) i que es poden trobar a les llibreries de manera que es poden crear animacions ràpidament. Es poden crear llibreries i símbols propis! Per a emprar un símbol selecciona'l de la llibreria i arrossega'l a la pantalla per poder-lo fer servir i parametritzar-lo a gust.
Els símbols que vénen per defecte pertanyen al pack GENESIS32 amb uns quants sets de símbols. Sets més extensos addicionals estan disponibles a través de la seva compra.



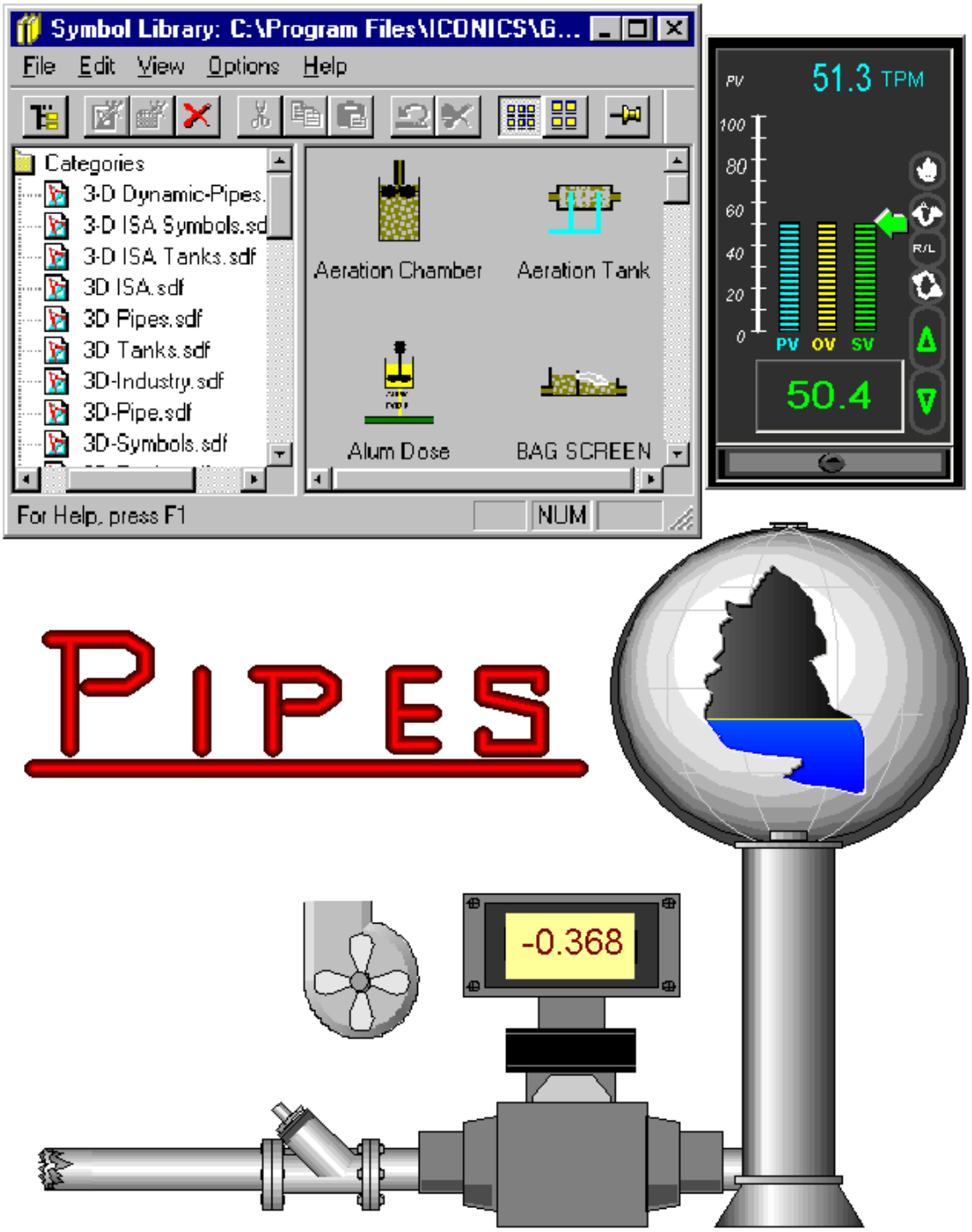
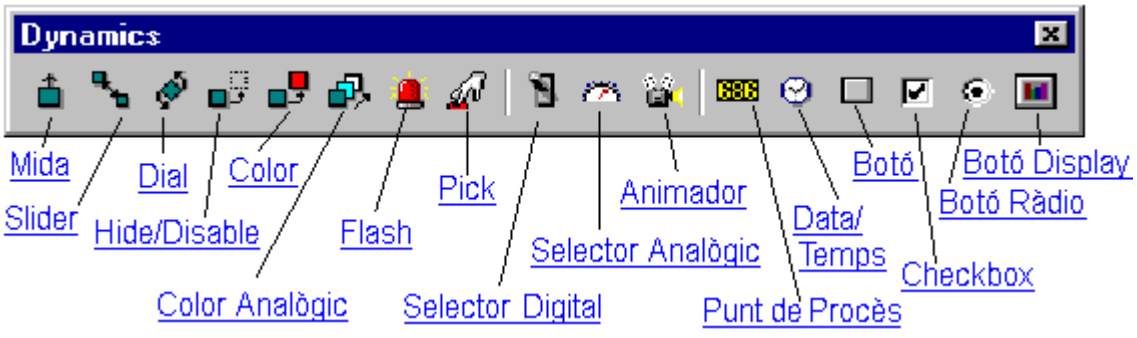


Fig. 2.37 - Exemples de Symbols i Smart Symbols

Passem ara a veure el següent menú, *el menú dinàmic*:



El menú dinàmic incorpora en general els elements de parametrització corresponents per a animar els dibuixos, les imatges, etc. Alhora incorpora elements predefinits com ara botons:

El primer botó, el de la mida, és tan intuïtiu com el seu nom indica. Permet modificar el tamany de l'element seleccionat.

En quant al Slider, ens permet moure en direcció a una trajectòria qualsevol element, text, dibuix, bitmap, metafile... Tan sols cal escollir la direcció de moviment.

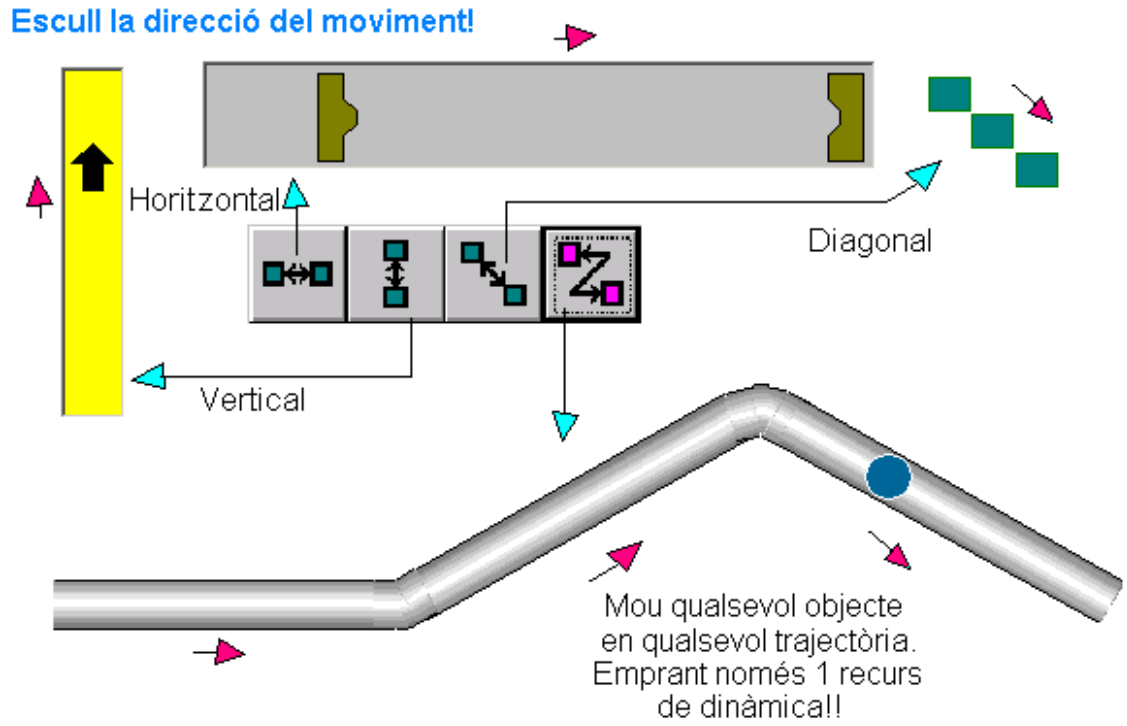


Fig. 2.38 - Opcions de d'un SLIDE

El botó dial, o moviment rotatiu permet, girar, rotar, donar funció de dial, representar, moure, etc. Simplement cal escollir un element, indicar un punt de rotació i un rang d'angles en que es bellugarà l'objecte. Aquesta opció és realment pràctica

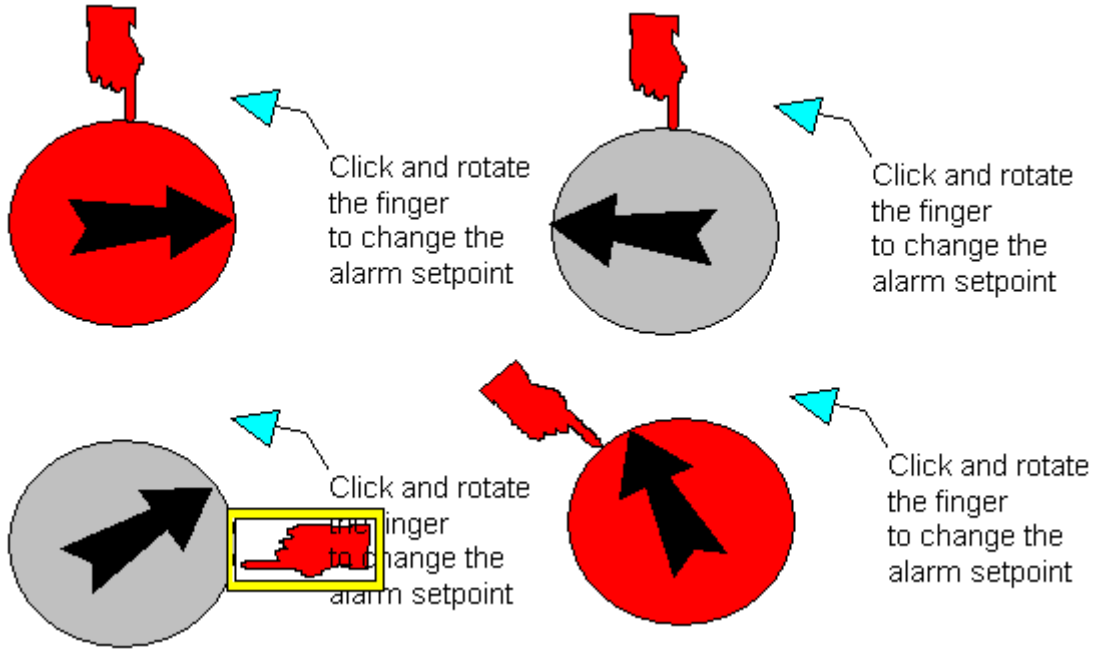
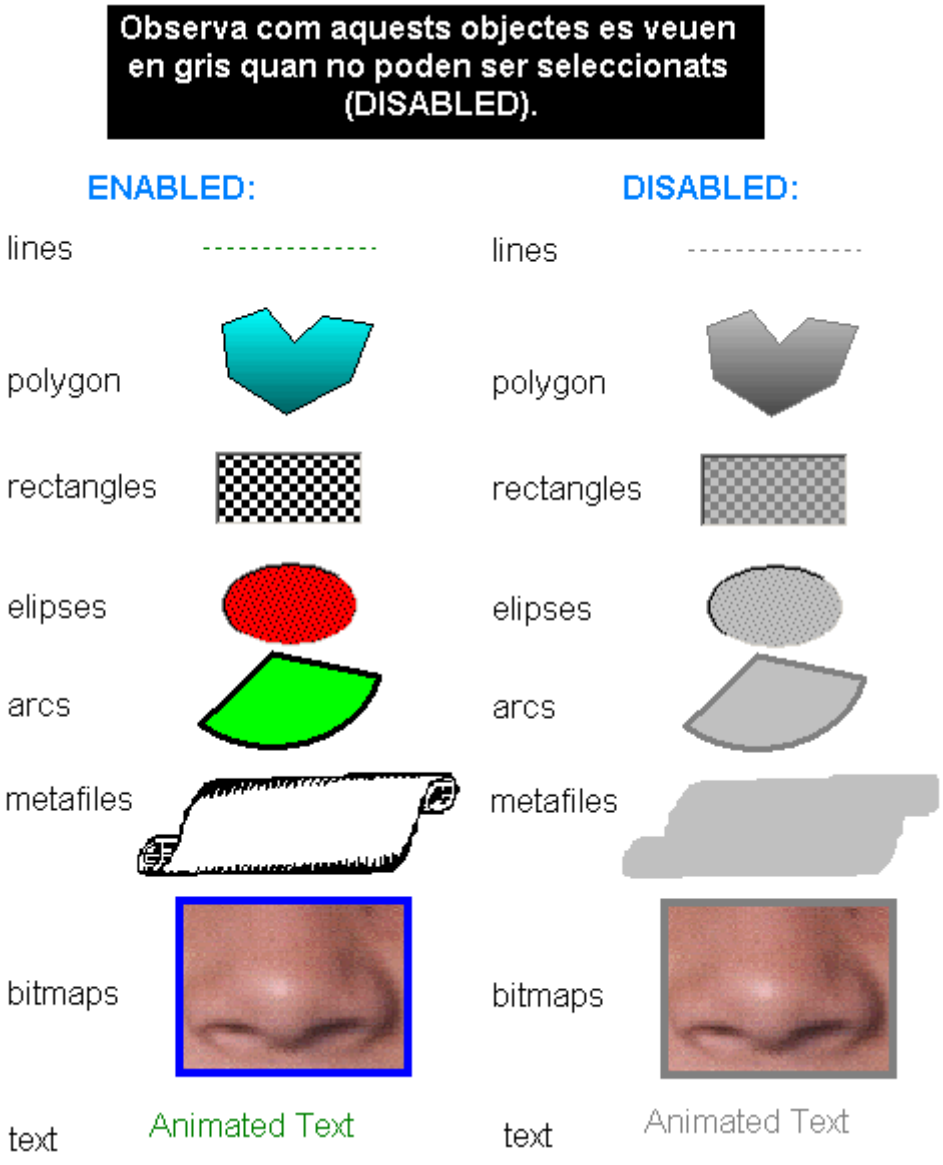


Fig. 2.39 - Exemples d'utilització del dial.

El botó de Hide/Disable permet fer desaparèixer un dibuix (HIDE) o bé inhibir el seu funcionament degut a alguna variable o entrada (DISABLE):



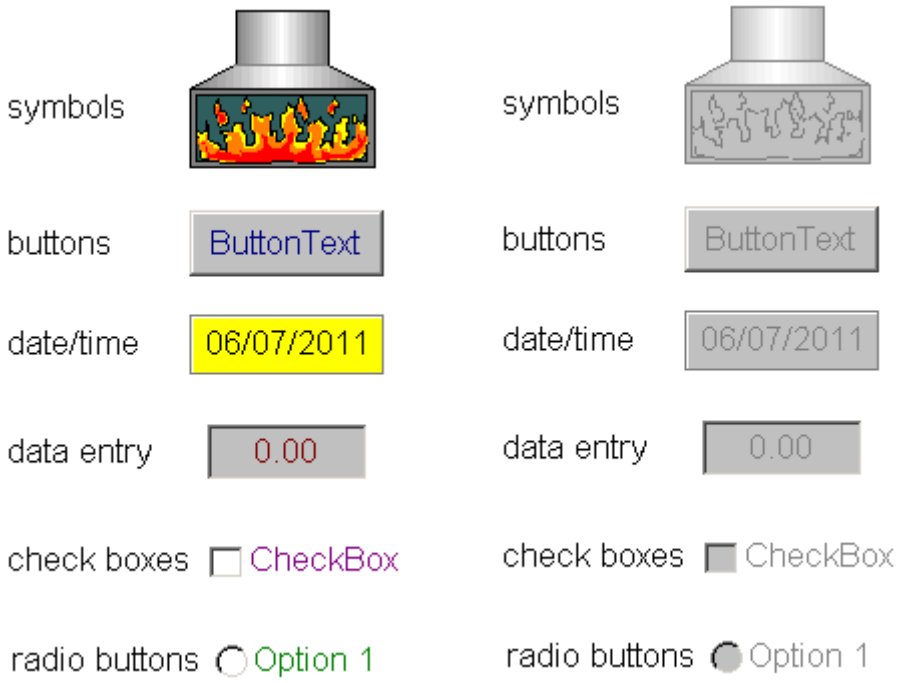


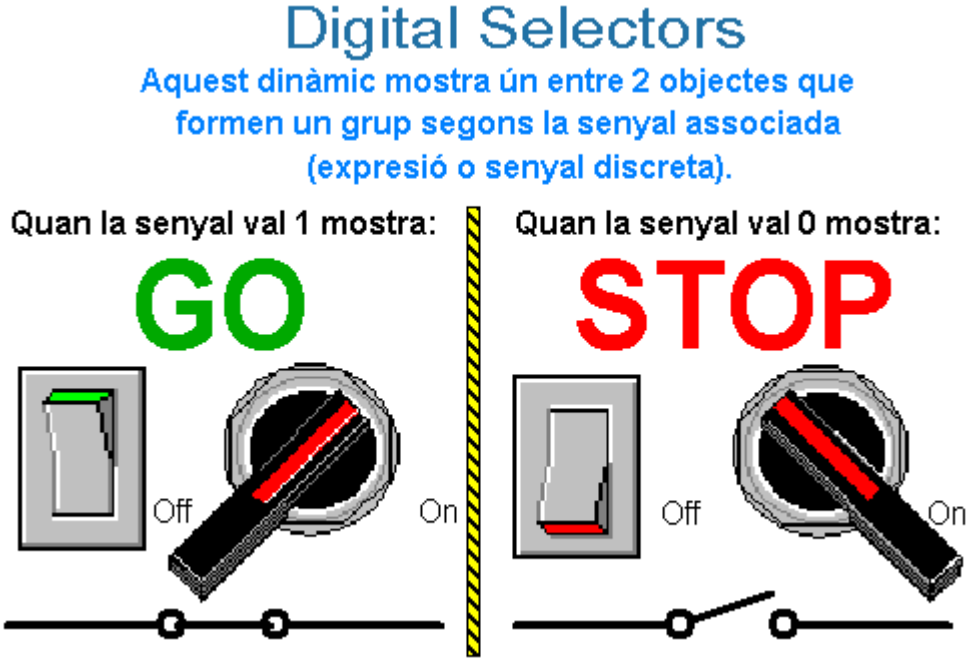
Fig. 2.40 - Visualització d'elements quan estan habilitats o deshabilitats.

Les funcions Color i Color Analògic ens permeten canviar el color d'un element. En el primer cas canvia de color a un altre instantàniament. En el segon cas el color canvia de l'actual al final escalant el canvi de colors. Podem modificar el color de la línia, del interior del dibuix, de la ombra, etc.

L'element de Flash s'encarrega de fer aparèixer-desaparèixer un element amb un ritme constant de temps (en segons) que es pot escollir. Aquesta opció pot donar la sensació de que l'element fa pampallugues.

Quan s'utilitza el Pick el que aconseguim és convertir qualsevol dibuix, imatge, símbol, text, bitmap, metafile, etc. en botó. El tipus de botó que serà s'especifica dins de la finestra que defineix l'objecte.

Els selectors digitals i analògics ens permet utilitzar més d'un element, símbol, dibuix, etc per a crear un visor o selector (digital/analògic respectivament mostrant 1 element segons el senyal).



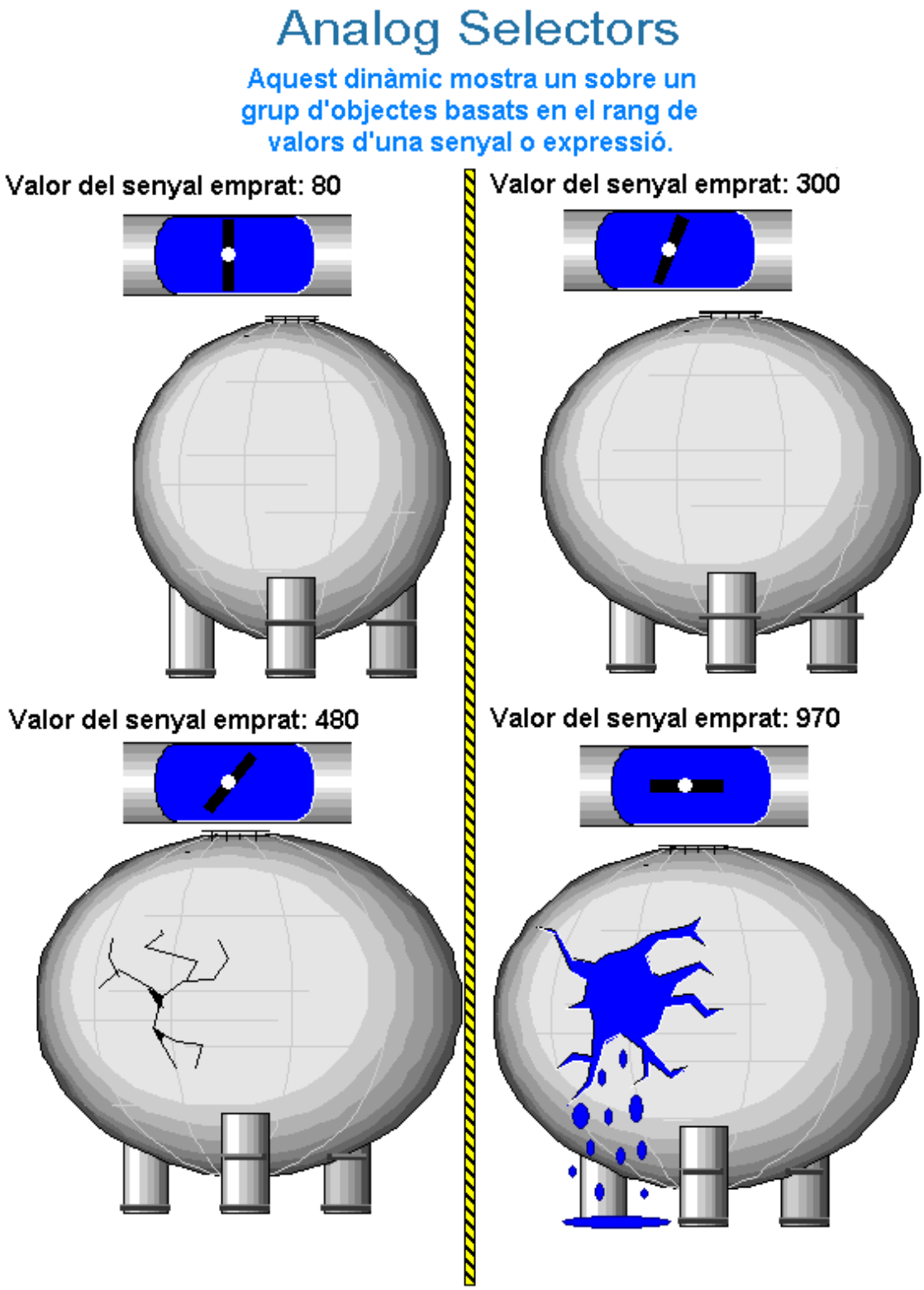


Fig. 2.41 - Exemples de selectors digitals i analògics.

Un dinàmic molt semblant és l'Animador, que s'encarrega de utilitzar un grup o conjunt de dibuixos que s'aniran succeint i repetint al llarg del temps (la freqüència es un valor determinable en aquest dinàmic). De manera que segons els senyals o expressions es succeiran els dibuixos (com els frames d'una pel·lícula) que donaran l'efecte de animació. Això permet la representació més visual, dinàmica i entenedora d'una aplicació.

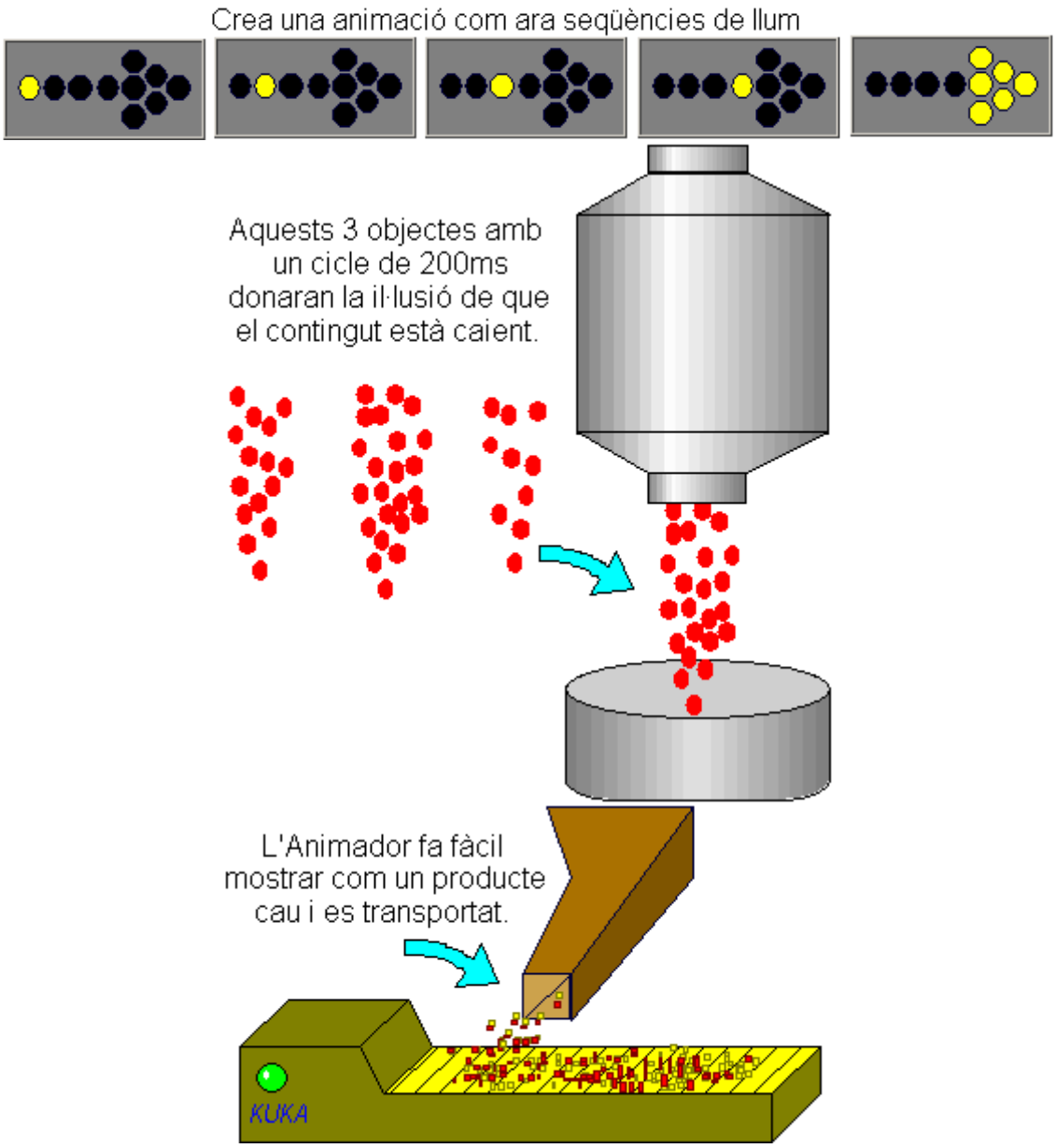


Fig. 2.42 - Aplicacions de l'Animador.

El punt de procés és un dinàmic que permet crear dades numèriques de lectura/escriptura.

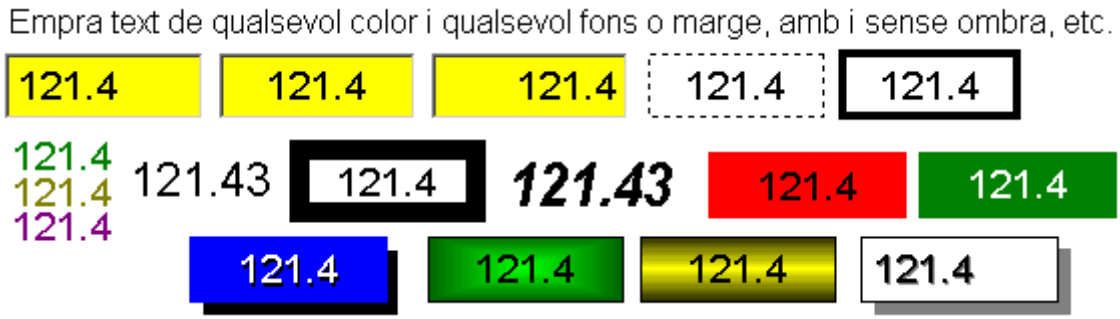


Fig. 2.43 - Exemples de punts de procés.

El dinàmic de data/temps permet visualitzar a la pantalla la hora actual i el dia dins del calendari tenint en compte que hi ha diverses maneres de mostrar aquesta informació (s'indica intrínsecament).

Els botons de checkbox i botó radio fan moltes funcions com els botons (Càrrega de pantalla emergent, anar enrere/següent, llençar una aplicació, descarregar/activar valor, establir un àlies, executar seqüències d'ordres de VBA, etc.). No obstant hi ha una petita diferència entre ells: Els check boxes són controls individuals que poden posar-se individualment a on/off, mentre que els botons radio són controls agrupats on només 1 pot estar activat (on) de tot el grup. Els botons radio a més a més també tenen propietats de grup, i així saben a quin set pertanyen.

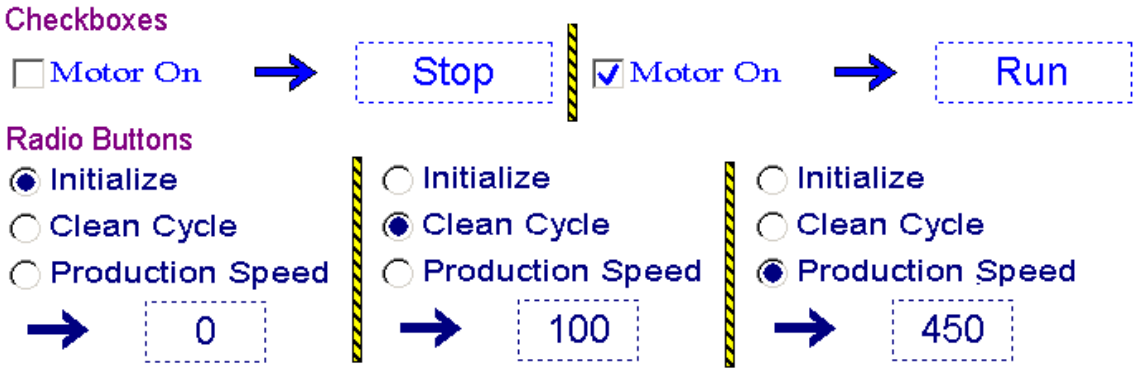
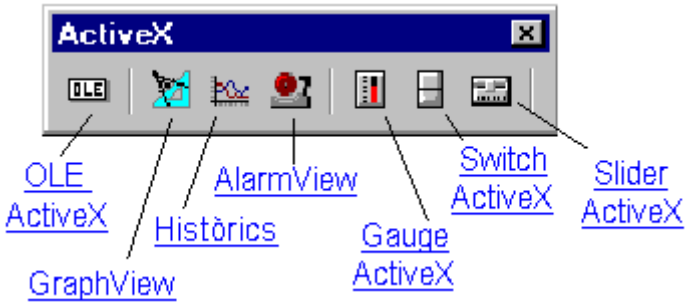


Fig. 2.44 - Exemples de Checkbox i Botó de radio.

El botó display és el darrer dins del menú de dinàmics. Aquesta eina tan pràctica crea automàticament una instantània de la imatge d'un arxiu de pantalla i crea un botó d'enllaç per tal que es vegi. És útil per a mostrar als usuaris on aniran quan estan a punt d'anar a fer clic en un botó per canviar de pantalla (hi ha disponibles 5 mides de botó diferents).

Fig. 2.45 – Exemples de botons display.

El menú més petit és el *menú D'ActiveX*:



El primer botó és el de OLE (Object Linking Embedding) ActiveX. Insereix objectes dins de l'HMI Studio que estan enllaçats a altres programes amb ActiveX (com ara Excel de Microsoft Office). Hi ha moltes altres possibilitats, a part d'inserir programes, com ara inserir vídeos o so.

Nota: Els controls ActiveX són objectes OLE "actius". Elements amb extensió *.ocx*. Aquesta tecnologia està específicament dissenyada per a facilitar la distribució de components a través de xarxes d'alta latència i proporcionen integració dels controls als navegadors web. Considerats com a programari tipus "plug-ins", a qualsevol "contenidor ActiveX": formes de

Visual Basic, pàgines HTML per a Internet Explorer, HMI GraphWorX,etc. Els ActiveX instal·lats es registren a l'ordinador i apareixerà en la llista de OLE.

Per visualitzar els històrics fer doble clic sobre la tendència temporal per mostrar el quadre d'eines flotant, fer clic a la mà per congelar la finestra. Arrossegar per desplaçar, Shift + arrossegar la captura de dades. Hi ha disponibles diversos tipus de gràfics per visualitzar les tendències.

AlarmView:

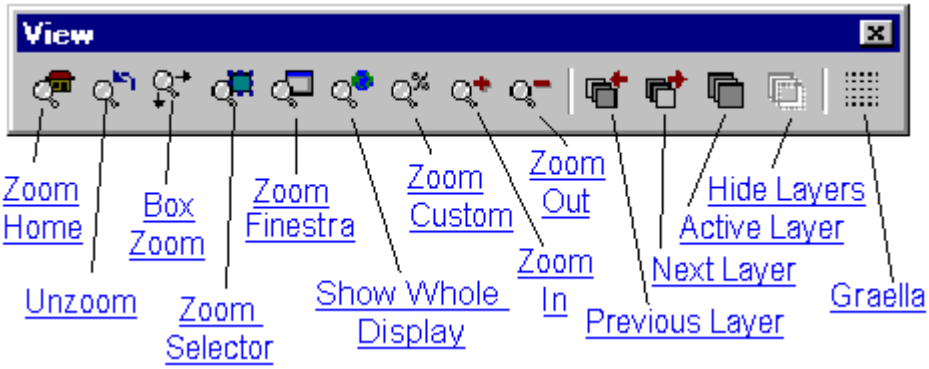
Fig. 2.46 - Característiques de AlarmView

El Mesurador de ICONICS ActiveX ofereix connectivitat OPC amb fabuloses representacions gràfiques de dades en temps real per als operadors. Es poden comprar controls opcionals d'ActiveX de ICONICS (ActiveX Toolbox).

El botó Switch ActiveX d'ICONICS ofereix connectivitat OPC mitjançant un operador fàcil representat per pun botó tipus switch

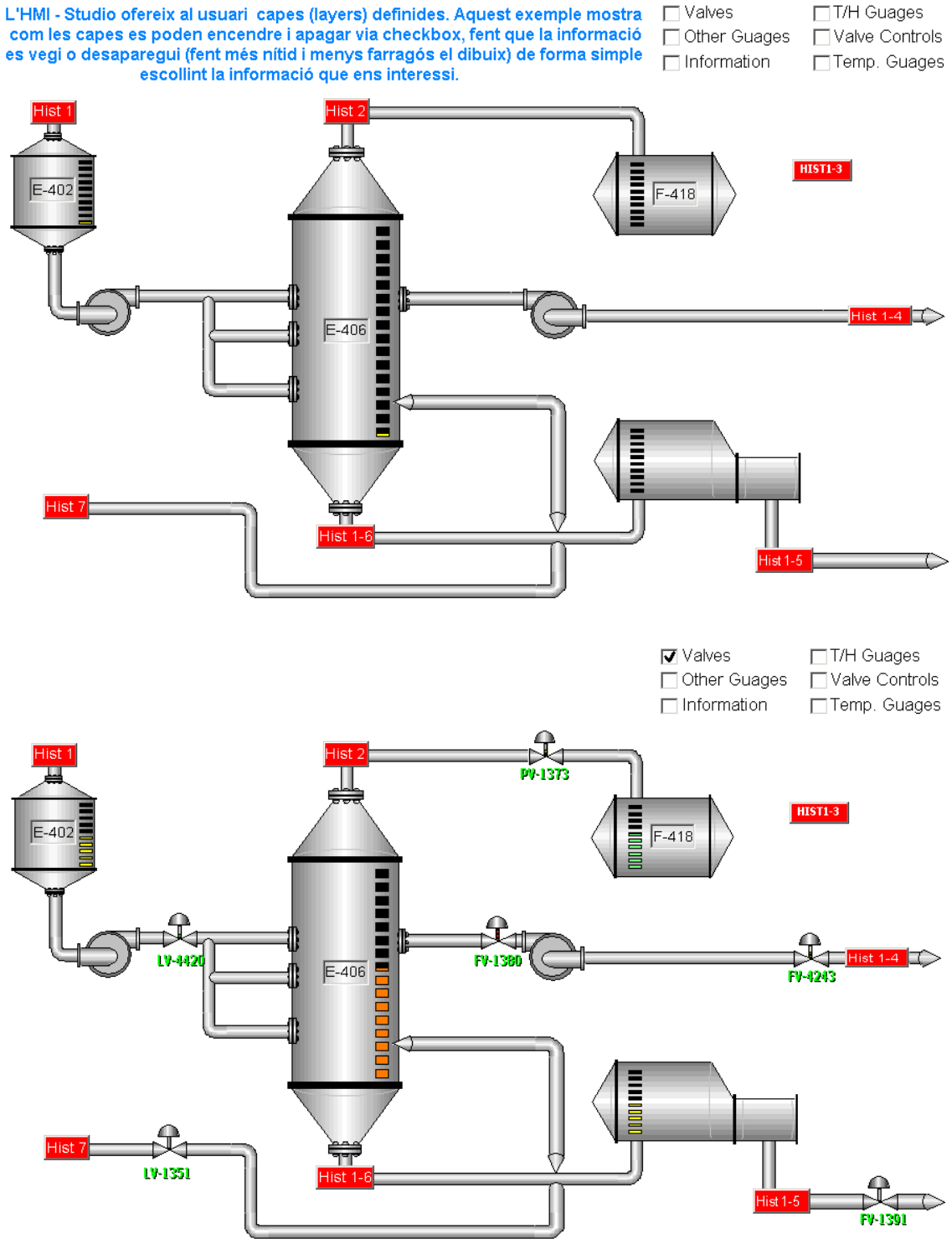
El botó Slider ActiveX d'ICONICS ofereix connectivitat OPC amb un control fàcil de l'operador sobre els valors de dades.

El darrer menú és *el menú de visió*:



Aquests botons són els més familiars per a tothom i són molt intuïtius. El Zoom home és el zoom per defecte, unzoom treu el zoom, mentre que zoom box et permet fer un requadre (aquest requadre és visualitzarà a tota la pantalla), el selector del zoom, zoom del tamany de la pantalla, zoom de la finestra que s'ha programat (si la forma no és la de la pantalla es pot veure deformat), zoom personalitzat en %, zoom in/out afegeix o extreu zoom.

En quant als layers (o capes) tenim els botons que ens permeten moure enrere i endavant les capes, activar la capa i amagar/fer invisible la capa.



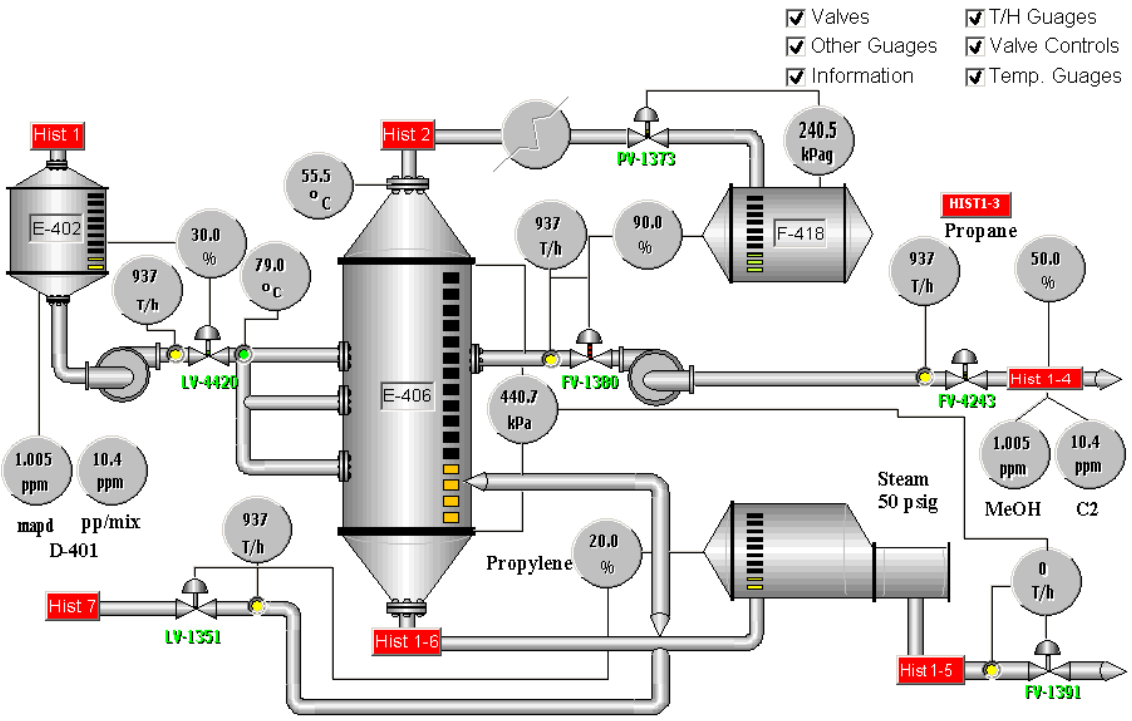


Fig. 2.47 - Exemple de capes que es poden activar/desactivar.

Com podem observar a la figura anterior, posar massa visors pot carregar en excés la pantalla d'informació.

Finalment en aquest menú podem activar o desactivar la graella de punts en el mode de configuració (aquesta graella no és visible al mode RUNTIME).

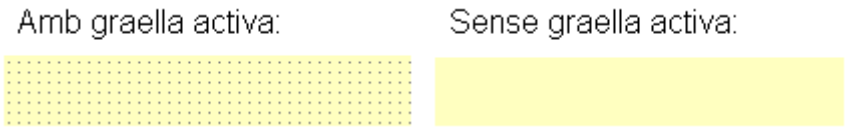
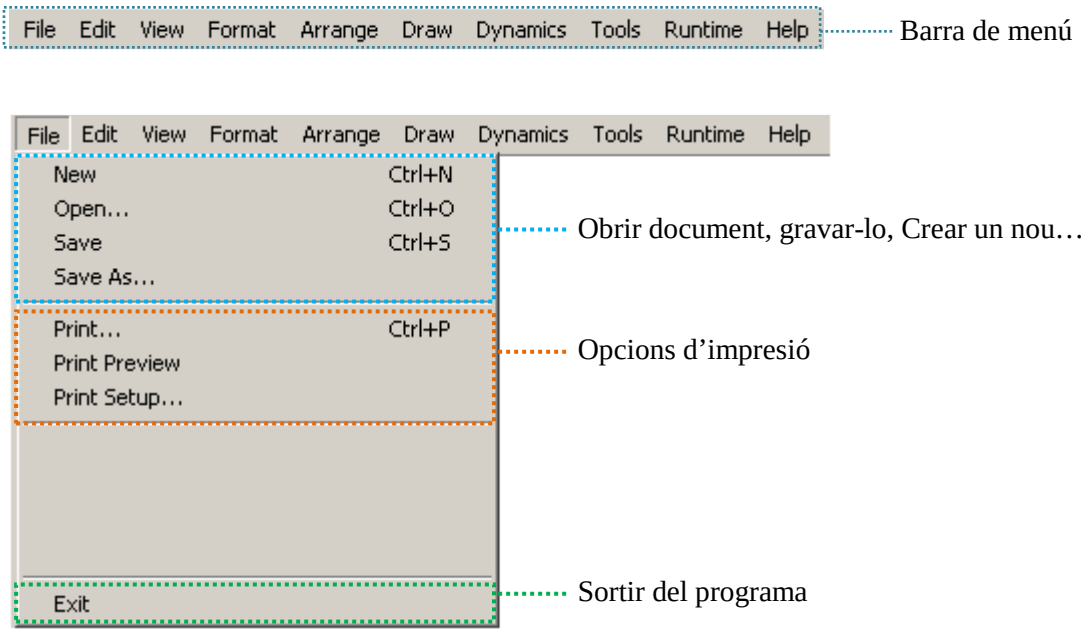
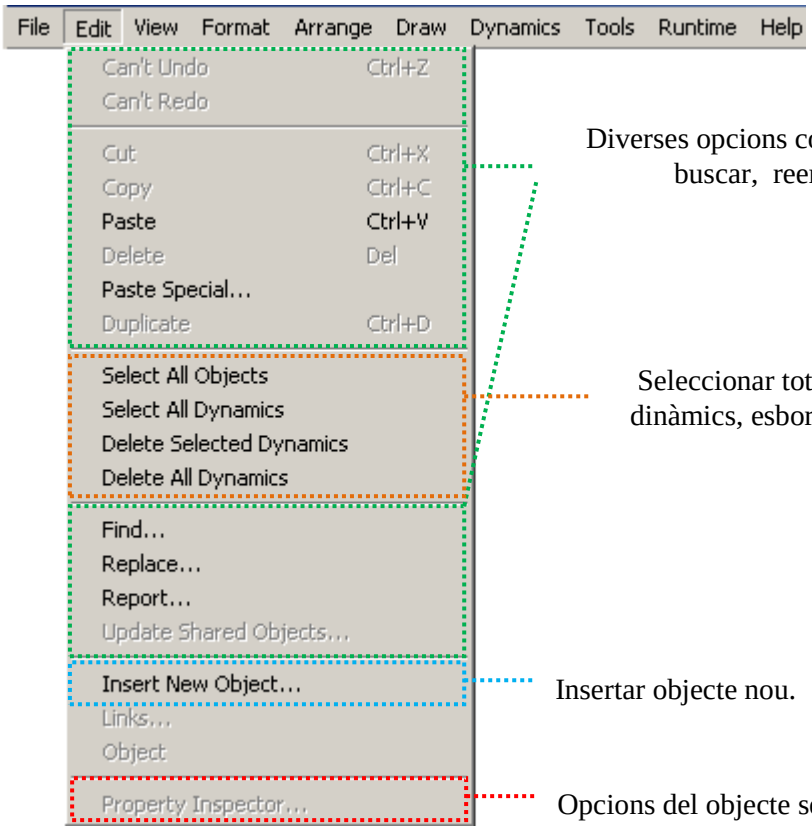


Fig. 2.48 - Graella on/off

Els menús interns del programa amb la descripció són els següents:



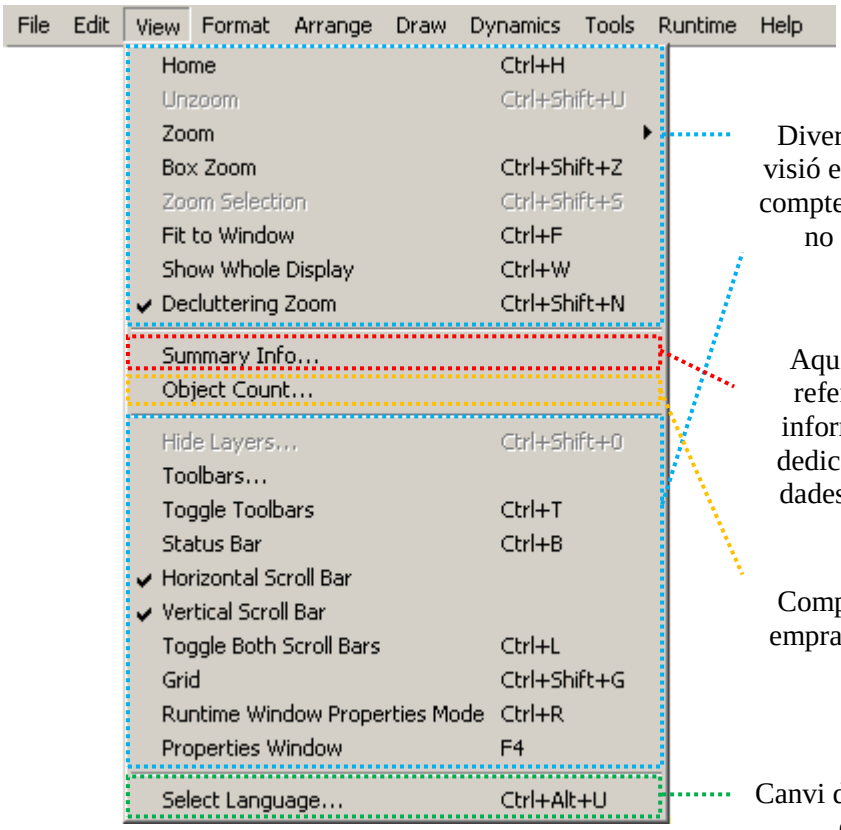


Diverses opcions com copiar, pegar, buscar, reemplaçar...

Seleccionar tots els objectes, tots els dinàmics, esborrar tots els dinàmics...

Insertar objecte nou.

Opcions del objecte seleccionat

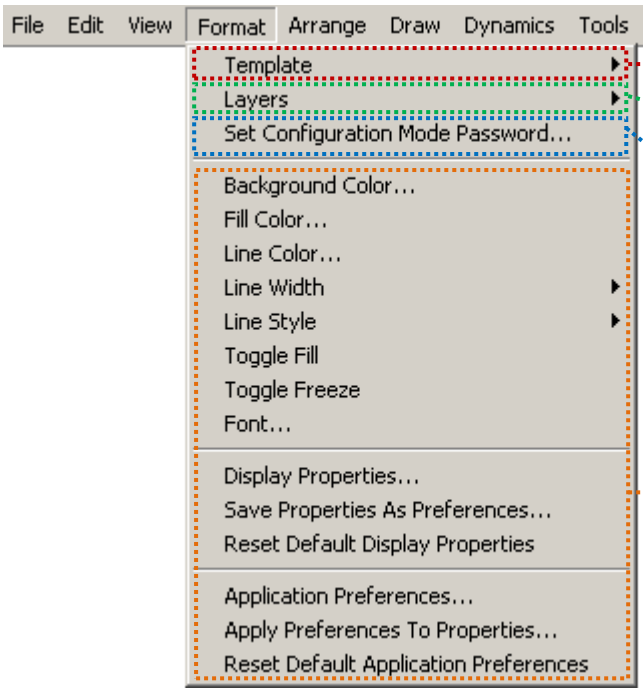


Diverses opcions de visió es poden tenir en compte aquí (mostrar o no , zoom, etc).

Aquí apareix informació referent al projecte, tant informació sobre el temps dedicat en el projecte, com dades afegides per l'autor.

Comptabilitza els objectes emprats segons el seu tipus.

Canvi de l'idioma (només si està instal·lat)

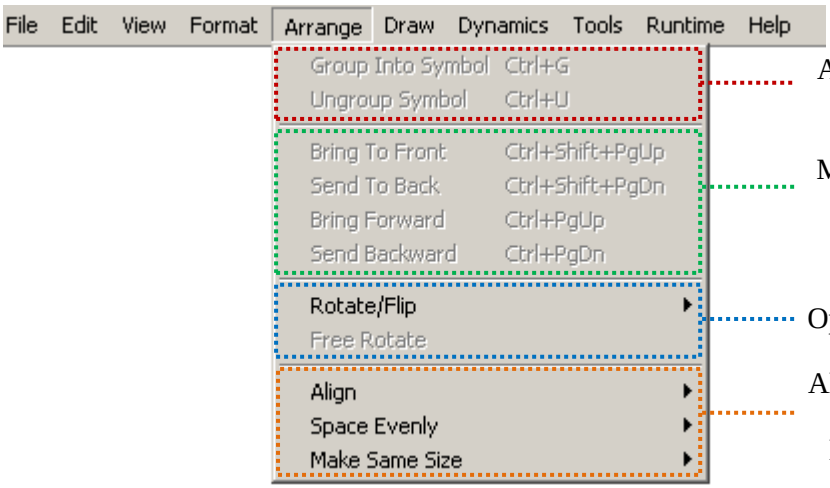


Crear/editar templates.

Ocultar/visualitzar capes.

Aplicar contrasenya a la edició del entorn.

Editar les propietats del display actual.

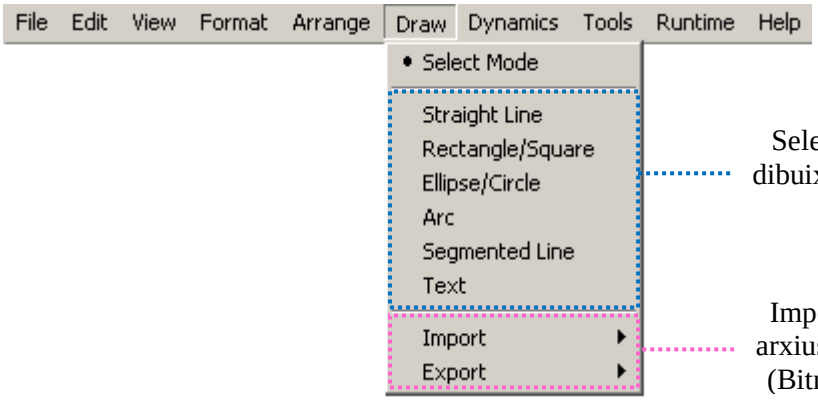


Agrupar/desagrupar un conjunt d'objectes.

Moure l'objecte al front o darrera.

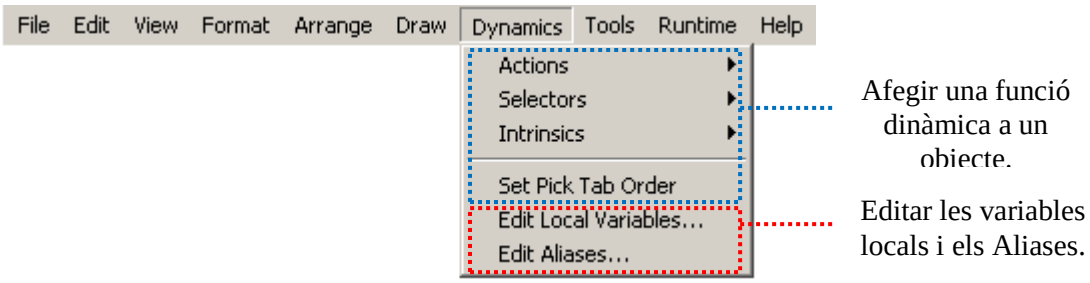
Opcions de rotació.

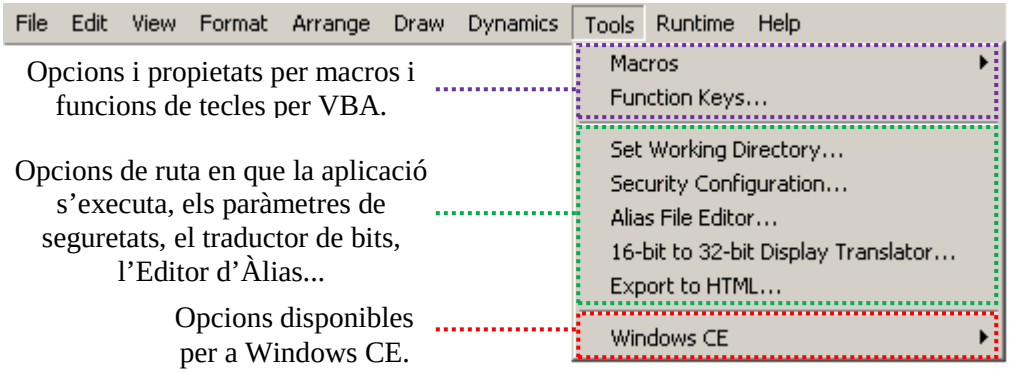
Alinear, canviar mides, i altres efectes segons l'objecte seleccionat.

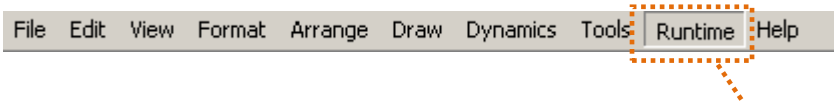


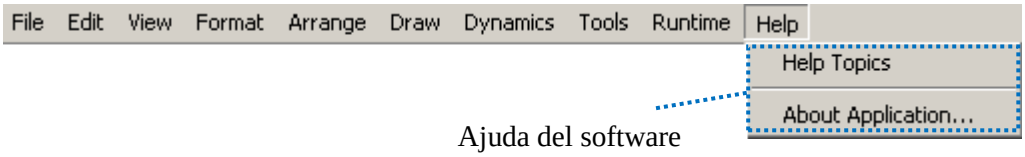
Seleccionar opcions de dibuix per a crear objectes

Importar o exportar arxius de tipus imatge (Bitmap i metafiles)









Previ a la programació, disseny i desenvolupament de la interfície per a arrencar els robots en automàtic extern i la tramesa de dades entre ambdues parts, es van realitzar dos programes de proves generals que em va permetre habituar-me al programari i entendre com funcionava. La primera pantalla de proves era purament de funcionament bàsic del entorn i la segona pantalla de proves ja intentava fer proves amb els robots.

A la figura Fig. 2.49 veurem algunes captures d'una de les pantalles de proves on es mostren alguns dels elements explicats en aquest punt:

Com podem observar podem crear els nostres propis símbols (com ara els dials que apareixen en la pantalla, o el slider de velocitat).

Es poden crear animacions dels objectes (com ara el ventilador).

Es poden associar objectes amb altres objectes (com gairebé tots els elements de la pantalla prova). També es poden associar botons a accions (com els leds vermell i verd).

I finalment, es pot anar a una pantalla diferent (com el botó "Anar a prova 1").

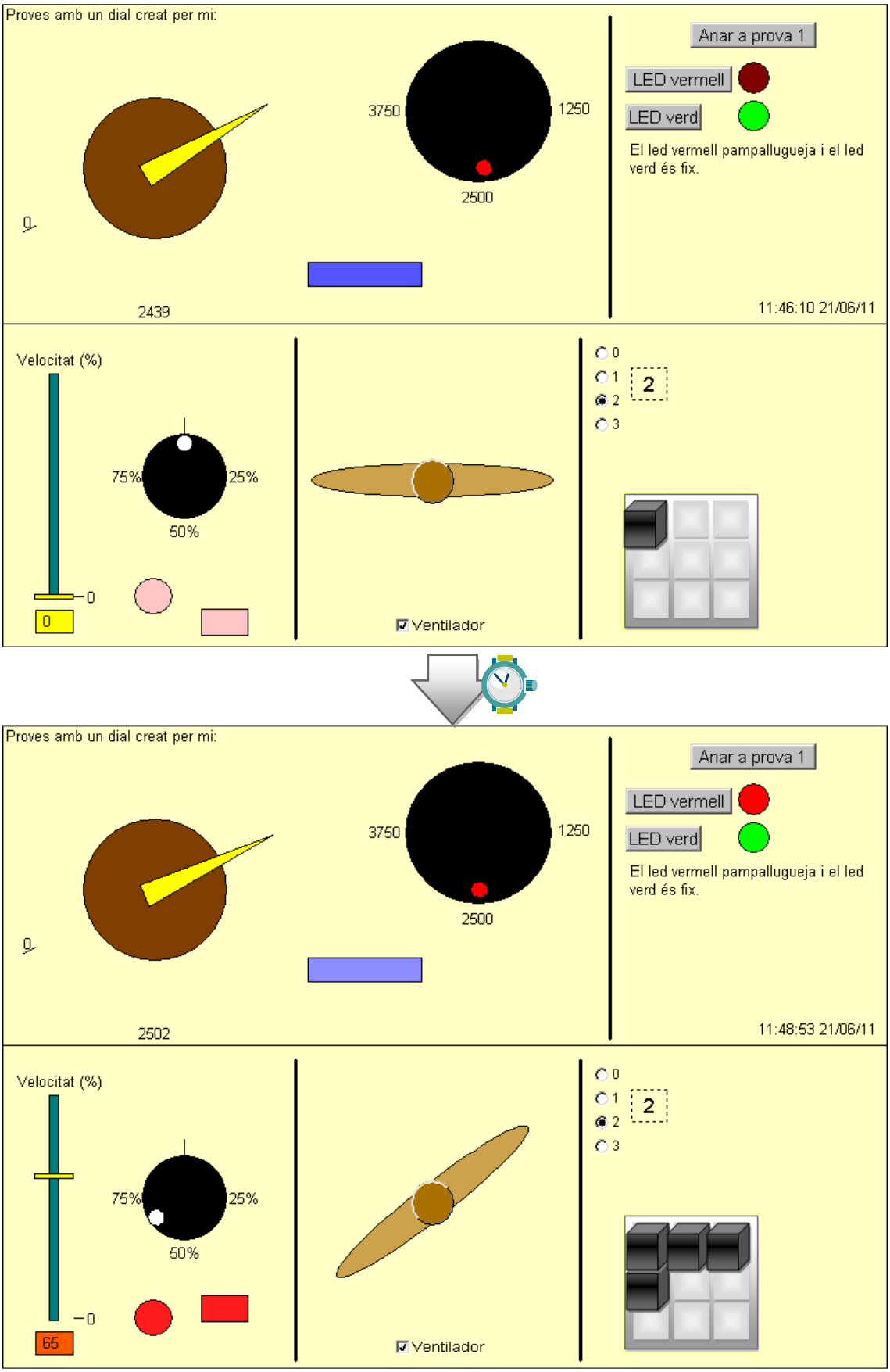


Fig. 2.49 - Execució de la pantalla d'una pantalla de prova

2.6 COMUNICACIONS

Les dades que estem llegint i enviant al robot no són dades crítiques que s'hagin de comprovar en temps real. La velocitat que ens ofereix OPC ja és idònia per el tipus de dades que nosaltres intercanviem.

Les comunicacions tindran lloc entre l'OPC que va inserit dins del HMI Studio (recordem que l'HMI Studio funciona amb items d'OPC o bé amb variables locals) i amb el ProConOS (OPC Server) que incorporen els robots de KUKA. Aquest darrer, està connectat directament amb Soft-PLC.

ProConOS constitueix un sistema d'execució dels programes PLC creats amb Multiprog. El sistema d'execució assumeix les tasques de control i, per això, està equipat amb les corresponents interfaces de comunicació. La tecnologia de compilació integrada permet obtenir màxim rendiment. La tecnologia OPC (OLE for Process Control) es una interfície oberta estàndard que permet intercanviar fàcilment dades entre els equips de la zona de producció i les aplicacions de PC.

El Soft-PLC compren el software encarregat d'executar tasques de control i un PLC, basat únicament en software, integrat en la unitat de control del robot. D'aquesta manera es pot establir una comunicació directa entre el PLC y el robot , alhora que controlar i accedir al robot, la cèl·lula i la línea des d'una única unitat de control. El Soft-PLC de KUKA ofereix un entorn de treball habitual amb superfície Windows, simplificant així tant el maneig com l'aprenentatge i reduint els costos de planificació. La unitat de control integrada ofereix a més a més altres avantatges: la adquisició d'un hardware extern par a visualització i el PLC deixen de ser necessaris i el baix número de components de hardware permet augmentar la fiabilitat, funcionalitat i rendiment del sistema de control.

Multiprog ofereix al usuari un entorn per a la creació de programes i configuracions de control amb amplies funcions de diagnòstic i documentació. La programació es realitza amb llenguatges definits: llista d'instruccions (AWL), diagrama de contactes (KOP), blocs de funcions (FB), text estructurat (ST) i llenguatge d'execució (AS).

Entre el HMI Studio i el robot (amb ProConOs) només cal establir els enllaços que es vulguin considerar dins dels grups que formen cada robot i , si escau, donar permisos d'usuari (dependrà de la configuració). D'aquesta manera ja es comparteixen les dades que han de tenir en comú i s'estableix una comunicació bastant directa.

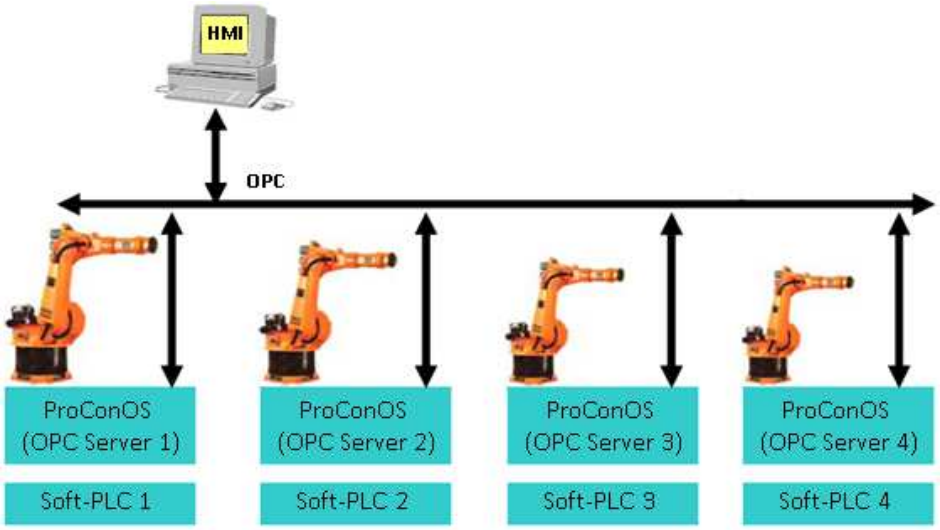


Fig. 2.50 - Esquema de comunicacions.

2.7 PROGRAMA ROBOT

Un robot de KUKA fa servir un llenguatge de programació anomenat KRL que és molt semblant als llenguatges de programació d'alt nivell com ara C. Els programes estan formats per mòduls de 2 arxius (l'arxiu .src i l'arxiu.dat). La programació es realitza sobre el fitxer .src mentre que el fitxer .dat conté les dades dels punts definits, etc.

S'han de tenir, però unes consideracions alhora de programar amb aquest llenguatge que citaré a continuació:

Variables i declaracions: De la mateixa manera que podem fer servir constants (és a dir, la indicació directa del valor en forma de números, símbols, etc...), podem emprar, també, en KRL variables i altres formes de dades.

Per a la programació de robots industrials són necessàries variables, per exemple, per el processament d'un sensor, memoritzar un valor llegit, emprar-les per a fer càlculs d'operacions aritmètiques, etc.

Una variable es representa al programa mitjançant un nom, on la denominació del nom pot seleccionar-se lliurement dins dels límits següents:

- Han de tenir una longitud màxima de 24 caràcters.
- Poden contenir lletres (A-Z), xifres (0-9), així com els símbols "_" i "$".
- No poden començar per xifra ni per "$" (donar que les variables del sistema comencen pel símbol del dòlar, no es fa servir com a primer caràcter en els noms definits per l'usuari).
- No poden ser paraula clau.

Amb el símbol igualtat (=) s'assignen valors a les variables. Una variables sempre ha de declarar-se abans del seu ús.

La durada d'una variable és el temps durant el qual la variable té assignat un lloc a la memòria. Dependrà de si la variable està declarada en un arxiu .SRC o a una llista de dades. En el primer cas la durada de la variable és limitarà al temps de funcionament del programa i en acabar el processament s' alliberarà l'espai en memòria. En el segon cas la durada no depèn del temps o del funcionament del programa, de manera que, la variable existeix mentre existeixi la llista da dades i són, per tant, permanents.

Objectes de dades: Com a objectes de dades s'entenen les unitats de memòria nombrables d'un tipus de dades determinat. Si el programador acorda un objecte de dades a un nom obtenim una variable. Un objecte de dades podria ser un número enter (INTEGER) o caràcter (CHARACTER) i caldria saber de quina manera s'especifica l'objecte de dades.

Tipus de dades	**Enter**	**Real**	**Booleà**	**Caràcter**
Paraula clau	INT	REAL	BOOL	CHAR
Significat	Número enter	Número coma flotant	Estat lògic	1 caràcter
Rang de valors	-2^{31} ... $2^{31}-1$	±1.1E -38 ... ±3.4E +38	TRUE / FALSE	Caràcter ASCII

Fig. 2.51 – Tipus de dades simple

Declaració i inicialització d'objectes de dades: La assignació d'un nom de variable a un tipus de dada i la rserva de la memòria es produeix en KRL mitjançant la declaració DECL, on per exemple:

DECL INT QUANTITAT, NUMERO → identifica, dues variables QUANTITAT i NUMERO del tipus número enter (INTEGER).

Alhora un mateix número es pot assignar de tres maneres diferents (decimal, hexadecimal i binari). Per a poder donar el valor en binari o decimal cal indicar entre comes simples el valor i una B o una H (segons convingui) davant del número. Veiem un exemple:

```
7 VALOR = 90          ;Sistema decimal
8 VALOR = 'B1011010'  ;Sistema binari
9 VALOR = 'H5A'       ;Sistema hexadecimal
```

Després de la declaració d'una variable cal que es prefixi un determinat valor. La primera assignació d'un valor a una variable s'anomena inicialització. La assignació d'un valor a una variable és una instrucció, amb la qual cosa, no pot estar a la secció de declaracions. No obstant la inicialització pot tenir lloc en qualsevol moment a la secció d'instruccions. Però totes les variables s'haurien d'inicialitzar convenientment directament després de la secció de declaració (veure la figura Fig. 2.52):

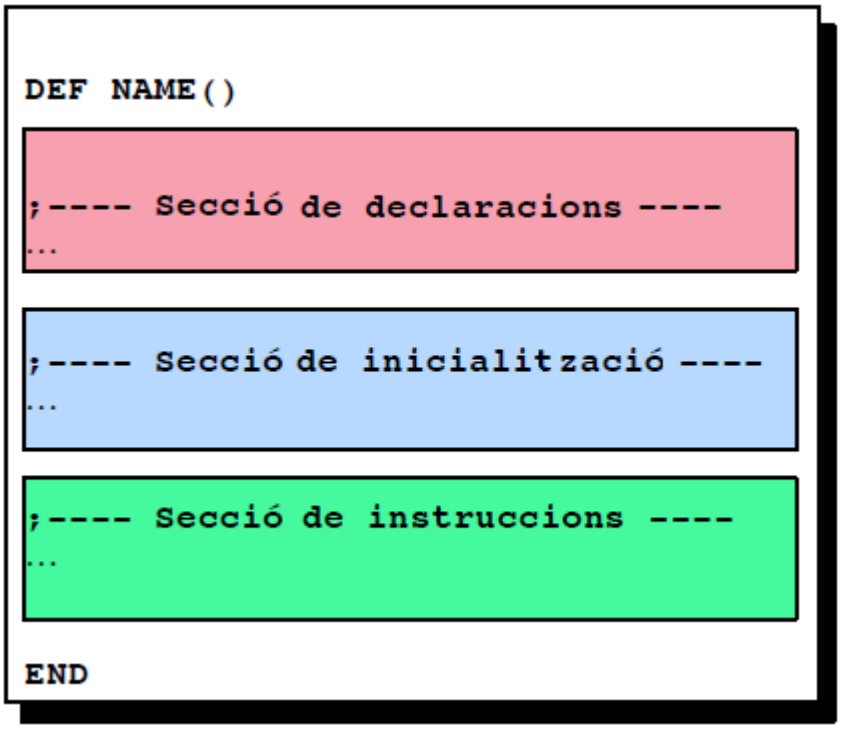


Fig. 2.52 - Estructura bàsica d'un programa del robot

Estructures: Predefinides hi ha les següents estructures:

```
STRUC AXIS    REAL  A1,A2,A3,A4,A5,A6
STRUC E6AXIS  REAL  A1,A2,A3,A4,A5,A6,E1,E2,E3,E4,E5,E6
STRUC FRAME   REAL  X,Y,Z,A,B,C
STRUC POS     REAL  X,Y,Z,A,B,C, INT S,T
STRUC E6POS   REAL  X,Y,Z,A,B,C,E1,E2,E3,E4,E5,E6, INT S,T
```

Si cal crear estructures noves, el llenguatge KRL ho permet. No obstant, ometre'm la informació donat que en cap programa que jo hagi desenvolupat per aquest projecte inclou estructures creades per l'usuari.

Per a llegir les dades d'una estructura només cal introduir el nom de la estructura + "." + el camp de la estructura que volem llegir o escriure. A continuació un exemple:

P1 declarat com a POS:

```
P1 = {X 34.4,Y -23.8,Z 100.0,A 90,B 29.5,C 3.5,S 2 T 6}
```

```
P1.X = 34.3
P1.Y = -23.8
P1.Z = 100.0
P1.A = 90
P1.B = 29.5
P1.C = 3.5
P1.S = 2
P1.T = 6
```

Com podem observar les estructures tipus POS inclouen S i T (Status i Turn). Aquestes son les dades que es guarden en un punt gravat com a PTP. Aquest Status i Turn serveixen per a seleccionar una posició específica, posició inequívoca del robot, on per a diferents posicions dels eixos s'obté el mateix punt a l'espai (veure figura Fig. 2.53).

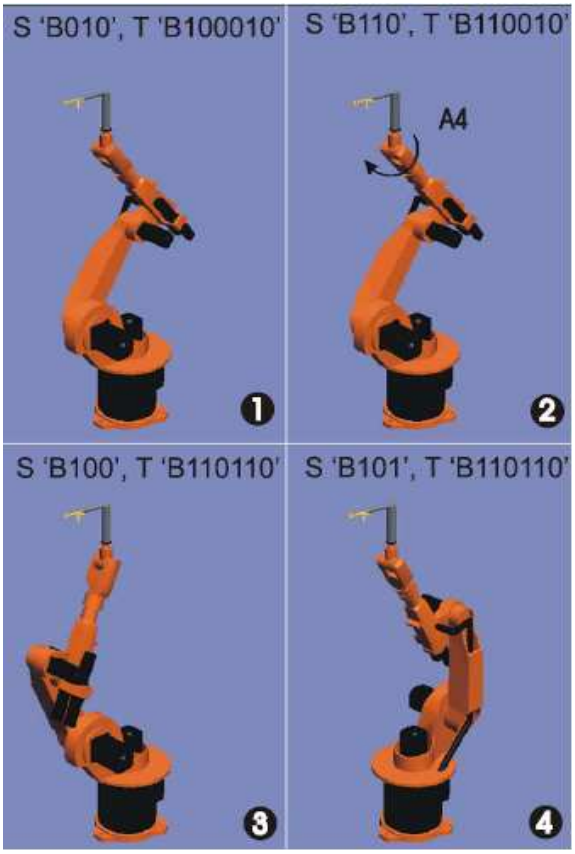


Fig. 2.53 - Exemples de posicionament al mateix punt del espai amb diferents S i T

D'aquesta manera el robot sap exactament amb quina posició mecànica està gravat el punt.

Un cop vist una mica com funciona el llenguatge KRL hauríem de veure quin tipus d'estructuració té la programació i algunes de les instruccions de que disposa el programador alhora de fer un programa. Va la dir que el codi dels programes que jo he realitzat per a aquest projecte així com la seva explicació es troba al capítol 3.1 de la memòria. Abans però veurem quines són algunes de les funcions bàsiques del llenguatge KRL per tal de programar un robot de KUKA.

La estructura de programa sempre és la mateixa:

```
DEF titol_programa ()
INI

...                    → Cos del programa

END
```

Ara mostro quin són els tipus d'instruccions amb que podem omplir el cos del programa en KRL, començant pels tipus de moviments que podem tenir.

2.7.1 Moviments

Disposem de tres tipus de moviments diferents. El robot farà els càlculs pertinents per a calcular la seva trajectòria a partir dels moviments que tingui el programa i del Advance (veure "*Moviments amb aproximació (CONT)*" en aquest mateix subapartat).

Amb la finalitat d'assegurar-se que la posició del robot coincideix amb les coordenades del punt del programa actual, s'executa un funcionament anomenat COI (coincidència de trajectòria). Això és realitza a velocitat reduïda . El robot es mou a les coordenades del punt en el que es situa l'indicador de bloc (COI).

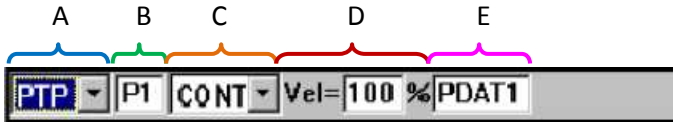
Cal fer COI després de seleccionar el programa CELL i abans de que el mode automàtic extern arrenqui. Per a fer COI realitzarem un PTP a HOME. Per defecte hi ha 6 posicions de HOME diferents. Generalment s'utilitzà més la posició de HOME1 que és per defecte la posició de zero mecànic del robot (que recordem és A1=0º, A2=-90º, A3=90º, A4=0º, A5=0º i A6=0º).

Un punt HOME es recomana posar-lo al inici i final del programa ja que aquest representa una posició inequívocament definida.

Moviment específic dels eixos PTP

La eina es mou al llarg de la trajectòria més ràpida al punt final.

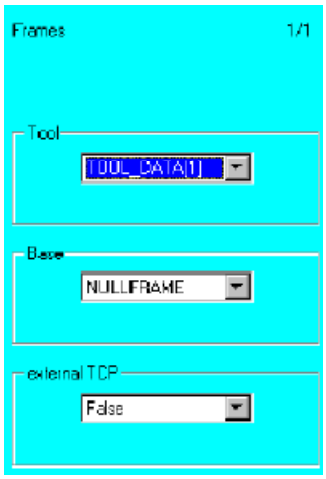
Per a introduir punts acostuma a ser útil emprar el formulari inline. Aquest formulari ens permet modificar els paràmetres d'una instrucció de manera que no hagis de recordar la seva estructura. Has de decidir però quin tipus de moviment s'escull, si té o no contorn (veurem els seus efectes una mica més endavant), indicar la velocitat en aquell punt i els altres paràmetres de moviment (com ara especificar el moviment relatiu a quina eina, base i si es treballa amb Eina externa o TCP Extern).



	Den.camp	Funció	Àrea de valors
A	**PTP**	Tipus de movimient	PTP, LIN, CIRC
B	**P1**	Denominació del punt	màxim 23 caràcters
	Tool	Eina nro	Nullframe, Tool_Data[1]...[16]
	Base	Peça nro	Nullframe, Base_Data[1]...[32]
	external TCP	El robot condueix la eina / la peça	True, False
C	**CONT**	Posicionamient aproximat actiu	" ", Cont
D	**Vel=100%**	Velocitat	1 a 100% del valor màxim (predefinició 100%)
	PDAT1	Paràmetre de movimient	
E	Acceleration	Acceleració	0 ... 100
	Approximation Distance *	Àrea de aproximació	0 ... 100

A → Indiquem tipus de moviment *PTP* (Point-to-point), en aquest cas.

B → Denominació del punt (si tenim el robot situat al punt podem fer TOUCH UP i guardarà les dades d'aquest punt segons la eina, base i TCP extern).



TOOL: Seleccionar entre les 16 eines que es poden determinar (ja cal tenir fet el TCP offset abans de seleccionar la eina).

BASE: Aquí cal escollir entre les 32 bases de coordenades que es poden determinar (cal tenir les bases entrades abans de escollir-la).

External TCP: Cal informar a la unitat de control si el robot condueix eina o peça.

El robot condueix la eina: external TCP = False
El robot condueix la peça: external TCP = True

C → Aproximació activada o desactivada.

 *Moviments PTP amb parada exacta:* En moviments PTP amb parada exacta, el robot es desplaça al punt de destinació amb posicionament exacte.

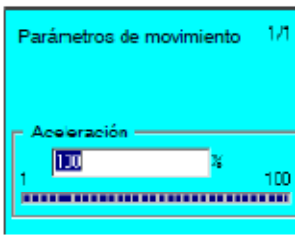
Moviments PTP amb posicionament aproximat: Quan s'introdueix una aproximació el moviment del robot passa al punt de destinació de la següent instrucció vorejant el punt indicat a la instrucció. (Per a major enteniment veure el apartat "*Moviments amb aproximació (CONT)*" d'aquest mateix apartat).

D → Velocitat de trajectòria

Com que es tracta d'un punt PTP no podem determinar la velocitat exacta donat que no coneixem la trajectòria que el robot calcularà com la més ràpida. No obstant, si podem parametritzar de manera percentual aquesta velocitat. Si tenim en compte que el 100% de la

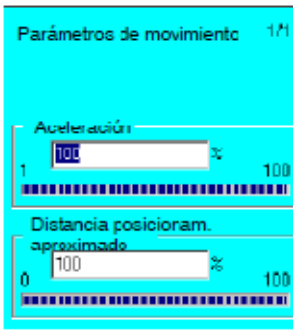
velocitat equival a la velocitat més ràpida que el robot pugui abastar, podem mirar de reduir aquesta velocitat (fins a un mínim d'un 1%) segons les nostres necessitats de programa.

E → Parametrització del moviment.



Aceleración: Aquí es pot reduir la acceleració que s'utilitza durant el moviment.

Segons la longitud tel recorregut, el valor de la acceleració i el valor del entorn d'aproximació, podria ser que no pugui abastar la velocitat de desplaçament programada.



Aquest pot ser el cas, quan els eixos del canell del robot s'han de moure infinitament ràpid degut al pas a través de la posició estesa, i per aquesta raó, es superen els valors màxims admissibles. Parar atenció a la utilització de valors pràctics abastables.

Distancia de posicionamiento aproximado: A quí es pot reduir el entorn de posicionament aproximat que es fa servir durant el moviment (en tant per cent i només si CONT està actiu).

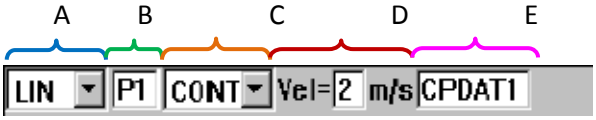
NOTA: Recordem que les variables "S" (Status) i "T" (Turn) La especificació del estat del Status i Turn només s'avalua en els moviments PTP. Aquesta és la raó per la qual el primer moviment d'un programa ha de ser PTP.

Moviment relatiu a la trajectòria - linear LIN

La eina és guiada a una velocitat definida (coordinant els eixos entre sí) al llarg d'una línia recta.

Dins el moviment linear i circular podem determinar la velocitat constant del moviment en m/s (fins a un màxim de 2m/s). Els moviments lineals s'utilitzen quan es requereix per a la aproximació d'un punt un guiat de recorregut exacte amb velocitat predeterminada.

Com en tots els moviments podem emprar el formulari inline per ajudar-nos a introduir la instrucció sense oblidar cap paràmetre.

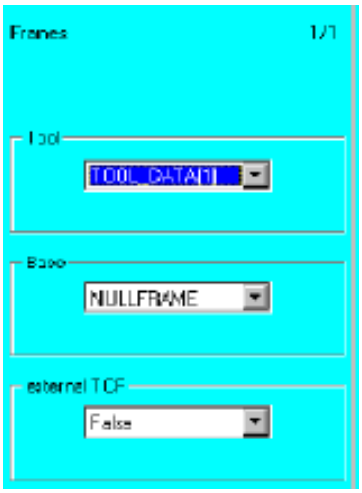


	Den.camp	Funció	Àrea de valors
A	**LIN**	Tipus de movimient	PTP, LIN, CIRC
B	**P1**	Denominació del punt	màxim 23 caràcters
	Tool	Eina nro	Nullframe, Tool_Data[1]...[16]
	Base	Peça nro	Nullframe, Base_Data[1]...[32]
	external TCP	El robot condueix la eina / la peça	True, False

C	**CONT**	Posicionamient aproximat actiu	" ", Cont
D	**Vel=2m/s**	Velocitat	0,001 ... 2m/s (predefinició 2m/s)
E	**CPDAT1**	Paràmetre de movimient	
	Acceleration	Acceleració	0 ... 100
	Approximation Distance *	Àrea de aproximació	0 ... 300

A → Indiquem tipus de moviment *LIN* (Linear), en aquest cas.

B → Denominació del punt (si tenim el robot situat al punt podem fer TOUCH UP i guardarà les dades d'aquest punt segons la eina, base i TCP extern).



TOOL: Seleccionar entre les 16 eines que es poden determinar (ja cal tenir fet el TCP offset abans de seleccionar la eina).

BASE: Aquí cal escollir entre les 32 bases de coordenades que es poden determinar (cal tenir les bases entrades abans de escollir-la).

External TCP: Cal informar a la unitat de control si el robot condueix eina o peça.

El robot condueix la eina: external TCP = False
El robot condueix la peça: external TCP = True

C → Aproximació activada o desactivada.

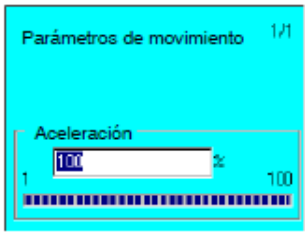
 *Moviments LIN amb parada exacta:* En moviments LIN amb parada exacta, el robot es desplaça a cadascun dels punts de destinació amb posicionament exacte.

Moviments LIN amb posicionament aproximat: Quan s'introdueix una aproximació el moviment del robot passa al punt de destinació de la següent instrucció vorejant el punt indicat a la instrucció. (Per a major enteniment veure el apartat "*Moviments amb aproximació (CONT)*" d'aquest mateix apartat).

D → Velocitat de trajectòria

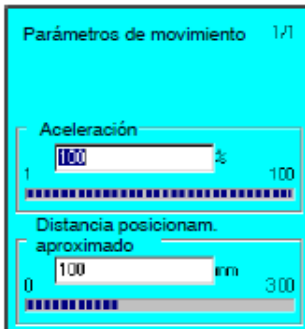
Aquí es determina la velocitat amb la qual el robot ha d'executar el moviment. Pot introduir el valor a través del teclat o modificar-lo mitjançant la tecla de la funció d'estat que es troba a la dreta del display.
Depenent de la longitud del camí recorregut, la magnitud de la acceleració i el programa d'aproximació a l'entorn, podria passar que la velocitat de desplaçament programada no es pugui abastar.

E → Parametrització del moviment.



Aceleración: Aquí es pot reduir la acceleració que s'utilitza durant el moviment.

Segons la longitud tel recorregut, el valor de la acceleració i el valor del entorn d'aproximació, podria ser que no pugui abastar la velocitat de desplaçament programada.



Aquest pot ser el cas, quan els eixos del canell del robot s'han de moure infinitament ràpid degut al pas a través de la posició estesa, i per aquesta raó, es superen els valors màxims admissibles. Parar atenció a la utilització de valors pràctics abastables.

Distancia de posicionamiento aproximado: A quí es pot reduir el entorn de posicionament aproximat que es fa servir durant el moviment (en mil·límetres i només si CONT està actiu).

Quan efectuem un moviment del tipus lineal no només cal tenir en compte la trajectòria sinó també cal establir un format d'orientació. En el cas dels moviments linears tenim els següents:

Control d'orientació – Standard: Durant el moviment en una trajectòria, la orientació de la eina canvia contínuament de la posició inicial a la posició final. Això s'aconsegueix rotant i girant la direcció de la eina.

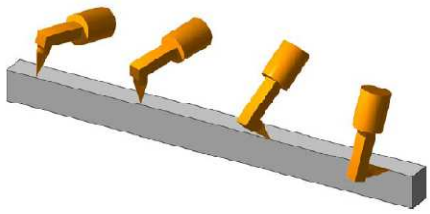


Fig. 2.54 - Moviment lineal amb control d'orientació Standard

Control d'orientació – Wrist PTP: Durant el moviment en una trajectòria la orientació de la eina canvia contínuament de la posició inicial a la posició final. Això succeeix per la transformació lineal (moviment específic de l'eix) dels angles de l'eix del canell del robot. El problema de la singularitat 3 (posició α5) es pot evitar emprant aquesta opció, donat que no hi ha control de la orientació rotant i girant la direcció de la eina.

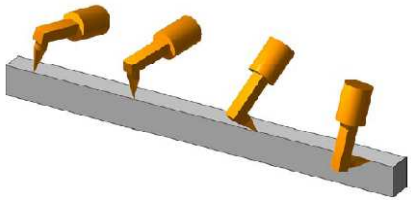


Fig. 2.55 –Moviment lineal amb control d'orientració Wrist PTP

Control d'orientació – Constant: La orientació programada en el primer punt es manté des del punt inicial fins al punt final.

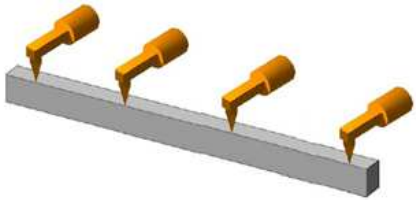
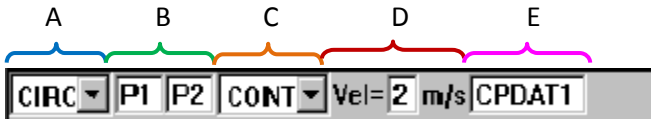


Fig. 2.56 - Moviment lineal amb control d'orientació Constant

Moviment relatiu a la trajectòria - circular CIRC

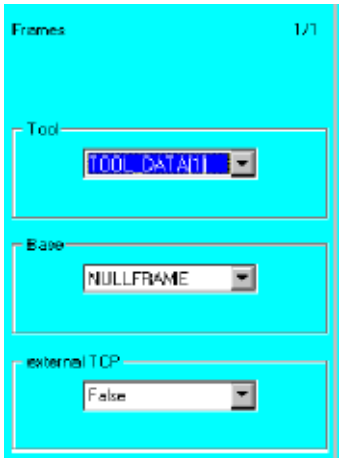
La eina és guiada a una velocitat definida al llarg de la trajectòria circular.



	Den.camp	Funció	Àrea de valors
A	**CIRC**	Tipus de movimient	PTP, LIN, CIRC
B	**P1**	Denominació del punt auxiliar	màxim 23 caràcters
B	**P2**	Denominació del punt	màxim 23 caràcters
	Tool	Eina nro	Nullframe, Tool_Data[1]...[16]
	Base	Peça nro	Nullframe, Base_Data[1]...[32]
	external TCP	El robot condueix la eina / la peça	True, False
C	**CONT**	Posicionamient aproximat actiu	" ", Cont
D	**Vel=2m/s**	Velocitat	0,001 ... 2m/s (predefinició 2m/s)
E	**CPDAT1**	Paràmetre de movimient	
	Acceleration	Acceleració	0 ... 100
	Approximation Distance *	Àrea de aproximació	0 ... 300

A → Indiquem tipus de moviment *CIRC* (Circular), en aquest cas.

B → Denominació del punt (si tenim el robot situat al punt podem fer TOUCH UP i guardarà les dades d'aquest punt segons la eina, base i TCP extern).



TOOL: Seleccionar entre les 16 eines que es poden determinar (ja cal tenir fet el TCP offset abans de seleccionar la eina).

BASE: Aquí cal escollir entre les 32 bases de coordenades que es poden determinar (cal tenir les bases entrades abans de escollir-la).

External TCP: Cal informar a la unitat de control si el robot condueix eina o peça.

El robot condueix la eina: external TCP = False
El robot condueix la peça: external TCP = True

C → Aproximació activada o desactivada.

Moviments CIRC amb parada exacta: En moviments CIRC amb parada exacta, el robot es desplaça a cadascun dels punts de destinació amb posicionament exacte.

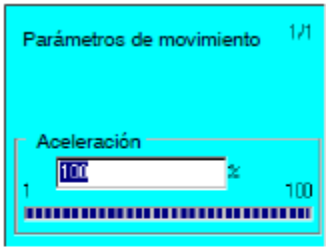
Moviments CIRC amb posicionament aproximat: Quan s'introdueix una aproximació el moviment del robot passa al punt de destinació de la següent instrucció vorejant el punt indicat a la instrucció. (Per a major enteniment veure el apartat "*Moviments amb aproximació (CONT)*" d'aquest mateix apartat).

D → Velocitat de trajectòria

Aquí es determina la velocitat amb la qual el robot ha d'executar el moviment. Pot introduir el valor a través del teclat o modificar-lo mitjançant la tecla de la funció d'estat que es troba a la dreta del display.

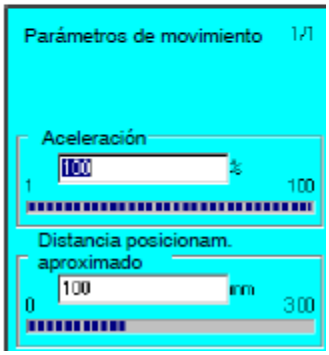
Depenent de la longitud del camí recorregut, la magnitud de la acceleració i el programa d'aproximació a l'entorn, podria passar que la velocitat de desplaçament programada no es pugui abastar.

E → Parametrització del moviment.



Aceleración: Aquí es pot reduir la acceleració que s'utilitza durant el moviment.

Segons la longitud tel recorregut, el valor de la acceleració i el valor del entorn d'aproximació, podria ser que no pugui abastar la velocitat de desplaçament programada.



Aquest pot ser el cas, quan els eixos del canell del robot s'han de moure infinitament ràpid degut al pas a través de la posició estesa, i per aquesta raó, es superen els valors màxims admissibles. Parar atenció a la utilització de valors pràctics abastables.

Distancia de posicionamiento aproximado: A quí es pot reduir el entorn de posicionament aproximat que es fa servir durant el moviment (en mil·límetres i només si CONT està actiu).

Quan efectuem un moviment del tipus circular no només cal tenir en compte la trajectòria sinó també cal establir un format d'orientació. En el cas dels moviments circulars tenim els següents:

Control d'orientació – Standard: Durant el moviment en una trajectòria, la orientació de la eina canvia contínuament de la posició inicial a la posició final. Això s'aconsegueix rotant i girant la direcció de la eina.

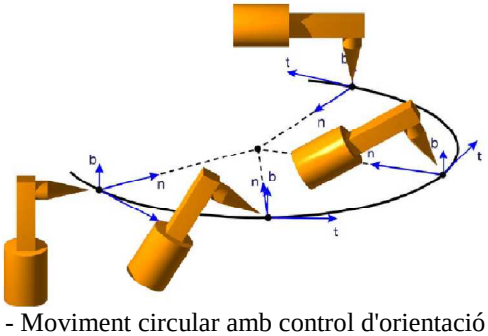


Fig. 2.57 - Moviment circular amb control d'orientació Standard

Control d'orientació – Wrist PTP: Durant el moviment en una trajectòria la orientació de la eina canvia contínuament de la posició inicial a la posició final. Això succeeix per la transformació lineal (moviment específic de l'eix) dels angles de l'eix del canell del robot. El problema de la singularitat 3 (posició α5) es pot evitar emprant aquesta opció, donat que no hi ha control de la orientació rotant i girant la direcció de la eina.

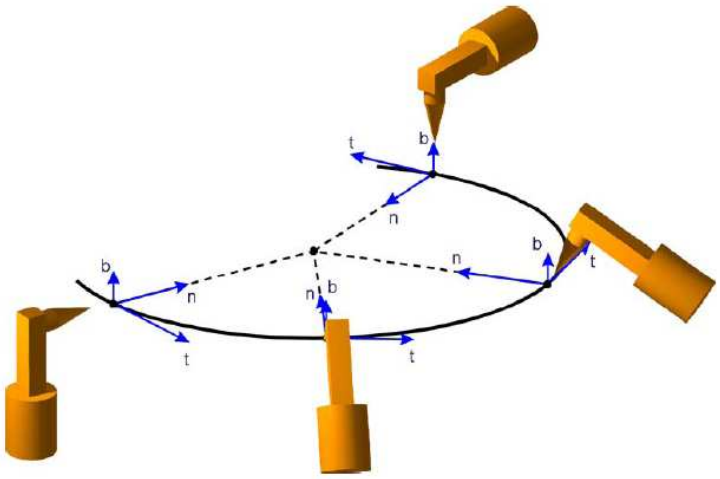


Fig. 2.58 - Moviment circular amb control d'orientació Wrist PTP

Control d'orientació – Constant: La orientació programada en el primer punt es manté des del punt inicial fins al punt final.

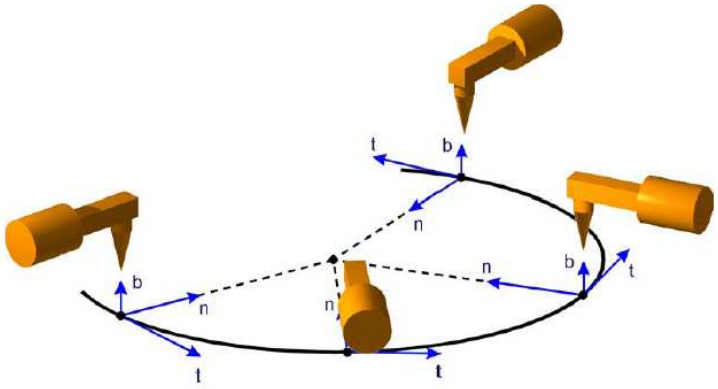


Fig. 2.59 - Moviment circular amb control d'orientació Constant

Moviments amb aproximació (CONT): Són exactament els mateixos moviments anteriors però si hem seleccionar el contorn (CONT) veurem que el moviment es veu afectat considerablement. Durant una aproximació, el robot no es mou exactament a cada posició programada, ni es frena totalment,

En general el temps de cicle millorarà si emprem moviments aproximats (donat que el robot no s'atura justament en arribar al punt seleccionat i aprofita el moviment que ja tenia del punt anterior per dirigir-se al següent) i també es redueix el desgast.

Com veiem gràficament en la figura següent (Fig. 2.60) quan no s'aproxima cap punt cal accelerar i desaccelerar en cada punt de manera que el robot s'atura en el punt exacte, i això fa que es perdi molt temps de cicle. En el moment que aproximem encara que siguin 2 punts (segon la gràfica B) veiem que el temps millora considerablement.

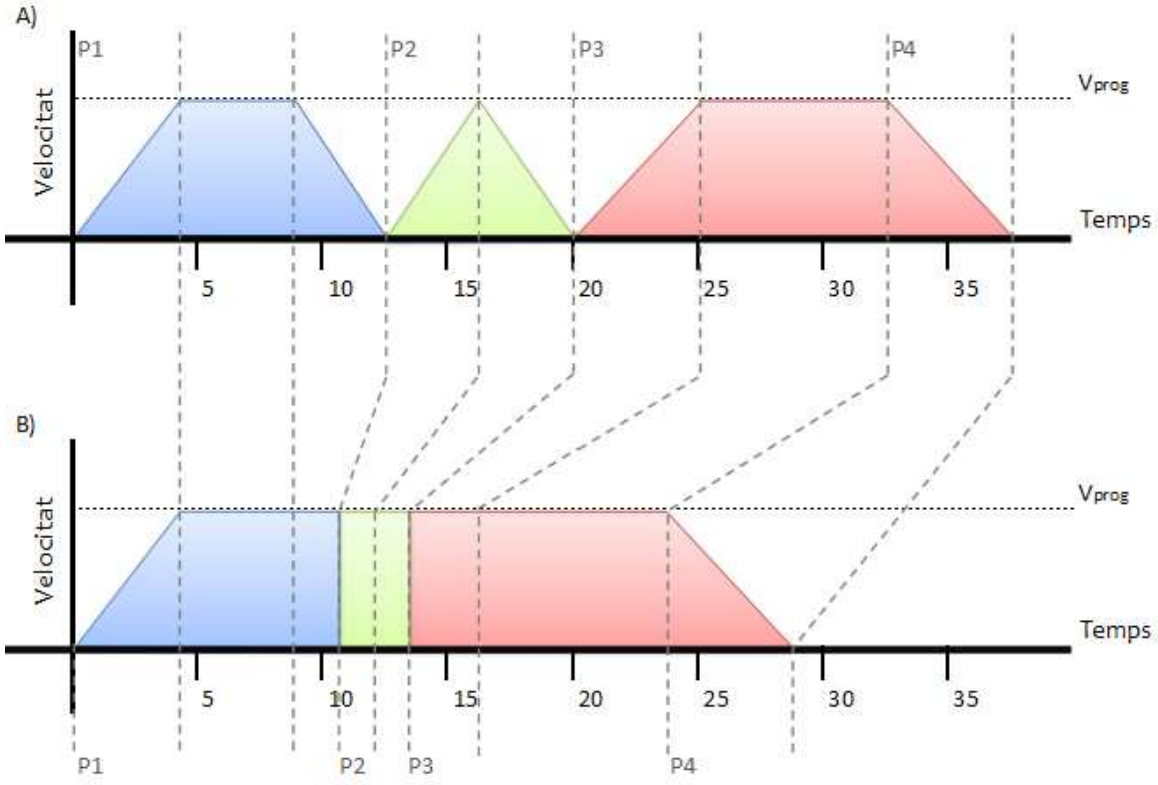


Fig. 2.60 – Gràfiques velocitat/temps. A) sense aprox. de punts B) Amb aproximació de punts

Abans de veure les aproximacions de moviments, introduirem el concepte de l'advance donat que les trajectòries resultants de cada moviment poden veure's afectats a causa seva.

El punter de programa (línia blanca), que es pot veure per pantalla en executar un programa, indica sempre el bloc que està processant el robot actualment. El punter del Advance, d'altre banda, no és visible i va tres línies de moviment davant del punter de programa (per defecte es pren 3, però el valor es pot determinar entre 0 i 5).Veure figura Fig. 2.61.

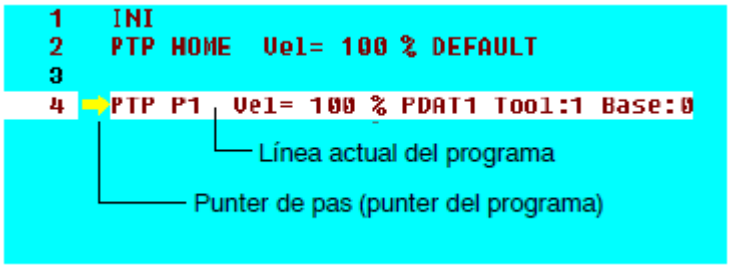


Fig. 2.61 - Exemple de punter de programa i linia actual

Per a poder calcular la trajectòria d'un moviment d'aproximació és necessari llegir i calcular la trajectòria abans que el punter de programa. Però l'Advance no només processa les trajectòries, també processa dades i comandes lògiques per a controlar la perifèria.

Ja veurem que això pot arribar a influir en el comportament amb els moviments. Existeixen també instruccions que aturen el punter del Advance (com ara instruccions d'entrades/sortides). Si l'Advance s'atura apareixerà un missatge per la pantalla que indicarà "*Aproximación no posible*" (només en els modes de test: T1 i T2).

- Veiem primer els moviments d'aproximació per PTP:

En les aproximacions PTP no sabem mai per a on li anirà millor al robot fer el contorn degut a que es regeix per la velocitat màxima fent que els eixos es moguin més aviat poc.

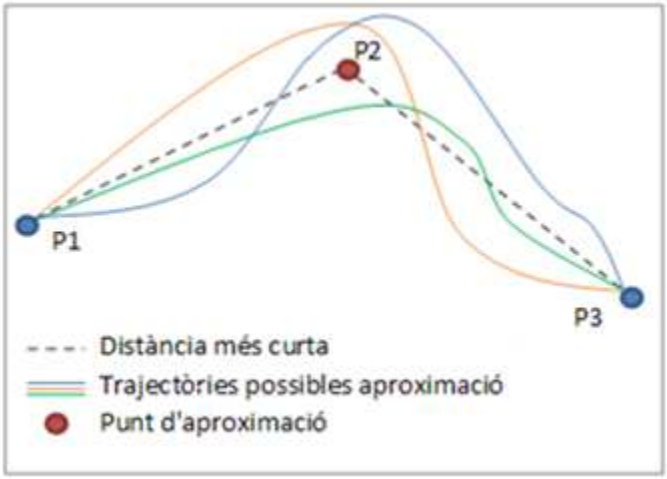


Fig. 2.62 – Moviment d'aproximació en PTP

El valor de la distància d'aproximació especifica el tamany del rang de col·locació aproximada. La trajectòria no pot ser fixada, ni és fiable.

Aquest tipus d'aproximacions es fan quan el robot es mou en l'espai lluny de possibles objectes o elements amb els que podria col·lisionar.

- A continuació coneixerem els casos d'aproximació per LIN i CIRC:

En les aproximacions LIN sabem que la aproximació tindrà lloc per la part interna de les dues rectes de distància mínima. Aquesta aproximació es dóna a partir de dues corbes parabòliques (simètriques a velocitat constant).

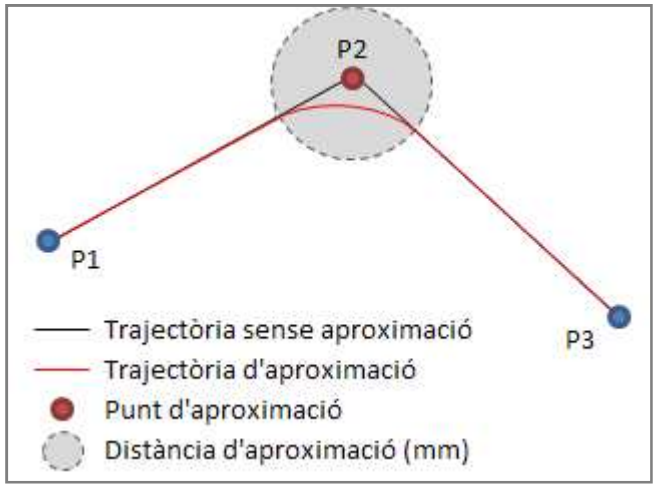


Fig. 2.63 - Moviment d'aproximació en LIN

El valor introduït com a "distància d'aproximació" especifica la distància del punt final i del punt on comença el moviment d'aproximació. La trajectòria que resulta no és un arc, donat que fora de la distància d'aproximació efectua el moviment lineal normal.

En les aproximacions CIRC sabem que la aproximació tindrà lloc per la part interna de les dues trajectòries. Aquesta aproximació es dóna a partir de dues corbes parabòliques (simètriques a velocitat constant) com en el cas del moviment lineal. Podem aproximar dues corbes normalment sense cap dificultat extra.

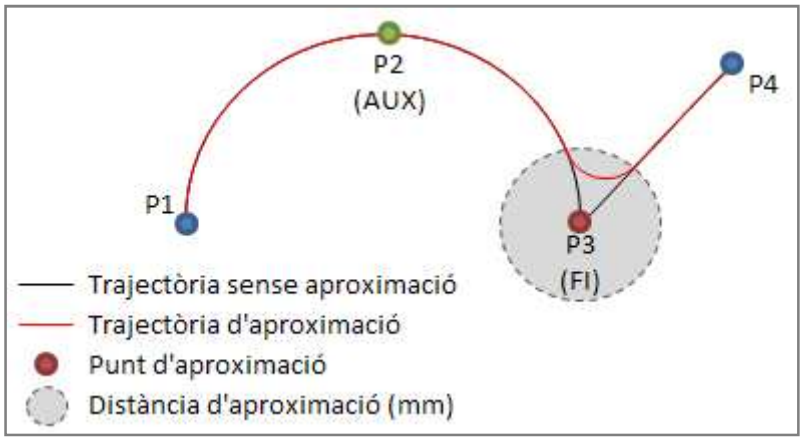


Fig. 2.64 - Moviment d'aproximació en CIRC

Exactament igual que en els moviments lineals, el valor introduït com a "distància d'aproximació" especifica la distància del punt final i del punt on comença el moviment d'aproximació.

Però el tema de les aproximacions es complica una mica degut al punter Advance (que sempre hem de tenir present). A vegades l'execució del programa és totalment diferent degut al advance i a si s'ha tingut en consideració o no. En el següent punt parlarem de la lògica. Però aquesta lògica també té a veure el Advance i els moviments.

2.7.2 Lògica

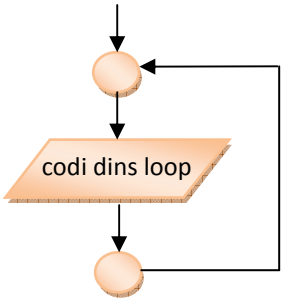
En qualsevol programació d'alt nivell no només es fan servir moviments de la eina d'un punt a un altre sinó que sovint cal prendre decisions, repetir parts del codi, establir condicions, bucles, etc. Per a dur a terme tota aquesta lògica de programa disposem de les següents instruccions:

Espera de temps: Si emprem la funció WAIT cal indicar el temps d'espera. Aquesta comanda sempre atura el punter d'Advance, fins i tot per un temps d'espera de 0segons.

```
WAIT time=1 sec        → Espera durant 1 segon
```

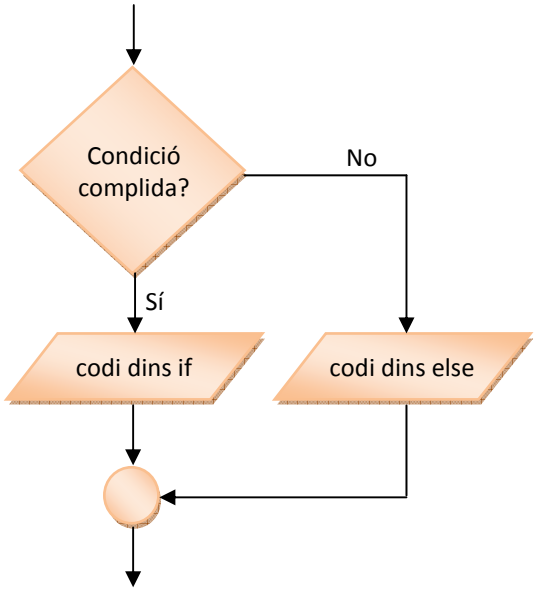
Bucle cíclic: Les execucions cícliques poden programar-se dins d'un LOOP. El bloc dins de la instrucció LOOP es repetirà contínuament. Si es desitja acabar amb l'execució repetida del bloc es pot trucar a la funció EXIT.

```
LOOP
...                    → línies de codi a repetir en el bucle sense fi
ENDLOOP
```



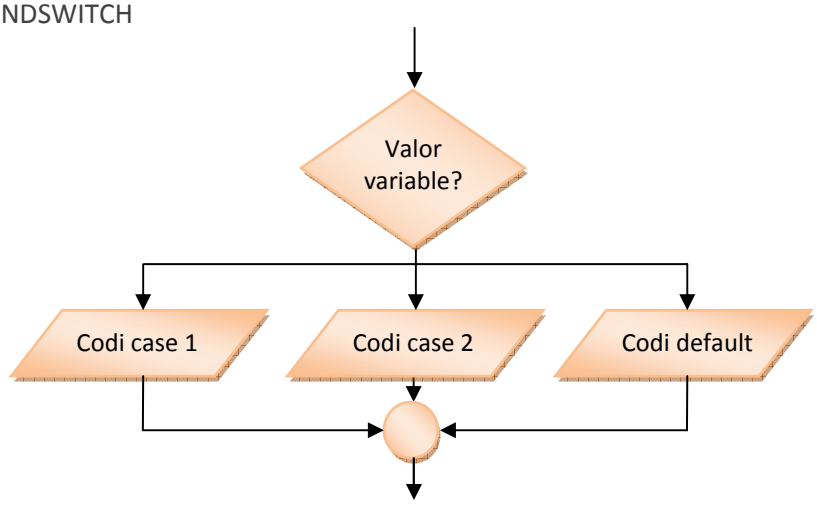
Condició: No hi ha límit en el número de declaracions contingudes dins dels blocs de la declaració. Diverses instruccions IF es poden introduir una dins l'altre. La paraula ELSE i el segon bloc de declaració poden ometre's.

```
IF condició THEN
...          → línies de codi a processar si es compleix la condició
ELSE
...          → línies de codi a processar si no es compleix la condició
ENDIF
```



Branca condicional, casos: La instrucció SWITCH selecciona entre diverses branques del programa. S'executa només aquella branca del programa que esdevé el cas del valor de la variable i un cop executada la branca del CASE corresponent, es salta directament a la línia de ENDSWITCH.

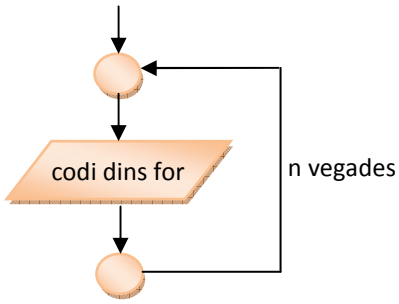
```
SWITCH variable
CASE 1       → línies de codi a processar si variable val 1
...
CASE 2
...          → línies de codi a processar si variable val 2
DEFAULT
...          → línies de codi a processar si variable no val cap dels casos
               anteriors.
ENDSWITCH
```



Repetició numerada del codi: La instrucció FOR ens permet repetir un numero de n de vegades les instruccions contingudes.

```
FOR n = 0 TO 20
...
ENDFOR
```

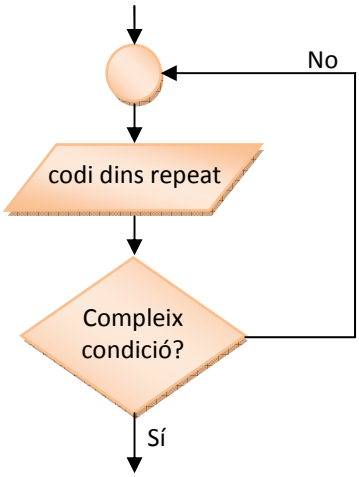
→ línies de codi a repetir "n" vegades (en aquest cas 20)



Repetició fins que es compleix condició: La instrucció REPEAT ens permet repetir un numero de n de vegades les instruccions contingudes.

```
REPEAT
...
UNTIL condició
```

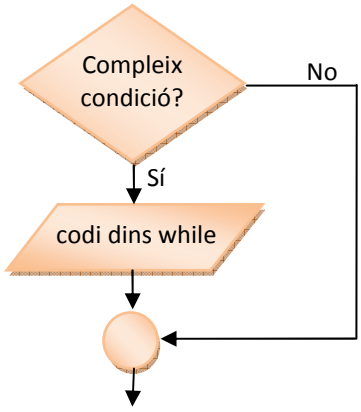
→ línies de codi a executar fins que es compleixi condició



Mentre condició es compleix, repeteix: La instrucció WHILE ens permet repetir un numero de n de vegades les instruccions contingudes.

```
WHILE condició
...
ENDWHILE
```

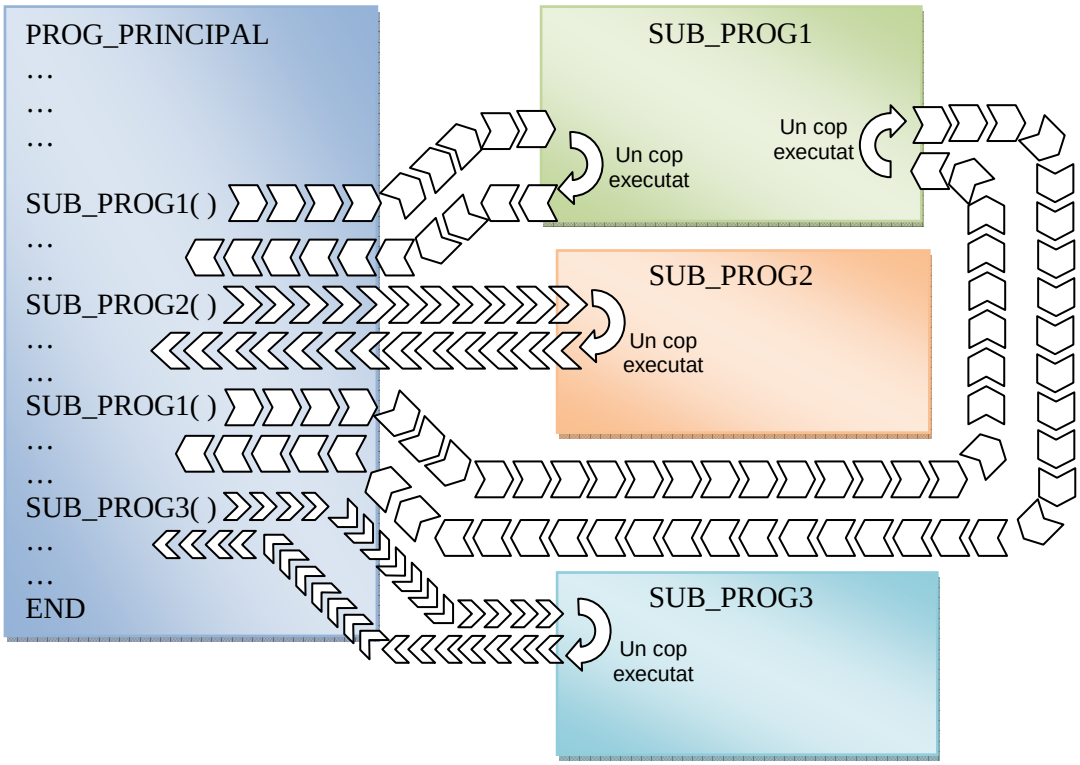
→ línies de codi a executar mentre es compleixi condició



Subprogrames – Rutines: Es defineixen com un nou programa, i es criden com si fora una funció en altres llenguatges de programació. Es pot executar una mateixa subrutina diverses vegades en un mateix programa. A continuació un exemple.

```
DEF PROG_PRINCIPAL ( )
INI
PTP HOME
...                  → Línies de codi del programa principal
...
SUB_PROG1( ) → Crida del subprograma (l'executa i torna a la línia següent)
...
...                  → Més línies de codi del programa principal
PTP HOME
END
```

```
DEF SUB_PROG1 ( )
INI
...          → Cos del subprograma
END
```



2.7.3 Entrades i sortides

El número d'entrades i sortides es configura per a la comunicació entre el control del robot i la perifèria (com ara eines, sensors, etc).

Les funcions d'espera dependents del temps i de senyals, a més a més de la funció de commutació i pols, poden provocar l'aturada del procés.

Senyal dependent de la funció d'espera, WAIT FOR: Els paràmetres de la plantilla inline especifiquen el següent:

	Denom. del camp	Funció	Rang de valors
A	WAIT FOR IN	Tipus de funció de espera	WAIT FOR IN, WAIT FOR OUT
B	1	Entrada / sortida	1 ... 1024
C	Demo	Text llarg	El text llarg, que s'ha assignat a una entrada o sortida
D	State	Estat	TRUE, FALSE
E	CONT	Pos. de aproximació	" ", CONT

A → Aquí s'indica si cal esperar a la senyal de una entrada o de una sortida.

B → Número de entrada/sortida o bé la condició:

La condició d'espera es pot programar, per exemple, de la forma següent:

Operació lògica interna: L'operador està situat dins de una expressió entre parèntesis.

Operació lògica externa: L'operador està situat entre expressions amb parèntesis.

Són possibles les operacions mixtes: Un màxim de 12 operands es poden incloure dins d'una mateixa forma.

C → Text opcional que es post escriure per a donar informació addicional. No afecta en la programació de la instrucció.

D → Amb aquest camp pot indicar l'estat que, al presentar-se, continuï amb el programa de moviments/instruccions.

E → Aquest darrer camp d'entrada té la possibilitat d'activar la opció "CONT" (Continue). Aquesta opció ocasiona, que la condició és controlada en el moment que el punter de processament en advance arribi a la línia "WaitFor". Si la condició en aquell moment es

compleix, el punter d'Advance salta a la propera línia. En cas contrari queda detingut allà fins que la condició s'hagi complert. Veiem alguns exemples:

1] Si la funció "WAIT FOR" es selecciona per activar l'aturada del Advance, sempre s'arriba a la posició exacta, fins i tot quan es compleix la condició:

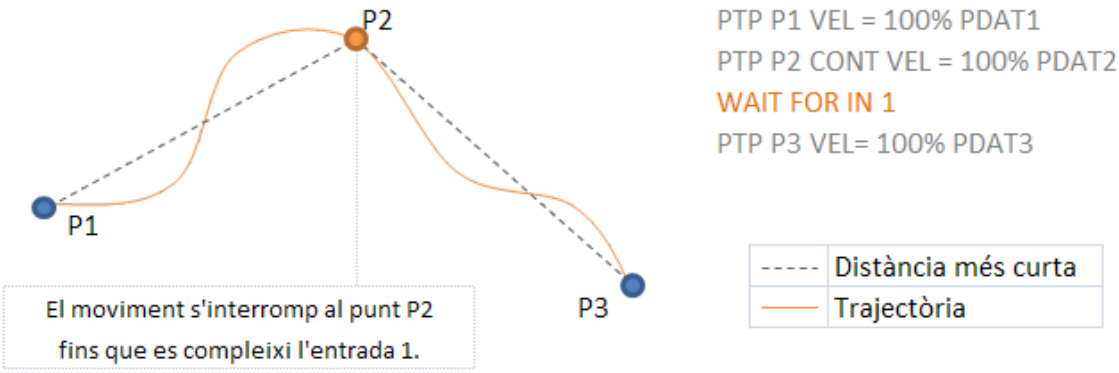
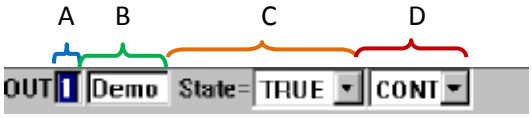


Fig. 2.65 – Exemple 1 "WAIT FOR"

2] Si se selecciona "WAIT FOR" amb CONT, l'event es comprova durant el moviment sense detenir l'Advance. Si es compleix la condició de l'event, llavors es donarà lloc a una aproximació de la posició.

Fig. 2.66 – Exemple 2 "WAIT FOR"

Funció de commutació simple (OUT): Posa una sortida en True o False. La plantilla inline especifica el següent:



	Denom. del camp	Funció	Rang de valors
A	1	Sortida	1 ... 1024
B	Demo	Text llarg	Denominació de la sortida
C	State	Estat	TRUE, FALSE
D	CONT	Pos. de aproximació	" ", CONT

A → Aquí s'introdueix el número de la sortida que ha de ser activada/desactivada.

B → En aquest camp es pot modificar el text llarg corresponent a la sortida. (Aquest pas s'efectua des del grup d'usuari expert, si es tracta d'un operador no pot modificar aquest paràmetre que no afecta a la programació).

C → Aquí indiquem a quin estat volem posar la sortida True/False.

D → Aquest darrer camp d'entrada té la possibilitat d'activar la opció "CONT" (Continue). Aquesta opció ocasiona, que la condició és controlada en el moment que el punter de processament en advance arribi a la línia "WaitFor". Si la condició en aquell moment es compleix, el punter d'Advance salta a la propera línia. En cas contrari queda detingut allà fins que la condició s'hagi complert. Veiem alguns exemples:

1] Si la funció "OUT" No inclou el CONT:

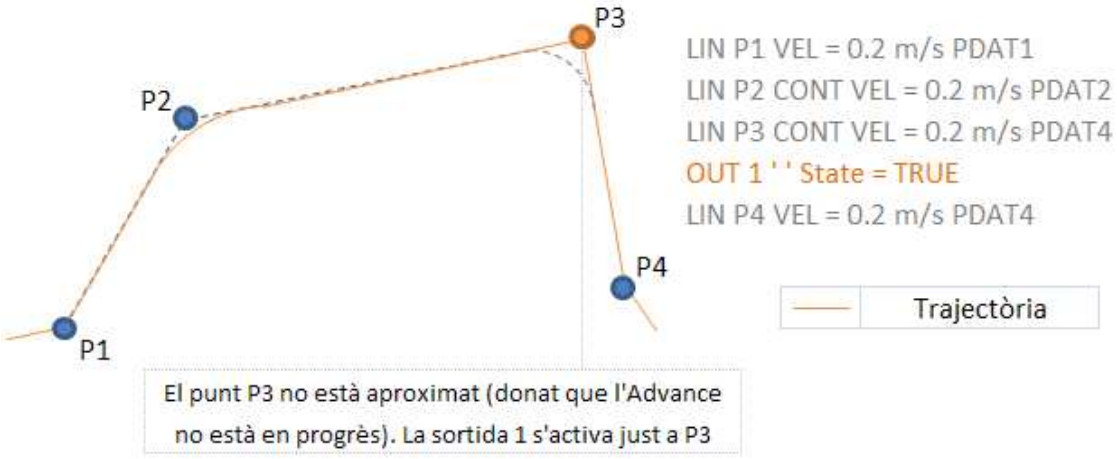


Fig. 2.67 – Exemple funció "OUT" sense CONT.

2] Si la funció "OUT" inclou el CONT:

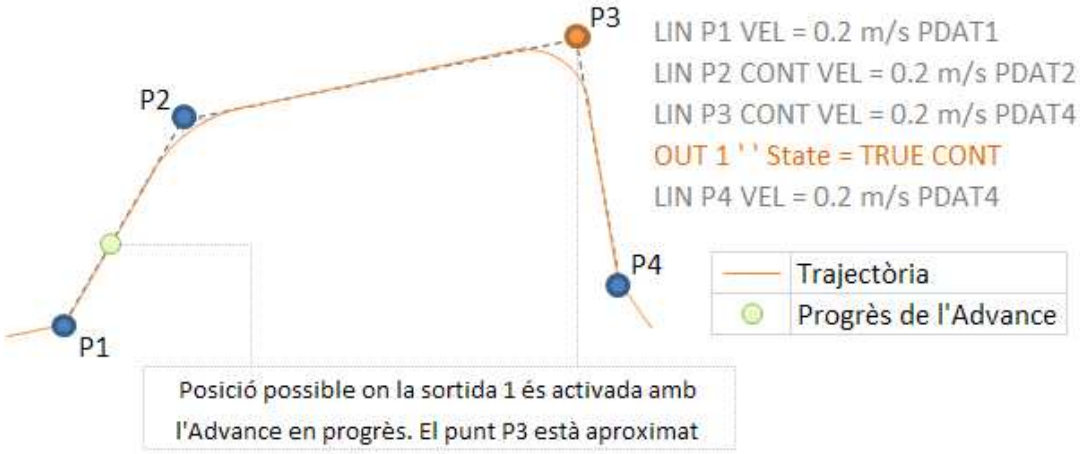
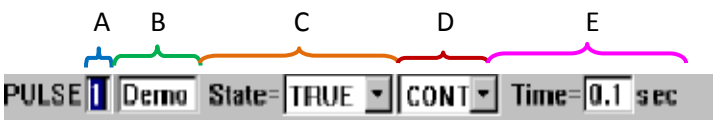


Fig. 2.68 – Exemple funció "OUT" amb CONT.

Funció d'impuls simple (PULSE): Posa una sortida en True o False durant un temps d'impuls determinat. La plantilla inline especifica el següent:



	Denom. del camp	Funció	Rang de valors
A	1	Sortida	1 ... 1024
B	Demo	Text llarg	Denominació de la sortida
C	State	Estat	TRUE, FALSE
D	CONT	Pos. de aproximació	" ", CONT
E	Time	Longitud de l'impuls	0.1 ... 3 seg

A → Aquí s'introdueix el número de la sortida que ha de ser activada/desactivada.

B → En aquest camp es pot modificar el text llarg corresponent a la sortida. (Aquest pas s'efectua des del grup d'usuari expert, si es tracta d'un operador no pot modificar aquest paràmetre que no afecta a la programació).

C → Aquí indiquem a quin estat volem posar la sortida True/False.

D → Aquest darrer camp d'entrada té la possibilitat d'activar la opció "CONT" (Continue). Aquesta opció ocasiona, que la condició és controlada en el moment que el punter de processament en advance arribi a la línia "WaitFor". Si la condició en aquell moment es compleix, el punter d'Advance salta a la propera línia. En cas contrari queda detingut allà fins que la condició s'hagi complert.

E → Aquest camp determina la longitud del impuls entre 0,1 i 3segons. El valor de cada pas és, en aquest cas, 0,1segons.

Funció de commutació dependent de la trajectòria (SYN OUT): S'encarrega d'executar les funcions de commutació temps-distància dependents de la trajectòria.

	Denom. del camp	Funció	Rang de valors
A	1	Sortida	1 ... 1024
B	Demo	Text llarg	Denominació de la sortida
C	State	Estat	TRUE, FALSE
D	at	Instant en que s'executa la funció de commutació	START, END, PATH
	a disposició, quan s'utiliza la opció "PATH"	Distància al punt de destinació, en que s'ha d'activar la sortida*	-2000 ... 2000 mm
E	Delay	Retard de la acció de commutació	-1000 ... 1000 ms

A → Aquí s'introdueix el número de la sortida que ha de ser activada/desactivada.

B → En aquest camp es pot modificar el text llarg corresponent a la sortida. (Aquest pas s'efectua des del grup d'usuari expert, si es tracta d'un operador no pot modificar aquest paràmetre que no afecta a la programació).

C → Aquí indiquem a quin estat volem posar la sortida True/False.

D → Definició del segment de la trajectòria on cal activar/desactivar la sortida seleccionada anteriorment. START (inici) o END (final). Si per el contrari es desitja determinar una sortida en relació a la trajectòria, cal seleccionar PATH (trajectòria).

E → Aquí es pot determinar un retard en temps (ms) respecte el moment en que s'activa/desactiva la sortida. Aquesta és una instrucció una mica més complexa que les anteriors, anem a veure amb uns quants casos per entendre com funciona. Exemples:

1] El punt d'inici i el punt final són punts executats sense aproximació:

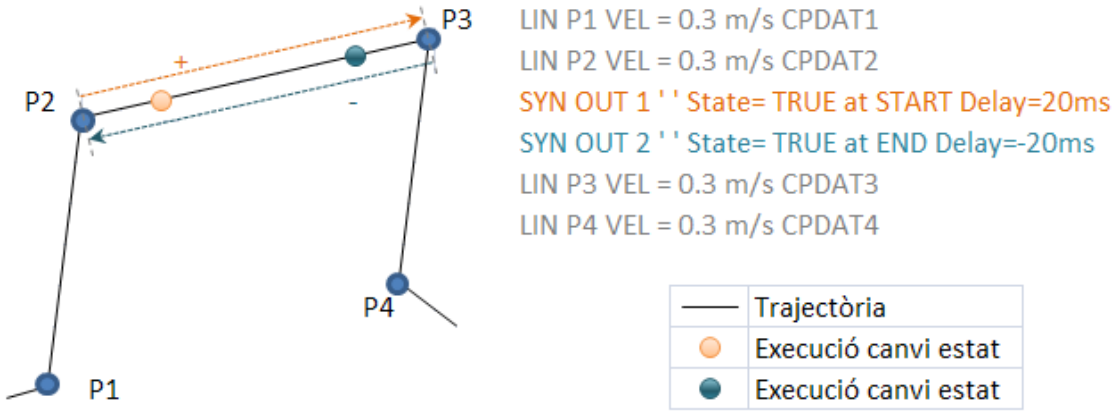


Fig. 2.69 – Exemple 1 funció "SYN OUT".

Entre el P2 i el P3 s'estableix l'espai del límit de canvi. On P2 és el punt d'inici i P3 és el punt final. Si els valors especificats estan fora dels marges, el control automàticament els introdueix dins el límit de canvi.

2] El punt d'inici és sense aproximació i el punt final és aproximat:

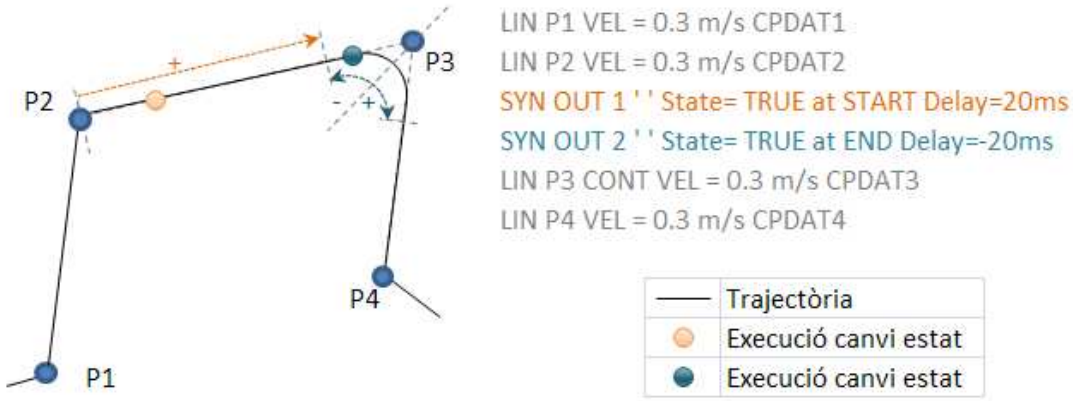


Fig. 2.70 – Exemple 2 funció "SYN OUT".

El punt d'inici és el punt P2, el punt final és el punt P3. El Rang d'aproximació quan hi ha aproximació és considera des del centre de l'arc de l'aproximació. La entrada 2 es posarà a cert a l'esquerra del centre de l'aproximació del punt final (P3).

3] El punt inicial i final són aproximats:

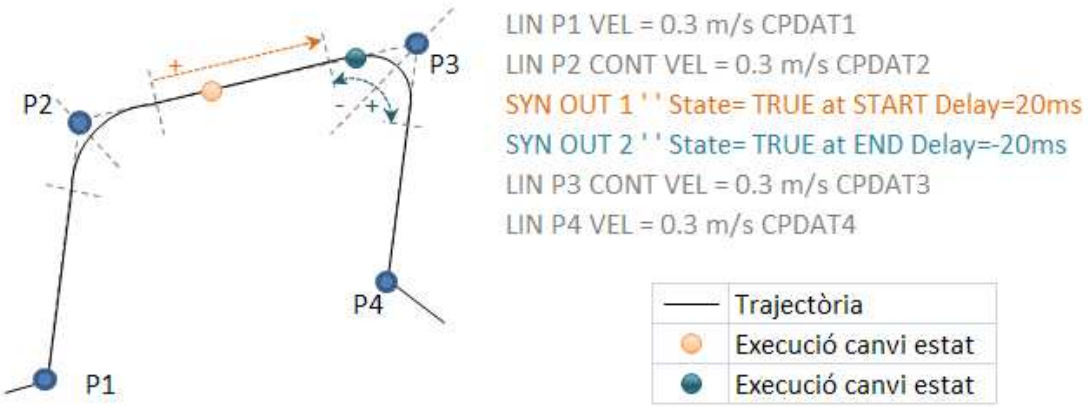


Fig. 2.71 – Exemple 3 funció "SYN OUT".

El punt d'inici és el punt P2, el punt final és el punt P3. El Rang d'aproximació quan hi ha aproximació és considera des del centre de l'arc de l'aproximació. Ara tots dos punts són aproximats. La entrada 1 es posarà a cert quan hagi acabat la aproximació i estigui en trajectòria i tindrà de límit de canvi fins a arribar a la aproximació del punt següent. La entrada 2 es posarà a cert a l'esquerra del centre de l'aproximació del punt final (P3).

4] Selecció en qualsevol punt de la trajectòria PATH:

Si a la declaració de SYNOUT-PATH amb la especificació de trajectòria es programa amb moviments PTP, serà rebutjat per l'interpretador quan el moviment sigui executat. La declaració SYNOUT-PATH es refereix al proper bloc de moviment programat.

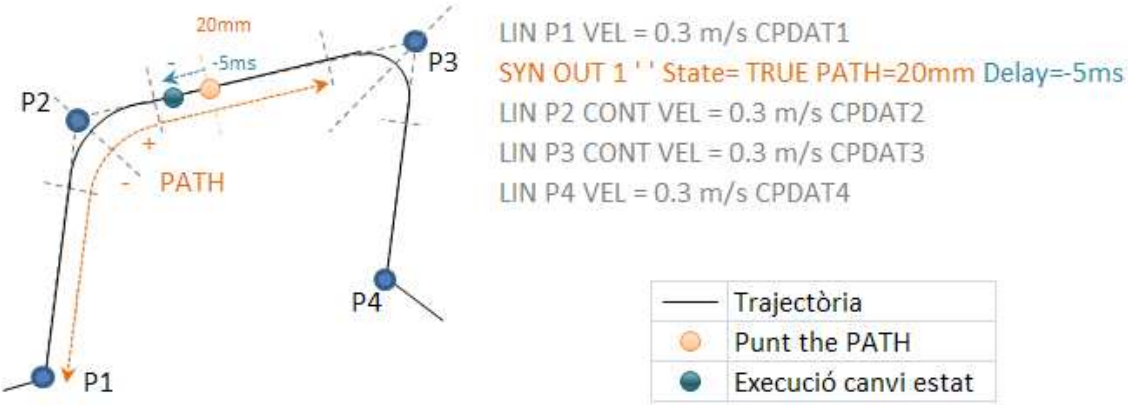
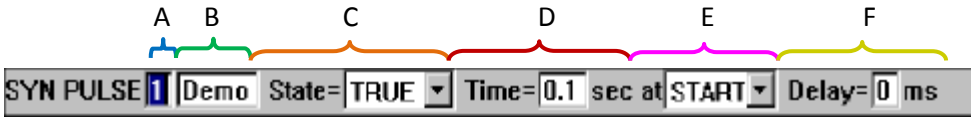


Fig. 2.72 –Exemple 4 funció "SYNOUT".

El PATH segueix tota la trajectòria del robot que estigui traçada per moviments LIN o CIRC i compta els mm introduïts a partir dels quals s'executarà la sortida. El punt en que s'avalua la distància és l'immediat i s'hi pot afegir un retard (o avançament) en temps des d'aquest punt per a l'execució del canvi d'estat de la sortida.

En aquest cas no importa si els moviments són aproximats o no, importa exclusivament la trajectòria que s'executa.

Funció de impuls dependent de la trajectòria (SYN PULSE): S'encarrega d'executar les funcions d'impuls dependents de la trajectòria.



	Denom. del camp	Funció	Rang de valors
A	SYN PULSE	Sortida	1 ... 1024
B	Demo	Text llarg	Denominació de la sortida
C	State	Estat	TRUE, FALSE
D	Time	Longitud de l'impuls	0.1 ... 3 seg
E	at	Instant en que s'executa la funció de commutació	START, END, PATH
	a disposició, quan s'utiliza la opció "PATH"	Distància al punt de destinació, en que s'ha d'activar la sortida*	-2000 ... 2000 mm
F	Delay	Retard de la acció de commutació	-1000 ... 1000 ms

A → Aquí s'introdueix el número de la sortida que ha de ser activada/desactivada.

B → En aquest camp es pot modificar el text llarg corresponent a la sortida. (Aquest pas s'efectua des del grup d'usuari expert, si es tracta d'un operador no pot modificar aquest paràmetre que no afecta a la programació).

C → Aquí indiquem a quin estat volem posar la sortida True/False.

D → Aquest camp determina la longitud del impuls entre 0,1 i 3segons. El valor de cada pas és, en aquest cas, 0,1segons.

E → Definició del segment de la trajectòria on cal activar/desactivar la sortida seleccionada anteriorment. START (inici) o END (final). Si per el contrari es desitja determinar una sortida en relació a la trajectòria, cal seleccionar PATH (trajectòria).

F → Aquí es pot determinar un retard en temps (ms) respecte el moment en que s'activa/desactiva la sortida.

Sortida analògica (ANOUT): A través d'aquesta funció, es determinen sota control de programa, les sortides analògiques de la unitat de control del robot. Hi ha dues formes de programar-ho (de forma estàtica o dinàmica).

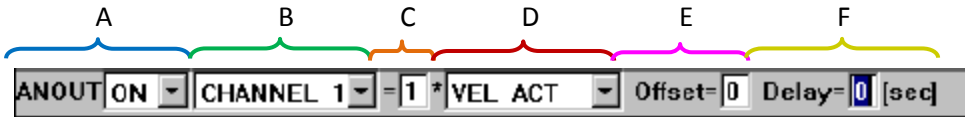
ESTÀTICA: Amb aquesta opció s'assigna a una sortida analògica un valor fix.

	Denom. del camp	Funció	Rang de valors
A	ANOUT	Sortida analógica	1 ... 32
B	0	Tensió de sortida	0 ... 1

A → Indicar quina sortida analògica es vol fixar.

B → Aquí pot modificar-se el valor introduït o per defecte. El valor de cada pas, és en aquest cas, de 10mV.

DINÀMICA: Amb aquesta opció s'activa una sortida analògica depenent de la velocitat i la tecnologia.



	Denom. del camp	Funció	Rang de valors
A	ANOUT	Connectar o desconnectar la sortida analógica	ON, OFF
B	CHANNEL 1	Sortida analógica	1 ...32
C	1	Multiplicador	0 ... 10
D	VEL ACT	Paràmetre de velocitat o de tecnologia	VEL ACT, TECHVAL1 ... TECH-VAL6
E	Offset	Tensió offset	-1 ... 1
F	Delay	Retard	-0.2 ... 0.5 seg

A → En aquest camp determinem si desitgem activar o desactivar la sortida analògica.

B → Indicar quina sortida analògica es vol activar o resetejar.

C → Factor de multiplicació amb que han de ser multiplicats els paràmetres de velocitat o de tecnologia. El valor de cada pas, és en aquest cas, de 0,05.

D → Aquí podem indicar amb quins paràmetres de velocitat o tecnologia han d'estar connectats lògicament les sortides analògiques seleccionades

E → Valor de tensió d'offset per a la sortida analògica seleccionada. El valor de cada pas aquí és de 100mV.

F → Aplicar un valor de retard. El valor de cada pas és aquí de 1/100 segons.

Comentaris (;): El llenguatge de programació et permet introduir comentaris allà on es consideri oportú (sempre i quan no sigui dins d'una instrucció) de manera que es faci més entenedor el programa o puguis prendre's anotacions.

Per a incloure un comentari només cal posar ; (punt i coma) davant del text a comentar. També tenim un formulari inline per a afegir comentaris.

Exemple:

```
PTP HOME  Vel= 100 % DEFAULT
; texto cualquiera
PTP HOME  Vel= 100 % DEFAULT
```

2.8 MODE AUTOMÀTIC EXTERN

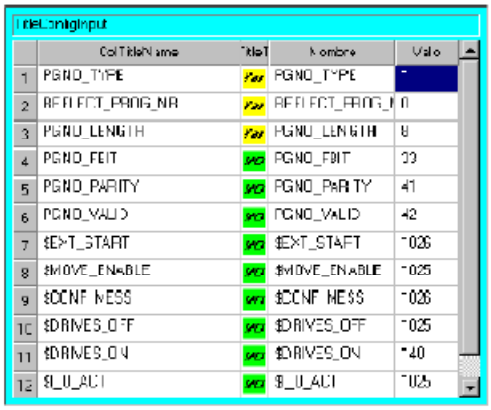
En línies de producció concatenades, és necessari poder iniciar els processos del robot des d'una sala o zona central.

A través del mode "Automàtic Extern", un ordinador central pot comunicar amb el control del robot i activar diferents processos. De la mateixa manera, la unitat de control del robot pot transmetre informació sobre els estats de servei i avisos d'anomalia al ordinador central.

En el KR C2 es realitza tot mitjançant la posta en marxa automàtica de l'equip, el programa d'organització específica de tecnologia CELL.SRC i a través de les funcions del mòdul P00.

A les senyals de la interfície "Automàtic Extern" se li han d'assignar entrades i sortides físiques de control del robot. Cal anar a "CONFIGURAR" la opció "Entradas/Salidas".

Entrades:



ColTitleName: Una descripció de la funció de la corrsponent variable o entrada.

Var Ein : Tipus variable (groc) o entrada (verd).

Nombre: El nom de la variable de la entrada corresponent.

Valor: El valor de la entrada o del número de canal (camp editable).

Sortides :

ColTitleName: Una descripció de la funció de la corrsponent variable o sortida.

Var Ein : Tipus variable (groc) o sortida (verd).

Nombre: El nom de la variable de la entrada corresponent.

Valor: El valor de la entrada o del número de canal (camp editable).

Si la interfície de E/S es commuta a activa mitjançant les variables del sistema $I_O_ACT amb valor TRUE, la sortida $_I_O_ACTCONF com a senyal de retorn es posa també a TRUE. Si s'han complert tota la resta de condicions de posta en marxa, pot inciar-se el programa CELL.SRC a través d'una senyal en línia $EXT_START. Tot això faria posar en marxa el sistema en la instal·lació.

Evidentment es pot iniciar el programa CELL.SRC en tot moment des de la superfície d'operació.

2.8.1 Programa d'organització específic de tecnologia CELL.SRC

Estructura del programa CELL.SRC amb comentaris:

Instrucció per a vincular els subprogrames externs definits per l'usuari:

```
;EXT EXAMPLE1 ( )
;EXT EXAMPLE2 ( )
;EXT EXAMPLE3 ( )
```

Seqüència de inicialització:

```
INIT
BAS INI
CHECK HOME
PTP HOME Vel=100% DEFAULT
AUTOEXT INI
```

Inici del bucle:

```
LOOP
```

Trucada del mòdul P00, per a poder trucar al número de programa des de l'ordinador central extern.

```
P00 (#EXT_PGNO, #PGNO_GET, DMY[ ] ,0 )
```

Estructura de control dependent del número de programa rebut:

```
SWITCH PGNO
```

Quan el número de programa PGNO = 1 comunicar-li al ordinador central la recepció del número de programa i crida del programa definit per l'usuari EXAMPLE1:

```
CASE 1
P00 (#EXT_PGNO,  #PGNO_ACKN, DMY[ ] ,0 )
;EXAMPLE1 ( )
```

Quan el número de programa PGNO = 2 comunicar-li al ordinador central la recepció del número de programa i crida del programa definit per l'usuari EXAMPLE2:

```
CASE 2
P00 (#EXT_PGNO,  #PGNO_ACKN, DMY[ ] ,0 )
;EXAMPLE2 ( )
```

Quan el número de programa PGNO = 3 comunicar-li al ordinador central la recepció del número de programa i crida del programa definit per l'usuari EXAMPLE3:

```
CASE 3
P00 (#EXT_PGNO,  #PGNO_ACKN, DMY[ ] ,0 )
;EXAMPLE3 ( )
```

Si per el número de programa transmès per l'ordinador central no se hi localitza cap ramificació CASE, produïm un tractament de l'error:

```
DEFAULT
P00 (#EXT_PGNO, #PGNO_FAULT, DMY[ ] ,0 )
```

Final de la estructura de control:

```
ENDSWITCH
```

Final del bucle:

```
ENDLOOP
```

Final del programa:

```
END
```

2.8.2 Entrades (automàtic extern)

PGNO_TYPE: No es tracta ni d'una entrada ni d'una senyal, sinó que és una variable. Amb el seu valor es pot prefixar el format amb el que es desitja que l'ordinador agafi la lectura del número de programa transmès.

PGNO_TYPE	Lectura com	Significat	Exemple
1	Binari	El número de programa és transferit del control superior com valor enter codificat en binari.	0 0 1 0 0 1 1 1 PGNO = 39
2	Valor BCN	El número de programa és transferit de control superior com valor decimal codificat en binari.	0 0 1 0 0 1 1 1 PGNO = 2 i 7 = 27
3	"1 de n"	El número de programa és transferit del control superior o de la perifèria com valor codificat "1 de n".	0 0 0 0 0 0 0 1 PGNO = 1 0 0 0 0 1 0 0 0 PGNO = 4

PGNO_LENGTH: Tampoc es tracta d'una entrada o senyal, de nou és una variable. Amb el seu valor es pot prefixar l'amplada del bloc de bits del número de programa transmès per l'ordinador principal.

PGNO_LENGTH = 1 ... 16

Exemple:

PGNO_LENGTH = 6 →El número de programa extern té un ample de 6bits.

Mentre que el PGNO_TYPE disposi de valor 2 (Lectura del número de programa com valor BCD), només es permetran amples de grups de bits de 4, 8, 12 i 16.

PGNO_FBIT: Entrada que representa el primer bit del número de programa.

PGNO_FBIT = 1 ... 1024

Exemple:

PGNO_FBIT = 5 →El número de programa extern comença amb $IN[5]

REFLECT_PROG_NR: Amb aquesta opció es defineix si el número de programa ha de reflexar-se (funció mirall) sobre la zona de sortida a definir. El valor de la variable pot ser modificat a través de la configuració de la interfície d'Automàtic Extern.

REFLECT_PROF_NR	Funció
0	Desactivada
1	activada

La activació s'efectua a partir de la sortida definida per mitjà de "PGNO_FBIT_REFL".

PGNO_PARITY: Entrada sobre la que es transmet el bit de paritat de l'ordinador principal.

PGNO_PARITY	Funció
Valor negatiu	Paritat senar
0	Sense avaluació
Valor positiu	Paritat parell

Mentre PGNO_TYPE disposi de valor 3 (lectura del número de programa com a valor "1 de n"), NO s'avaluarà PGNO_PARITY.

PGNO_VALID: Entrada sobre la qual es transmet la comanda de lectura del número de programa per l'ordinador principal.

PGNO_VALID	Funció
Valor negatiu	S'adopta el número amb flanc decreixent de la senyal
0	S'adopta el número amb flanc creixent de la senyal en la línia EXT_START
Valor positiu	S'adopta el número amb flanc creixent de la senyal

Mentre PGNO_TYPE disposi de valor 3 (lectura del número de programa com a valor "1 de n"), NO s'avaluarà PGNO_VALID.

EXT_START: Al col·locar aquesta senyal es pot iniciar, o bé continuar, un programa amb interfície de E/S activa.

Només s'avalua el flanc creixent de la senyal.

En servei automàtic extern no es produeix cap marxa COI i, per tant, no existeix cap aturada de programa a la primera posició programada. Això regeix tant per aturada amb efecte generador com abandonament de trajectòria (per exemple, degut a protecció d'operador), com al sortir de la trajectòria en forma manual.

La primera posició a desplaçar-se, en aquests casos, la posició memoritzada en $POS_RET abans de la interrupció. Per consegüent, abans de que l'operador posi el EXT_START haurà de garantir que el robot sigui realment a la posició o bé, pugui abastar-la sense perill.

El primer pas de moviment haurà de ser un PTP amb indicació absoluta del punt a destinació. Aquest és desplaçarà amb exactitud i a plena velocitat ignorant una instrucció programada de posicionament o aproximació.

MOVE_ENABLE: Aquesta entrada s'utilitza per el control dels accionaments del robot a través de l'ordinador principal.

MOVE_ENABLE	Funció
TRUE	Possibilitat de desplaçament manual i execució de programa.
FALSE	Aturada de tots els accionaments i bloqueig de totes les comandes actives.

Si els accionaments han estat aturats per l'ordinador principal, apareix a la finestra de missatges del KCP el missatge "LIBERACIÓB DE MOVIMIENTO GENERAL". El moviment del robot es podrà realitzar només en esborrar aquest missatge i després d¡activar una senyal de posta en marxa externa.

Durant el servei es configura, sovint la variable per la alliberació de moviment "$MOVE_ENABLE", al valor de "$IN[1025]". Si després s'oblida configurar una altra entrada, no es possible obtenir cap posta en marxa externa.

CHCK_MOVENA: Si la variable $CHCK_MOVENA té el valor "FALSE", pot obviar-se MOVE_ENABLE. Els valors de les variables només poden modificar-se al fitxer "C:\KRC\Roboter\KRC\Steu\MaDa\OPTION.DAT".

CHCK_MOVENA	Funció
TRUE	Control per MOV_ENABLE está activat.
FALSE	Control per MOVE_ENABLE está desactivat.

Per a poder emprar el control per MOVE_ENABLE, $MOVE_ENABLE ha d'estar configurada sobre la entrada "$IN[1025]". En cas contrari "$CHCK_MOVENA" no té cap efecte.

CONF_MESS: Activant aquesta senyal, l'ordinador principal pot autoesborrar els avisos de falles originats (confirmació). Naturalment només pot confirmar avisos de falla una vegada solucionada la causa que la va produir.

Només s'avalua el flanc creixent de la senyal.

DRIVES_ON: Mitjançant un impuls de nivell alt durant 20ms, com a mínim, en aquesta entrada, l'ordinador principal pot activar els accionaments del robot.
DRIVES_OFF: Mitjançant un impuls de nivell baix durant 20ms, com a mínim, en aquesta entrada, l'ordinador principal pot desactivar els accionaments del robot.

2.8.3 Sortides (automàtic extern)

STOPMESS: Aquesta sortida la posa el control del robot per indicar al ordinador principal que s'ha produït un missatge que requereix l'aturada del robot.

Per exemple en cas d'aturada d'emergència, alliberament de la marxa, protecció de l'operador, velocitat nominal de comandament, etc.

PGNO_REQ: Amb un canvi de senyal en aquesta sortida es demana al ordinador principal que transmeti un número de programa.

S'avaluen ambdós flancs de la senyal

Mentre PGNO_TYPE disposi de valor 3 (lectura del número de programa com a valor "1 de n"), NO s'avaluarà PGNO_REQ.

PGNO_FBIT_REFL: Sortida reflexada (mirall), que representa el primer bit del número de programa. Per la utilització d'aquesta opció ha d'assignar-se a la variable "REFLECT_PROG_NR" el valor "1".

Si un programa, que ha estat seleccionat pel PLC es desselecciona per l'operari, llavors la secció de sortida, començant amb PGNO_FBIT_REFL es posa en "FALSE". D'aquesta manera, el PLC pot evitar un nova posta en marxa manual del programa. També es posa a "FALSE" quan l'interpretador es troba al programa CELL.SRC

PGNO_FBIT_REFL = 1 ... 1024 (PGNO_LENGTH)

El tamany de la secció de sortida depèn de l'amplada de bits del número de programa (PGNO_LENGTH).

Exemple:

PGNO_FBIT_REFL = 5 →El número de programa comença amb $OUT[5].

APPL_RUN: Amb la activació d'aquesta sortida , el control del robot comunica al ordinador principal que es està processant en aquell precís moment un programa.

El valor de APPL_RUN no ha de ser menor a "0".

PERI_RDY: Amb la activació d'aquesta sortida el control del robot comunica al ordinador principal que els accionaments del robot estan actius.

ALARM_STOP: Aquesta sortida es reseteja quan es produeix un event d'aturada d'emergència.

USER_SAF: Aquesta sortida es reseteja en obrir un contacte del commutador de control del tancat de seguretat (en el mode de servei AUTO), o bé al deixar anar un pulsador d'home mort (en el mode de servei TEST).

ON_PATH: Aquesta sortida estarà activa mentre el robot es troba dins de la seva trajectòria programada.

Després d'un desplaçament COI, s'activa la sortida ON_PATH. Aquesta sortida romandrà activa fins que el robot surti de la seva trajectòria, es resetegi el programa o bé s'executi una selecció de pas. La senyal ON_PATH no disposa d'una finestra de tolerància, en el moment que el robot abandoni la trajectòria, la senyal es reseteja.

NEAR_POSRET: A través de una segona senyal, NEAR_POSRET, el ordinador principal pot determinar si el robot està dins d'un entorn esfèric a la posició memoritzada $POS_RET. El radi de la esfera pot ajustar-se per l'usuari en el fitxer $CUSTOM.DAT a través de la variable del sistema $NEARPATHTOL.

Amb aquesta informació, el ordinador principal pot decidir si el programa s'ha d'arrencar o no. La posició de retorn $POS_RET és la posició en la qual el robot ha abandonat la trajectòria.

Al canviar al mode de servei "Automàtic Extern" es controla si la variable "$NEAR_POSRET" està a "TRUE". En cas contrari, s'emet el corresponent missatge a la finestra de missatges.

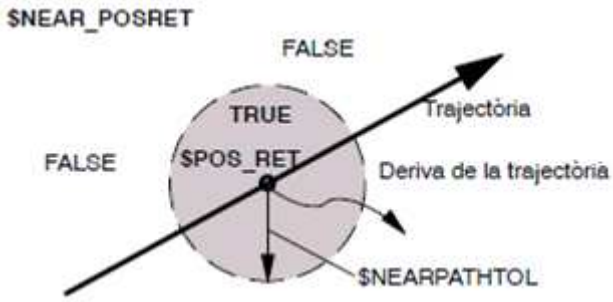


Fig. 2.73- $NEARPATHTOL

Condicions per NEAR_POSRET:

TRUE:

ON_PATH està actiu, o bé, si ON_PATH no està actiu: serà vàlida $POS_RET i la posició està dins de la esfera al voltant de $POS_RET.

FALSE:

ON_PATH està resetejat i $POS_RET no és vàlida, o la posició està fora de la esfera al voltant de $POS_RET.

Fitxer: $MACHINE.DAT

PRO_ACT: Aquesta sortida estarà activa quan hi hagi un procés actiu o bé la execució del programa en el nivell de robot estigui activa.

L'estat de senyal és derivada de la variable de sistema $PRO_STATE1:

$PRO_STATE1= #P_ACTIVE → $PRO_ACT = TRUE

Tota la resta d'estats de procés → $PRO_ACT = FALSE

El procés continua actiu mentre el programa o una interrupció segueixin processant-se. En cas d'aturada deguda a error, s'ha de diferenciar entre les tres possibilitats següents:

1. Si s'han activat interrupcions però no s'han tractat en el moment de la aturada degut a un error, el procés val com inactiu (PRO_ACT = FALSE).

2. Si s'han activat interrupcions i s'estaven processant en el moment de l'aturada degut a un error, el procés continuarà actiu (PRO_ACT = TRUE), fins que acabi el programa d'interrupcions o marxi fins a una ATURADA (PRO_ACT = FALSE).

3. Si s'han activat interrupcions i el programa d'usuari se'n va a una ATURADA, el procés es considera inactiu ($PRO_ACT = FALSE). Si després després d'aquest instant es compleix una condició d'interrupció, el procés romandrà actiu (PRO_ACT = TRUE), fins que s'hagi processat el programa d'interrupció o vagi cap a una ATURADA ($PRO_ACT = FALSE).

IN_HOME: Aquesta sortida comunica al ordinador principal si el robot es troba a la seva posició de partida (HOME).

ERR_TO_PLC: Activant aquesta sortida, el control del robot comunica al ordinador principal de que s'ha produït una falla de la unitat de control o de tecnologia.

La funció només està activa si el PLC_ENABLE té com a valor TRUE.

2.8.4 Altres variables

PGNO: En aquestes variables, el programa EXT_PGNO.SRC diposita el número de programa rebut (independentment del tipus de dades parametritzades) del ordinador principal com a valor numèric enter.

El programa d'organització específic de la tecnologia CELL.SRC assigna, a través d'aquestes variables, el número de programa al programa d'usuari corresponent.

PGNO_ERROR: Aquesta variable s'empra per a la gestió interna d'errors del programa EXT_PGNO.SRC i no podrà ser utilitzada o sobreescrita.

2.8.5 Declaracions

Aquesta taula està formada per els valors necessàris al fitxer "C:\KRC\Roboter\KRC\R1\System \$CONFIG.DAT

Nom del senyal	=	Notes
PGNO	0	Ocupació prèvia del número de programa.
PGNO_TYPE	1	Format de dades del número de programa: Binari
PGNO_FBIT	1	Situació del primer bit del número de programa.
PGNO_LENGTH	3	Ample del número de programa.
PGNO_PARITY	X	No contemplem la paritat.
PGNO_REQ	1	Requeriment d'un nou número de programa.
PGNO_VALID	4	El avís de que s'ha transmès el número de programa.
APPL_RUN	2	El avís de que s'està executant un programa
PGNO_ERROR	0	Ocupació prèvia de la marca d'error

Configuració de les entrades del fitxer \$MACHINE.DAT

Nom del senyal	E/S	Notes
$EXT_START	5 E	Posta en marxa externa.
$I_O_ACTCONF	3 S	Interface d'E/S activa.
$STOPMESS	4 S	Error d'aturada.
$CONF_MESS	6 E	Confirmació col·lectiva.

Ocupació de la interface:

Control	Nom del senyal
$IN[1]	PGNO bit 1
$IN[2]	PGNO bit 2
$IN[3]	PGNO bit 3
$IN[4]	PGNO_VALID
$IN[5]	$EXT_START
$IN[6]	$CONF_MESS
$IN[7]	$DRIVES_OFF
$IN[8]	$DRIVES_ON
$IN[9]	$MOVE_ENABLE

Control	Nom del senyal
$OUT[1]	PGNO_REQ
$OUT[2]	APPL_RUN
$OUT[3]	$I_O_ACTCONF
$OUT[4]	$STOPMESS
$OUT[5]	$PERI_RDY
$OUT[6]	$PRO_ACT

3. DESCRIPCIÓ DE LA SOLUCIÓ DONADA

Si recordem disposem de 4 robots del tipus KR 16 a la sala de formació de KUKA Robots Ibérica, S.A. Aquest robots tenen a la seva memòria gravats els programes que després des de la interfície voldrem posar en marxa.

3.1 PROGRAMES DELS ROBOTS

Els programes dels robots són diferents entre sí i emulen diverses tasques que et podries trobar en el desenvolupament d'un procés industrial. Tots aquests programes han estat desenvolupats per mi mateixa a les instal·lacions de KUKA a Vilanova i la Geltrú. La primera versió del programa es va fer mitjançant KRC Editor.

El KRC Editor ens permet programar a nivell expert i alhora fer servir el formulari inline (com podem observar al exemple de la figura Fig. 3.1) indicant la eina, la base, si té o no TCP extern, etc. És tracta d'una eina molt potent a l'hora de desenvolupar programes que després en mode manual caldria testejar al robot (de manera que puguem fet touch up en els punts determinats).

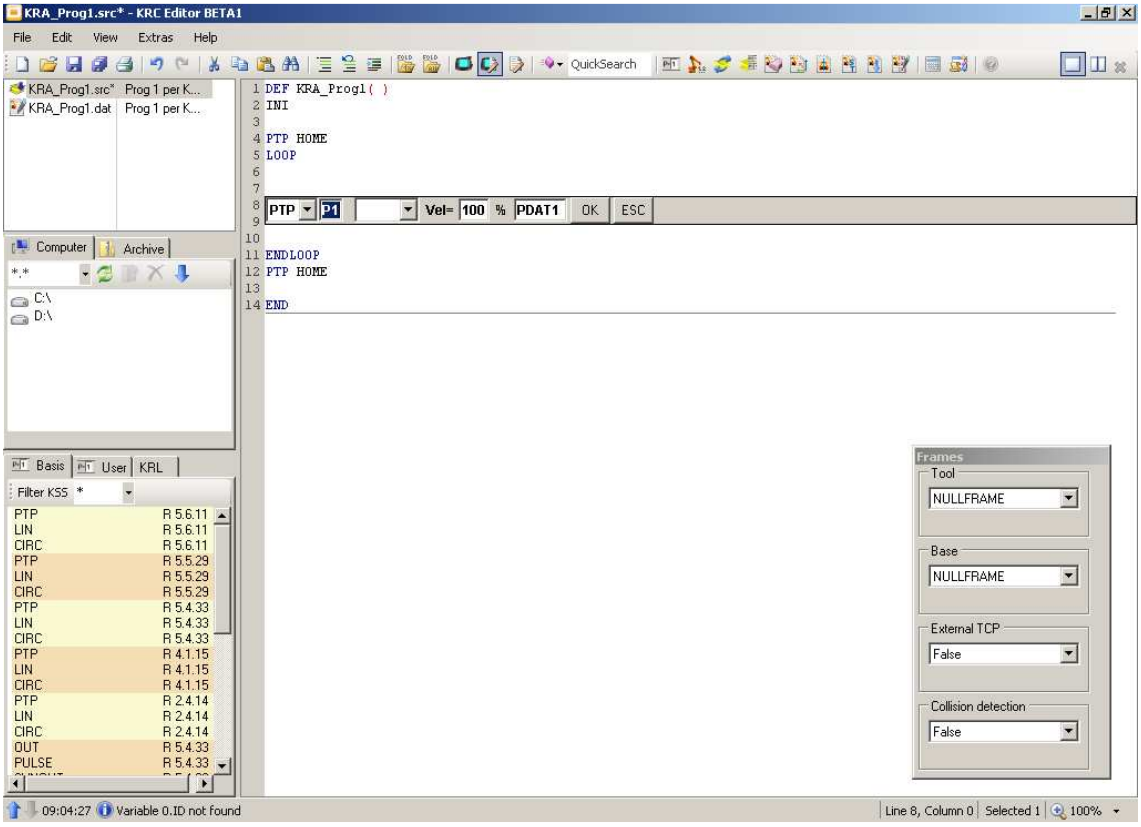


Fig. 3.1 - Exemple del programa KRC Editor.

Gràcies al editor, s'ha evitat escriure per complet el programa mitjançant la KCP. En aquest cas aquesta s'ha emprat per a comprovar el programa en mode manual així com per a determinar els punts per al arxiu.dat corresponent a cada programa. Un cop tenim el programa fet, igualment cal anar al robot i fer touch up de cada punt emprat en el programa i guardar-lo de la manera que correspongui tenint en compte la eina, la base, el tipus de moviment del canell del robot, etc.

3.1.1 Programes del primer KR 16

Hem considerat que el primer KR 16 es comportarà com un robot del tipus Pick & Place. Aquests robots són comuns en els processos industrials i acostumen a situar/posicionar elements dins d'un altre. Per exemple, si tinguéssim una placa PCB foradada, un robot pick & place posaria els components allà on toca (sense soldar-los).

Com tot robot de KUKA que es vulgui posar en marxa a través del mode automàtic extern necessitarem modificar el programa cell.src de manera que es pugui fer la crida dels diferents programes en enviar el número de programa.

A continuació veurem quins són els programes que he desenvolupat per al primer robot i que vull mostrar amb cadascun d'aquests programes.

1. Primer programa *KRA_Prog1* (potes al taburet):

Aquest és un programa senzill que mira de situar les potes d'un tamboret de disseny. Les potes, realment són 4 peces negres quadrades de 5cm de costat que el robot col·locarà en una superfície amb base plana on hem marcat els punts centrals on el robot deixarà la peça.

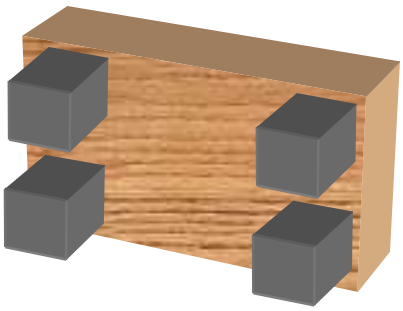


Fig. 3.2 - Simulació del resultat del programa.

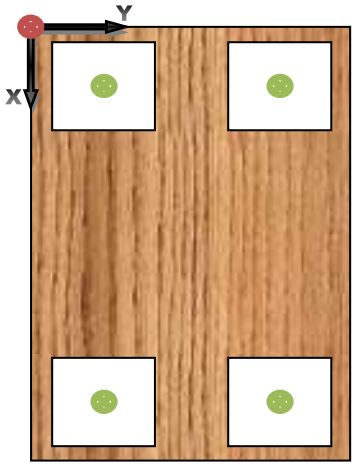


Fig. 3.3 - Situació de la base i de les peces.

La eina que té equipada el robot és una pinça pneumàtica que pot manipular aquest tipus de peces i al que li he fet el càlcul del TCP offset convenientment (com en vist en punts anteriors de la memòria). El programa està fet de manera que de moure la superfície on van les potes només caldria tornar a calcular ràpidament la base i la resta de punts es recalcularien convenientment.

A l'Annex d'aquesta memòria es trobarà el codi del programa (punt 8.1) així com l'esquema seguit en la programació a escala real de la peça. (punt 8.2).

Amb aquest programa es pretén demostrar que se sap posicionar les peces de forma correcta emprant punts que s'han marcat a mà respecte la base.

2. Segon programa *KRA_Prog2* (decoració del raburet):

En aquest cas es tractaria de presentar un programa que posicioni el mateix tamany de peça que el programa anterior però de metall. Aquestes peces però no se situen de forma recte respecte la situació de la base, sinó que estarien desplaçats en un angle de 45º. Afegim també la petita dificultat que ara les peces es troben més properes les unes de les altres.

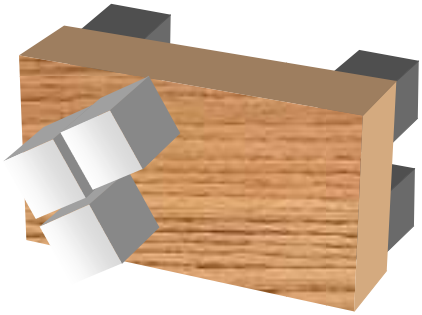


Fig. 3.4 - Simulació del resultat del programa

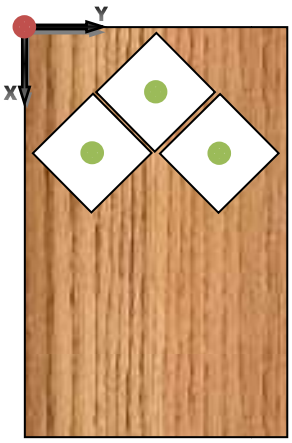


Fig. 3.5 - Situació de la base i de les peces.

La eina que té equipada el robot és una pinça pneumàtica que pot manipular aquest tipus de peces i al que li he fet el càlcul del TCP offset convenientment. El programa està fet de manera que tot i moure la superfície on van les potes només caldria tornar a calcular ràpidament la base i la resta de punts es recalcularien convenientment.

A l'Annex d'aquesta memòria es trobarà el codi del programa (punt 8.1) així com l'esquema seguit en la programació a escala real de la peça. (punt 8.2).

Amb aquest programa es pretén demostrar que se sap posicionar les peces de forma correcta, encara que es tracti de posicions inclinades respecte la base i amb menys espai per a la manipulació, emprant punts que s'han marcat a mà.

3.1.2 Programes del segon KR 16

El segon robot KR 16 el tractarem com a manipulador. Aquest manipulador s'encarregarà de tallar peces o serigrafiar peces. S'han determinat 3 programes diferents que incrementen en el nivell de programació respecte els programes del robot anterior.

1. Primer programa *KRB_Prog1* (Tall peça tipus 1)

En aquest cas, es tracta de repassar el contorn d'una peça determinada, com si es tingués una eina del tipus de tall. La eina talla en activar la sortida 1 (a la sala de formació es visualitzen les 16 primeres sortides en forma de led). En el nostre cas, no disposem d'eina de tall, però la pinça pneumàtica subjectant un sistema de bolígraf i activant/desactivant la sortida 1 tenim un comportament semblant al que tindríem en el cas real.

El programa serviria en els dos casos, però per a fer proves, ens serveix que el rotulador ressegueixi el patró a tallar de la peça mentre ens mostra el led 1 (pertinent a la sortida1) actiu.

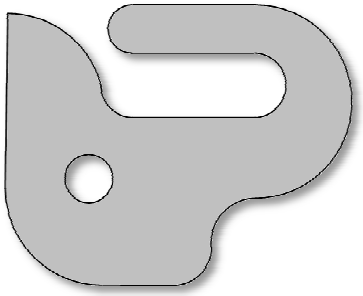


Fig. 3.6 - Simulació del resultat del programa

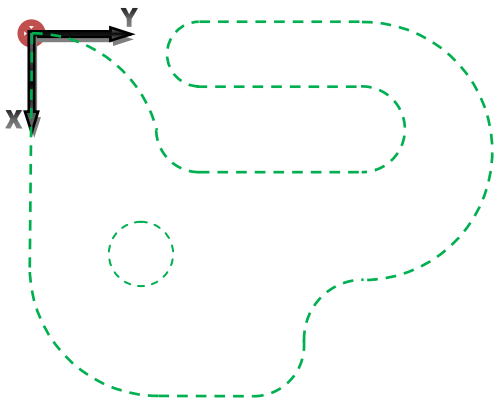


Fig. 3.7 - Situació de la base.

En aquesta peça en concret, es tracta de fer el programa que tallaria una peça amb moltes corbes i amb un tall en forma de circumferència en la part interna (com podem observar a la figura Fig. 3.6 - Simulació del resultat del programaFig. 3.6).

Aquest ja és un programa una mica més complex en que s'encadenen els punts per a realitzar una trajectòria a seguir. Aquesta peça tant es podria tallar en superfície plana com a corba, però, tant en un cas com en un altre és important determinar els punts amb la superfície amb la que es té previst treballar.

A l'Annex d'aquesta memòria es trobarà el codi del programa (punt 8.1) així com l'esquema seguit en la programació a escala real de la peça. (punt 8.2).

2. Segon programa *KRB_Prog2* (Serigrafiar "KUKA").

Per a fer-ho una mica més complicat encara, que en el cas anterior, aquest programa tracta d'escriure KUKA en el que representaria una placa. Com en el cas anterior la eina real que farem servir serà la pinça pneumàtica que subjectarà en tot moment una peça amb bolígraf. El càlcul del TCP es faria corresponentment per a tenir en compte el punt que guixa del boli o rotulador. No obstant, representa un procés industrial de serigrafiat en que cal activar la sortida 2 per a posar en marxa la eina. Veurem doncs que quan es mogui el robot traçant les lletres la sortida 2 estarà encesa i s'apagarà cada vegada que el robot s'elevi de la superfície de treball.

Fig. 3.8 - Simulació resultat final

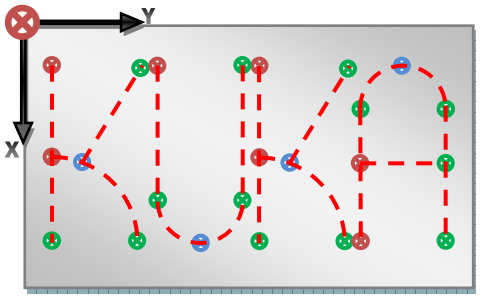


Fig. 3.9 - Situació de la base i dels punts.

Cal tenir en compte, que aquest programa ja fa servir moviments relatius a la base. És a dir, la base que es fa servir i que sempre es situarà com a la figura no només evita que en cas de moure la peça s'hagin de tornar a calcular els punts sinó que també ens ajuda en la programació el fet de tenir una base ben posicionada.

Aquest resulta un programa en que cal tenir molt present en quin punt ens trobem i cap a quina direcció i punt ens hem de moure (en termes absoluts o relatius).

A l'Annex d'aquesta memòria es trobarà el codi del programa (punt 8.1) així com l'esquema seguit en la programació a escala real de la peça. (punt 8.2).

3. Tercer programa *KRB_Prog3* (Tall peça tipus 2)

En aquest programa també es tractaria de repassar el contorn d'una peça determinada, com si es tingués una eina del tipus de tall. La eina talla en activar la sortida 1 (simulat amb 2n led de la cel·la robot). En el nostre cas, no disposem d'eina de tall, però la pinça pneumàtica subjectant un sistema de bolígraf i activant/desactivant la sortida 1 tenim un comportament semblant al que tindríem en el cas real. El programa serviria en els dos casos, però per a fer proves, ens serveix que el rotulador ressegueixi el patró a tallar de la peça mentre ens mostra el led 1 (pertinent a la sortida1) actiu.

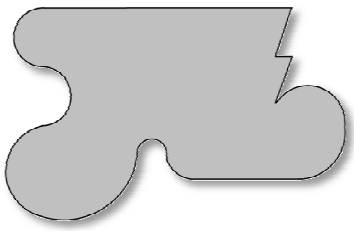


Fig. 3.10- Simulació resultat final.

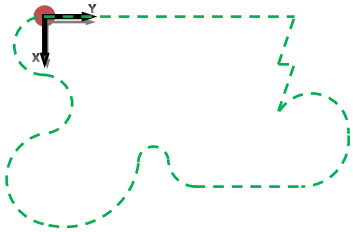


Fig. 3.11 - Situació de la base

En aquesta peça en concret, es tracta de fer el programa que tallaria una peça amb canvis de perfil sobtats i que no pot ser aproximat (com es pot observar a les figuresFig. 3.6 - Simulació del resultat del programa Fig. 3.10 i Fig. 3.11).

Aquest programa encadena els punts per a realitzar una trajectòria a seguir. Aquesta peça tant es podria tallar en superfície plana com a corba, però, tant en un cas com en un altre és important determinar els punts amb la superfície amb la que es té previst treballar.

3.1.3 Programes del tercer KR 16

El tercer robot (el manipulador B) tindria equipada una eina d'encolatge de sabateria. Si els dos models de sabata tenen la mateixa sola adherint diferents tipus de cobertura per a crear diversos tipus de sabata tenim la possibilitat de crear programes diferents per a la mateixa aplicació. Val a dir que s'ha escollit aquest plantejament perquè en les aplicacions d'adhesiu es tenen en compte situacions especials que no tenien lloc en les peces del robot anterior. Estem subjectes al càlcul.

Com bé us podeu imaginar, per a que l'adhesiu o cola, estigui repartit uniformement cal iniciar l'encolat quan ja es duu la velocitat de programa desitjada, així s'eviten punts d'acumulació de cola (és com en el cas de la tinta d'un bolígraf de tinta líquida, si deixes quiet en un punt abans d'iniciar un dibuix o escriptura deixarà una taca, mentre que si amb certa velocitat, es comença a guixar amb el bolígraf el traç serà regular). Ens interessa doncs, que en els instants en que s'hagi de començar a encolar, la sortida es posi en marxa en mig d'un moviment i per fer-ho al moment just calen càlculs i proves.

Toca, però, fer les proves amb la mateixa eina que en el robot 2 i simularem el moment d'obrir i tancar la cola amb la sortida 3 en cada programa.

1. Primer programa *KRC_Prog1* (Aplicar adhesiu tipus A)

En el primer programa d'encolat traçarem tan sols una sola línia de suposada cola (donat que no es farà servir cola de debò, com be he explicat amb anterioritat). Aquest programa pretén resseguir la bora d'una sola de sabata de dimensions conegudes per la part superior de la superfície, de manera que després se li pugui enganxar la part de la roba.

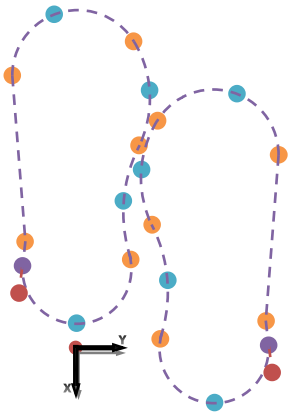


Fig. 3.12 - Situació de la base i punts.

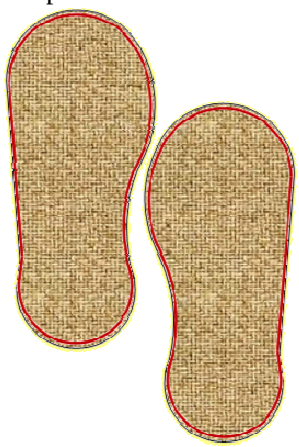


Fig. 3.13 - Simulació del resultat.

Com podem observar a la figura la situació de la base és va posar en inici aleatòriament. No obstant, sempre caldrà tornar a calcular-la en el mateix punt per a tenir sempre els punts en la mateixa zona. A l'Annex es troba la figura Fig. 3.12 a tamany real amb la situació exacta de la base.

A l'Annex d'aquesta memòria es trobarà el codi del programa (punt 8.1) així com l'esquema seguit en la programació a escala real de la peça. (punt 8.2).

2. Segon programa *KRC_Prog2* (Aplicar adhesiu tipus B)

En aquest cas, la sola de la sabata no serà encolada en la totalitat del seu entorn. Si bé en el cas anterior només hi havia un càlcul a efectuar, en aquest nou programa, hi apareixen 2 gairebé consecutius. Tampoc resseguirem tot el contorn de la sola si no s'hi ha d'aplicar cola, i s'executen les instruccions necessàries.

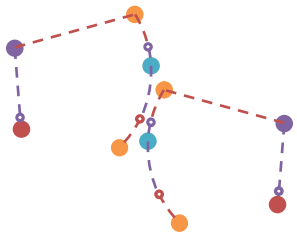


Fig. 3.14 - Punts i trajectòria del programa.

Fig. 3.15 - Simulació del producte

Es tracta d'un programa en la mateixa línia que l'anterior però una mica més elaborat en el preprogramat. Representa que es volen fer unes xancles d'estiu que només es subjecten al la part del davant per on passa el peu i amb els dits a l'aire.

A l'Annex d'aquesta memòria es trobarà el codi del programa (punt 8.1) així com l'esquema seguit en la programació a escala real de la peça. (punt 8.2).

3.1.4 Programes del quart KR 16

En el quart robot mirem de donar el rol al robot de paletitzador. Aquests programes, incorporen funcions lògiques més complexes que en els casos anteriors. Important menys que en cap altre cas el anar fent touch up en múltiples punts, en aquests exemples amb un mínim de punts es tracta de crear, amb eines lògiques, paletitzats i despaletitzats de peces.

Es té molt en compte el posicionament relatiu i les funcions de repetició o condició en els propers 4 programes. Així també s'evita la consecució d'un codi extens i repetitiu.

Es pren com a eina la pinça pneumàtica que tenen els robots de la sala de formació. A aquesta eina ja li he calculat el TCP offset i la base es pot posicionar quan calgui en la base 1.

1. Primer programa *KRD_Prog1* (Paletitzar 3x3)

El primer programa que ha de fer un paletitzador és, precisament, paletitzar. Així doncs, el primer programa s'encarregarà de paletitzar 9 peces en forma de cub de 5cm de costat en un palet de 3 files i 3 columnes. El palet serà pla, en el sentit que no estarà situat en zones corbes com qualsevol palet convencional.

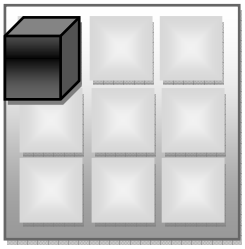


Fig. 3.17 - Resultat esperat

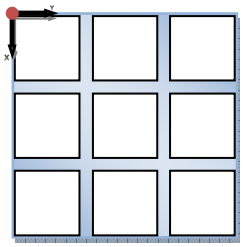


Fig. 3.16 - Esquema amb base.

Entre peces hi haurà una distància d'1cm i totes les posicions del palet seran relatives al punt calculat com a punt de palet. Un cop hagi col·locat les 9 peces el robot s'aturarà i acabarà el programa.

A l'Annex d'aquesta memòria es trobarà el codi del programa (punt 8.1) així com l'esquema seguit en la programació a escala real de la peça. (punt 8.2).

2. Segon programa *KRD_Prog2* (Despaletitzar 3x3)

Aquest programa tracta de fer el procés anterior a la inversa, és a dir, si tenim un palet de 3x3 torna aquestes peces a l'acumulador de peces (per la part d'adalt). És important comentar que els punts dels dos programes són diferents en quant a on s'agafaven les peces i on les hem de deixar ara.

Estem considerant doncs, que el palet que estem despaletitzant és el elaborat mitjançant el KRD_Prog1.

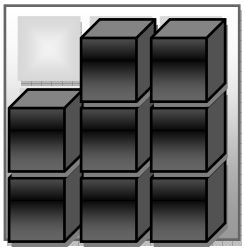


Fig. 3.19 - Resultats esperat

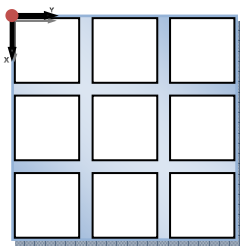


Fig. 3.18 - Esquema amb base.

Entre peces hi haurà una distància d'1cm i totes les posicions del palet seran relatives al punt calculat com a punt de palet. Un cop hagi col·locat les 9 peces el robot s'aturarà i acabarà el programa.

A l'Annex d'aquesta memòria es trobarà el codi del programa (punt 8.1) així com l'esquema seguit en la programació a escala real de la peça. (punt 8.2).

3. Tercer programa *KRD_Prog3* (Paletitzar en fila 5)

Si bé hem vist com es paletitzaria una matriu de peces, no s'ha vist com es faria una fila de peces, que en aquest cas serà de 5 (per tal crear una columna de peces, s'actuaria de la mateixa manera canviant l'eix X-Y de l'increment relatiu).

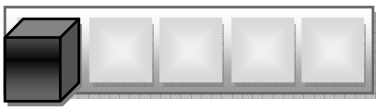


Fig. 3.20 - Simulació resultat primera peça

Es tracta d'un programa similar als altres dos i que comprèn el mateix tipus de lògica.

Entre les peces de la fila hi haurà una distància d'1cm i aquestes distàncies seran relatives al punt de palet. Un cop paletitzades les 5 peces el programa acabarà.

Si bé a l'Annex d'aquesta memòria es trobarà el codi del programa (punt 8.1), no s'hi ha afegir l'esquema que s'ha seguit en la programació a escala real de la peça. No obstant, comentar que la ubicació de la base seria la mateixa que per a la primera fila del paletitzat 3x3 (incloent-hi 2 peces més a la dreta).

4. Quart programa *KRD_Prog4* (Despaletitzar fila de 5)

En aquest cas despaletitzariem la situació anterior, és a dir, aquest palet només despaletitza una fila de 5peces que es trobin separades a 1cm.

Fig. 3.21 - Simulació resultat despaletitzat de la primera peça

La lògica empleada en aquest programa es manté al nivell de la lògica utilitzada a la resta de programes del quart robot.

NOTA:

Aquesta és una captura d'una de les pinces pneumàtiques equipades als robots de la sala de formació. Com podem observar les peces que manipula son cubs de 5cm de costat que tenen unes marques al mig de totes les cares per a que sigui més fàcil la seva manipul·lació.

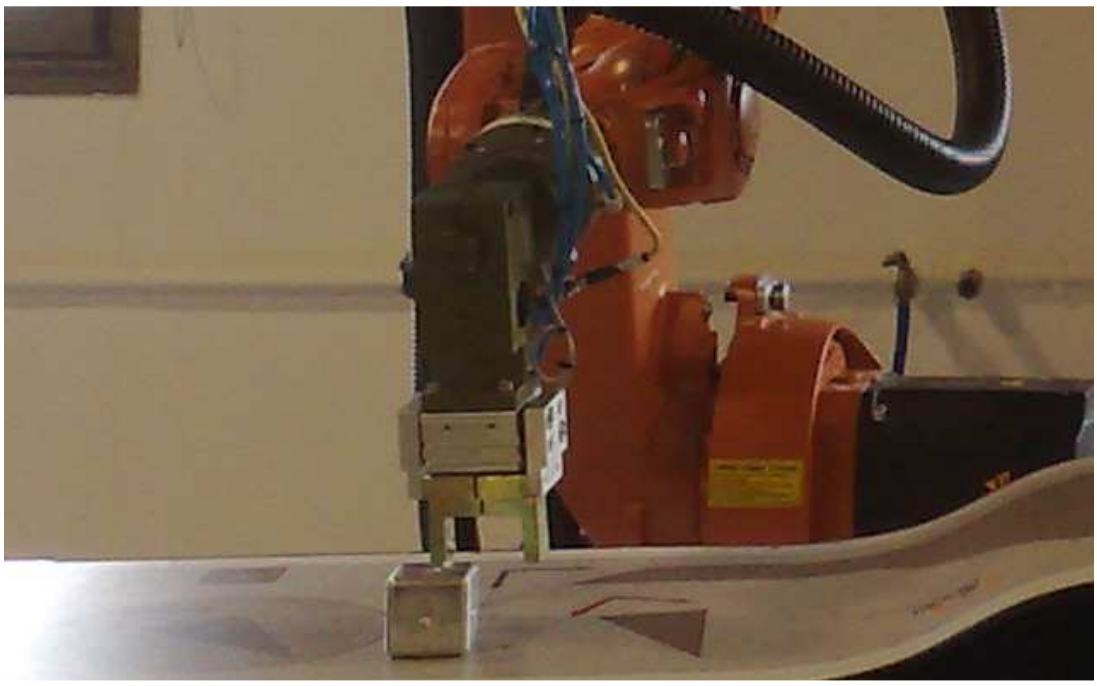


Fig. 3.22 - Eina equipada als robots de la sala de formació: Pinça pneumàtica.

3.2 PANTALLA HMI AMB HMI STUDIO

Aquest és un dels aspectes claus del projecte, donat que és la part que el client acaba veient. El més important és fer fàcil, visual i funcional la aplicació que ha de córrer per sobre de la execució del programa. Del que en surti de la interfície HMI i les seves pantalles derivarà la resta d'elements.

3.2.1 Estructuració de les pantalles

Abans de començar amb el disseny de les pantalles, és important tenir present com es pretén estructurar la interfície. Si tindrà o no tindrà pantalla principal, si tindrà menú de navegació, si no hi haurà navegació ja que tot estarà condensat en una sola pàgina... etc. En aquest subcapítol explicaré quina estructuració de pantalla i navegació hem escollit i per quines raons.

Primer, cal tenir present que es tracta d'una interfície que ha de governar sobre 4 robots. Llavors és primordial separar la informació de cadascun d'ell de la manera més senzilla possible. El més lògic és disposar d'una pantalla de visualització/manipulació per a cada robot (de forma que no es pugui confondre alhora de modificar algun paràmetre d'un robot en concret). I si tenim 4 robots una pantalla per cada robot esdevé 4 pantalles de tipus robot. No obstant, per a accedir a aquestes pantalles específiques necessitarem una pantalla principal, i de passada podem emprar aquesta pantalla per a visualitzar l'estat dels 4 robots de forma conjunta (en la pantalla principal no podrem modificar cap paràmetre, serà només de visualització, per a evitar que es pugui modificar, per error, les dades del robot equivocat). Així doncs, per a resoldre l'aplicació calen 5 pantalles.

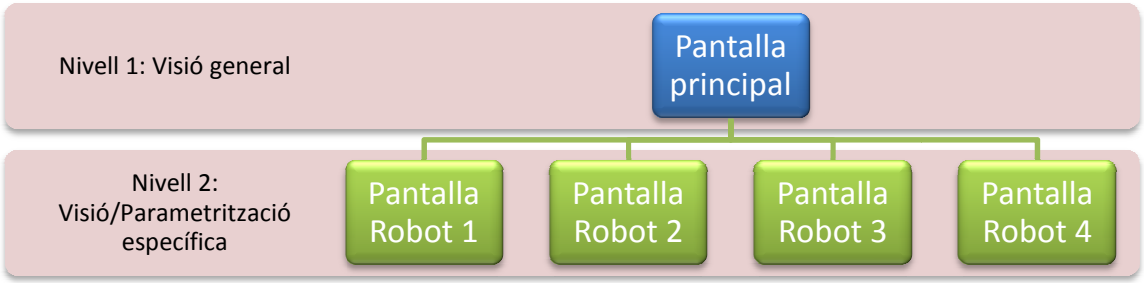


Fig. 3.23 – Estructura de pantalles

No obstant és important que, es tinguin en compte elements importants com ara la llengua. En quina llengua cal realitzar la aplicació? Per a evitar haver d'escollir-ne una, aquesta aplicació està desenvolupada en dos idiomes, l'anglès i l'espanyol. Per a cada llengua tenim 5 pantalles (com hem comentat al paràgraf anterior). Amb la qual cosa per la aplicació sencera amb 2 idiomes resultaran 10 pantalles diferents.

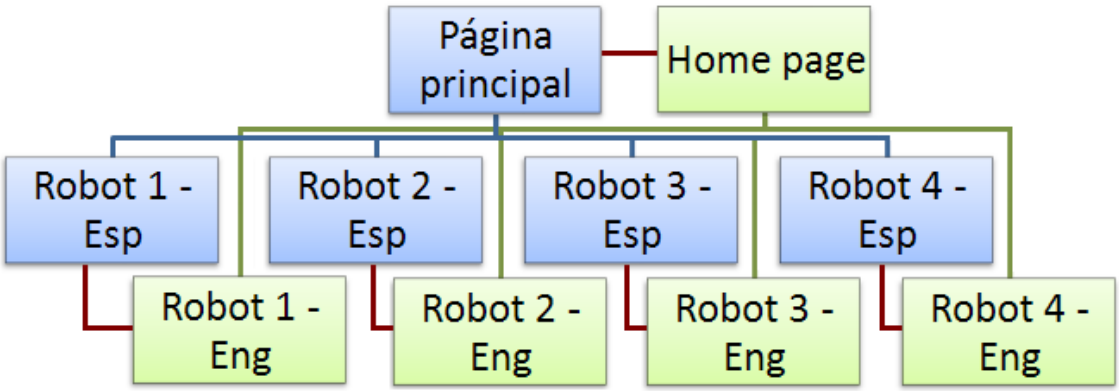


Fig. 3.24 – Estructura de pantalles tenint en compte 2 idiomes

Podrem saltar d'idioma a qualsevol pantalla de manera que pots estar a la estructura en arbre de color blau (Espanyol) i decidir passar a l'anglès en qualsevol moment, llavors passes a la mateixa pantalla en que es trobi l'operari en aquell instant però de la estructura en arbre verda (anglès). En color grana veiem com els canvis de l'idioma efectuen el salt entre les dues estructures en arbre.

Així penso que resolem el problema del idioma alhora que mantenim els dos nivells de pantalles i podem anar a qualsevol pantalla des de qualsevol pantalla.

3.2.2 Disseny

Abans de començar amb qualsevol disseny és summament important determinar de quina manera volem distribuïda la informació a la pantalla i fer un esquema de la estructura de cada tipus de pantalla (on apareguin la situació dels diferents elements a la pantalla). Si recordem tenim 2 tipus de pantalles/finestres: la principal general, les de robot específiques.

Per tal de seguir un ordre, veurem primer la estructura de les pantalles principals (recordem que tenim 2 una per a cada idioma, que seran iguals en contingut).

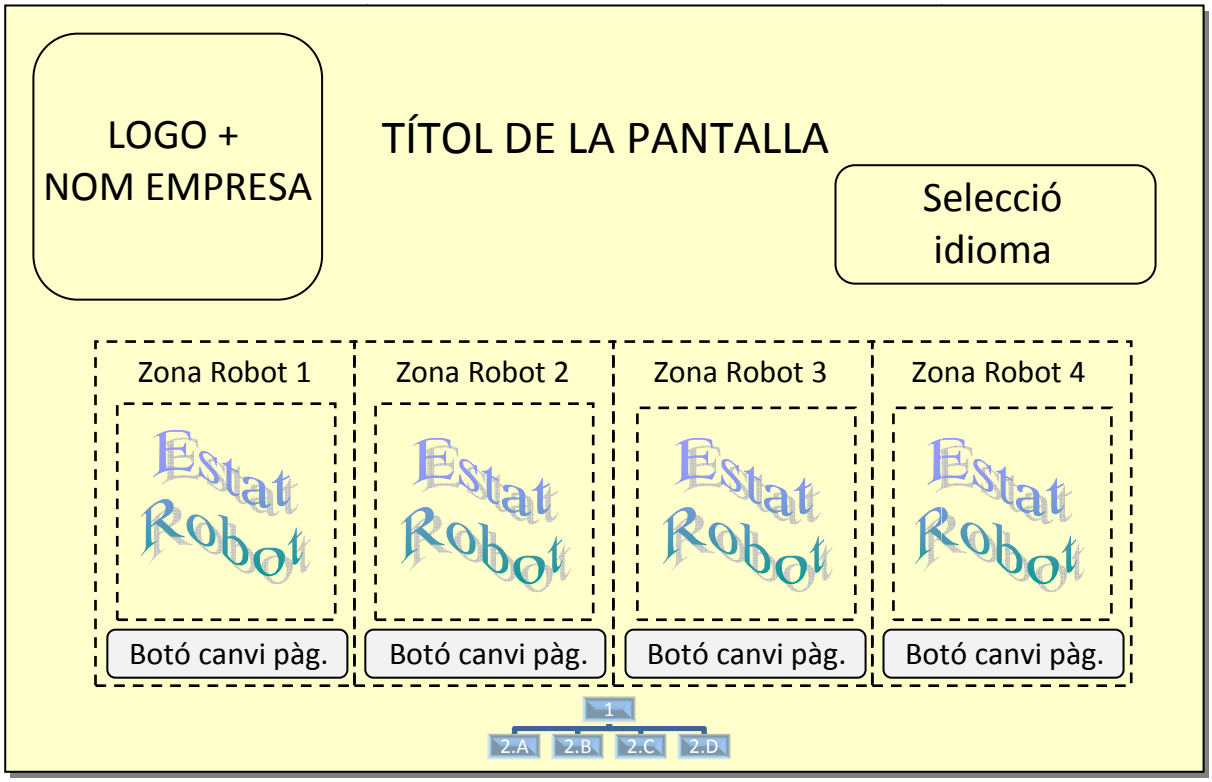


Fig. 3.25 - Estructura de la pantalla principal

Com podem observar de [illegible] ıpresa i el seu nom a la part esquerra de la pantalla (zona de menor interès). A la dreta trobem els botons de selecció de l'idioma i al centre de la part més alta de la pantalla i trobem el títol d'aquesta. A sota de tot això, repartits en 4 zones diferents, trobem el nom de cada robot i el seu número, els indicadors de l'estat del robot i el botó per accedir a la pàgina específica d'aquell robot (propers a la part baixa de la pantalla on acostumen a situar-se els menús de navegació). La pantalla principal doncs queda definida per la seva senzillesa, la distribució acotada dels diferents elements. Per a facilitar també la visualització de la estructura de pantalla, en la part més baixa trobem un esquema d'aquesta navegació on es marcaria la situació actual al usuari.

En quant a les pantalles específiques dels robots, seran realment semblants. Variarà el nom del robot (de cara a que ràpidament es pugui veure de quina pantalla específica es tracta) però la

resta de la estructura de la pàgina serà molt similar a les dels altres robots. Dins les pantalles específiques de robot també cal tenir en compte que tenim elements de visualització, elements de parametrització i elements que combinen visualització i parametrització. Tot seguit veurem els diferents elements que apareixeran en les pantalles de robots:

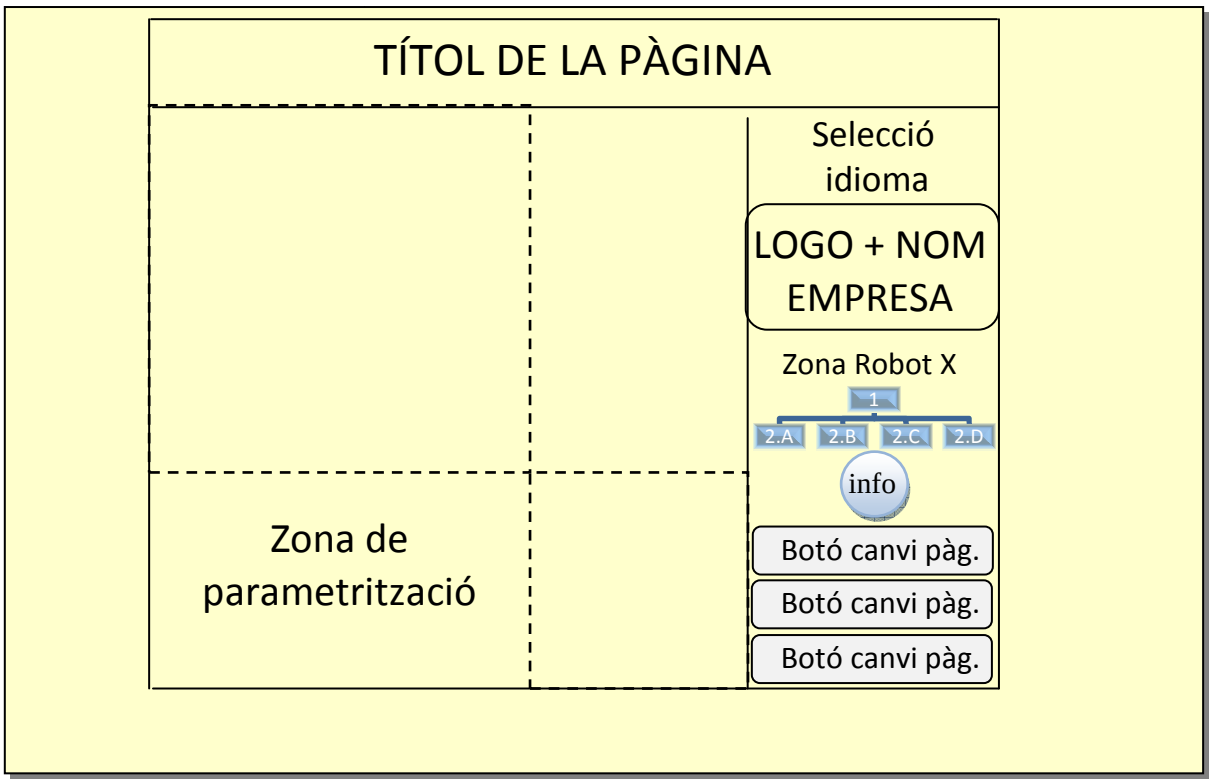


Fig. 3.26 - Estructura de la pantalla específica del robot

La informació més general de la finestra com ara el logo de la empresa, els botons de navegació la selecció de l'idioma, la assistència de pantalla, etc apareixerà a la dreta separat per una línia pronunciada, això formaria part del menú. A la part més amplia, estan ubicats els elements de parametrització i control del estat del robot. D'aquesta manera queda de nou tot localitzat en diferents zones i l'usuari pot tenir més clar on trobar els diferents elements.

Un cop vistes les plantilles de les pantalles, passarem a veure realment de quina manera és veuen aquestes en el mode d'execució i així interpretar amb més propietat les seves característiques.

Recordem que es tracta de 5pantalles per a cada idioma i que alhora les de robot tenen a la disposició de l'usuari la opció de visualitzar, sobre la pantalla mateixa, el text d'ajuda. Aquest text apareix en l'idioma seleccionat i facilita la interpretació de les funcionalitats sense que aquestes deixin de funcionar.

Les funcionalitats dels indicadors i dels elements de parametrització es veuran al proper subcapítol, amb la qual cosa en aquest apartat només es té en compte el disseny gràfic de les diferents pantalles.

Per començar veurem la pantalla en mode execució de la pantalla principal en Espanyol. En aquest instant de la captura els robots 1, 2, i 4 estan en marxa, mentre que el robot 3 està aturat.

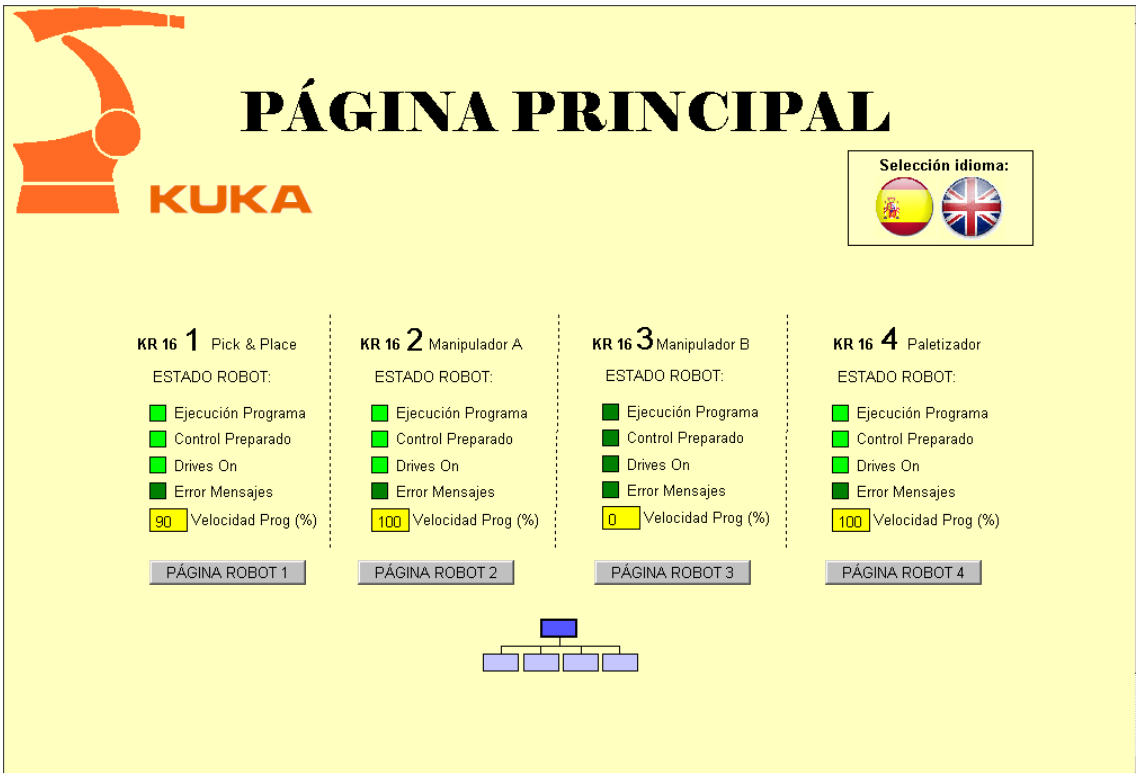


Fig. 3.27 - Pantalla principal en espanyol.

La mateixa pàgina i la mateixa situació seleccionant l'idioma anglès es veu de la forma següent:

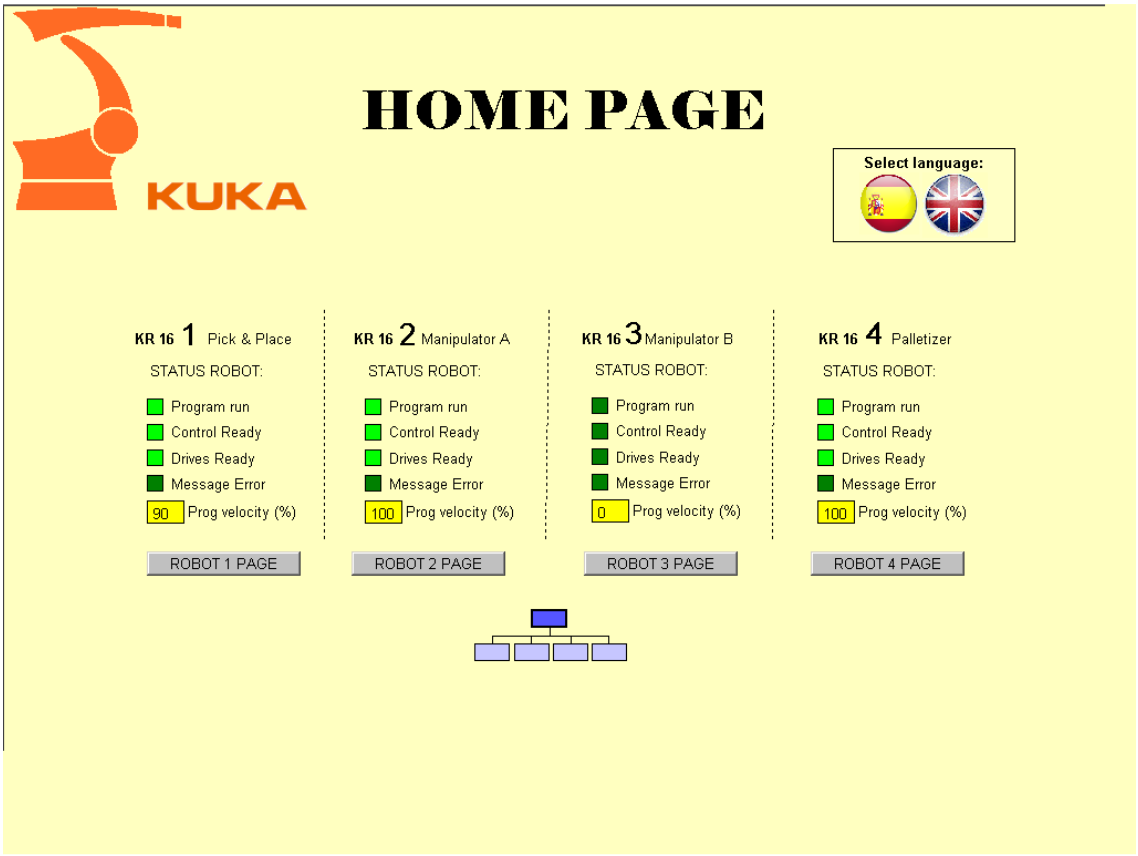


Fig. 3.28 - Pantalla principal en anglès.

En quant a les pantalles específiques de robot, en la mateixa situació, en espanyol seria:

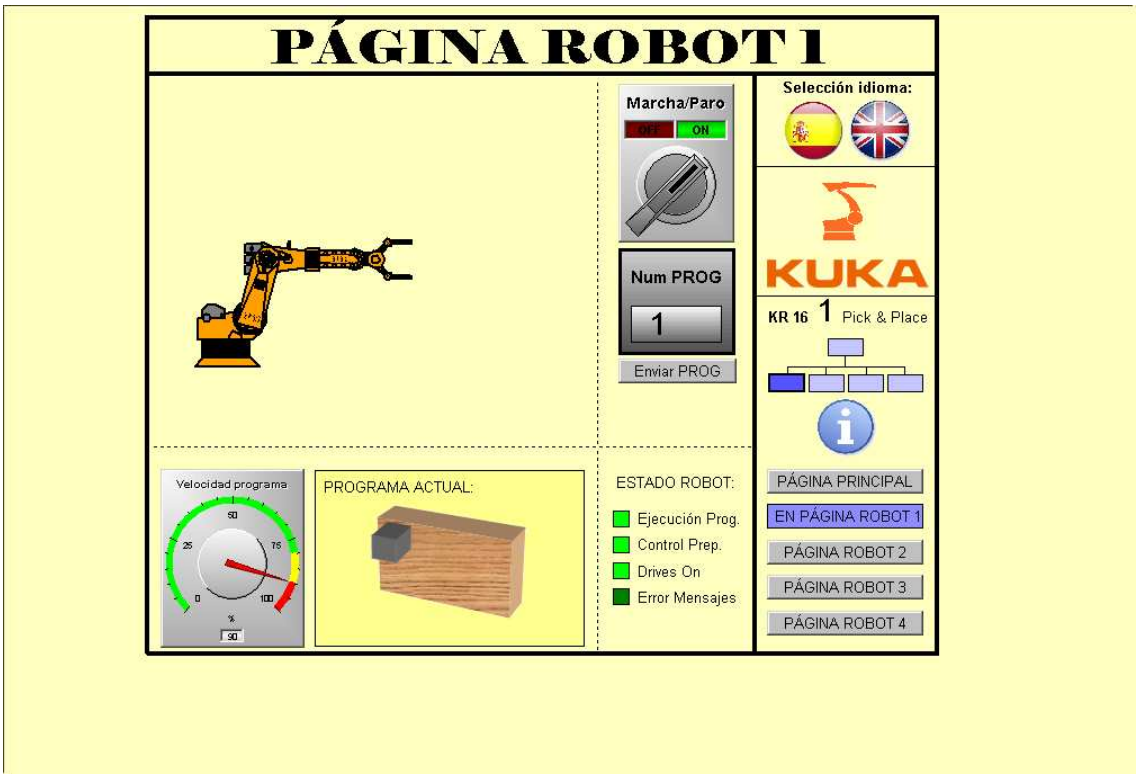


Fig. 3.29 - Pàgina de robot 1 en espanyol.

Si visualitzem el text d'ajuda la pantalla quedaria:

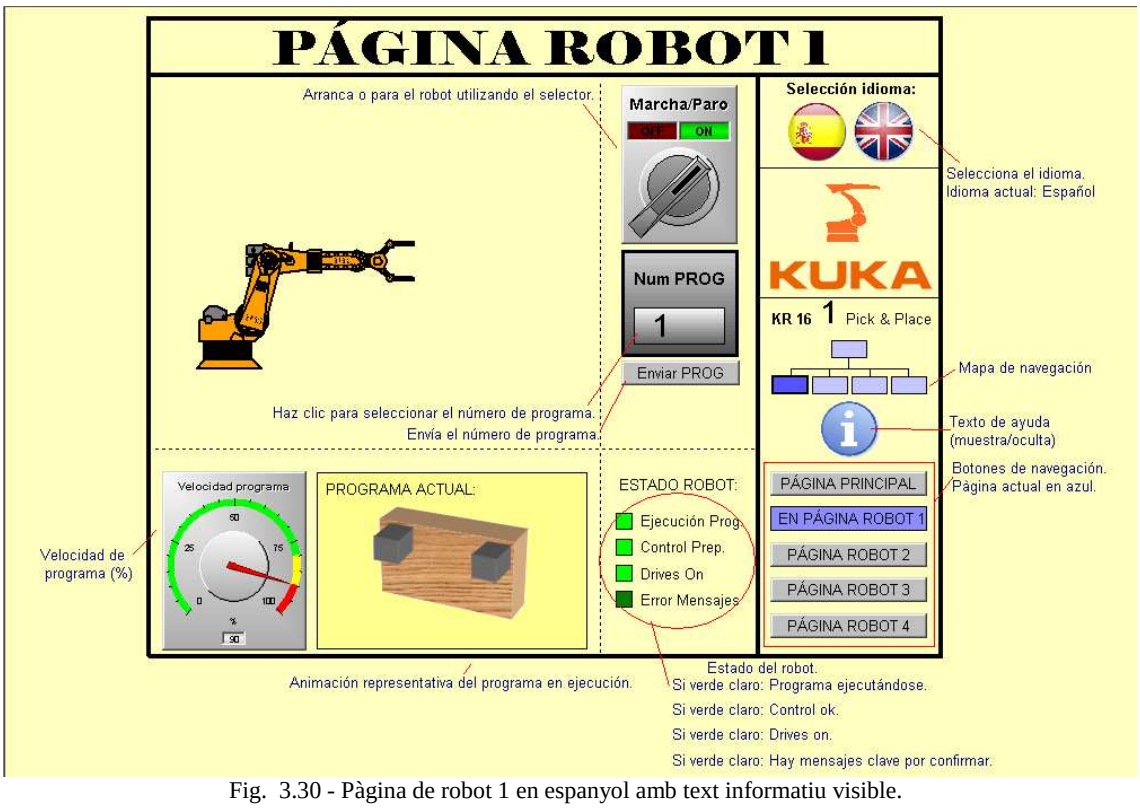


Fig. 3.30 - Pàgina de robot 1 en espanyol amb text informatiu visible.

La mateixa situació que la anterior emprant l'idioma anglès seria:

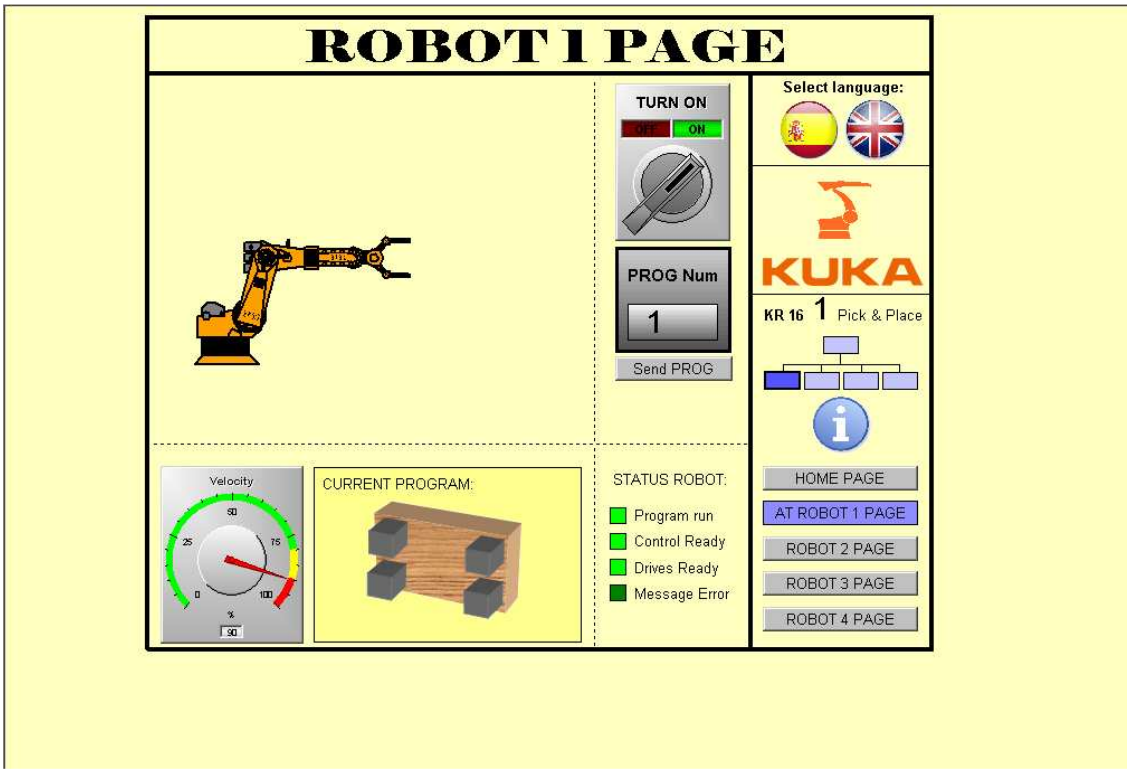


Fig. 3.31 - Pàgina de robot 1 en anglès.

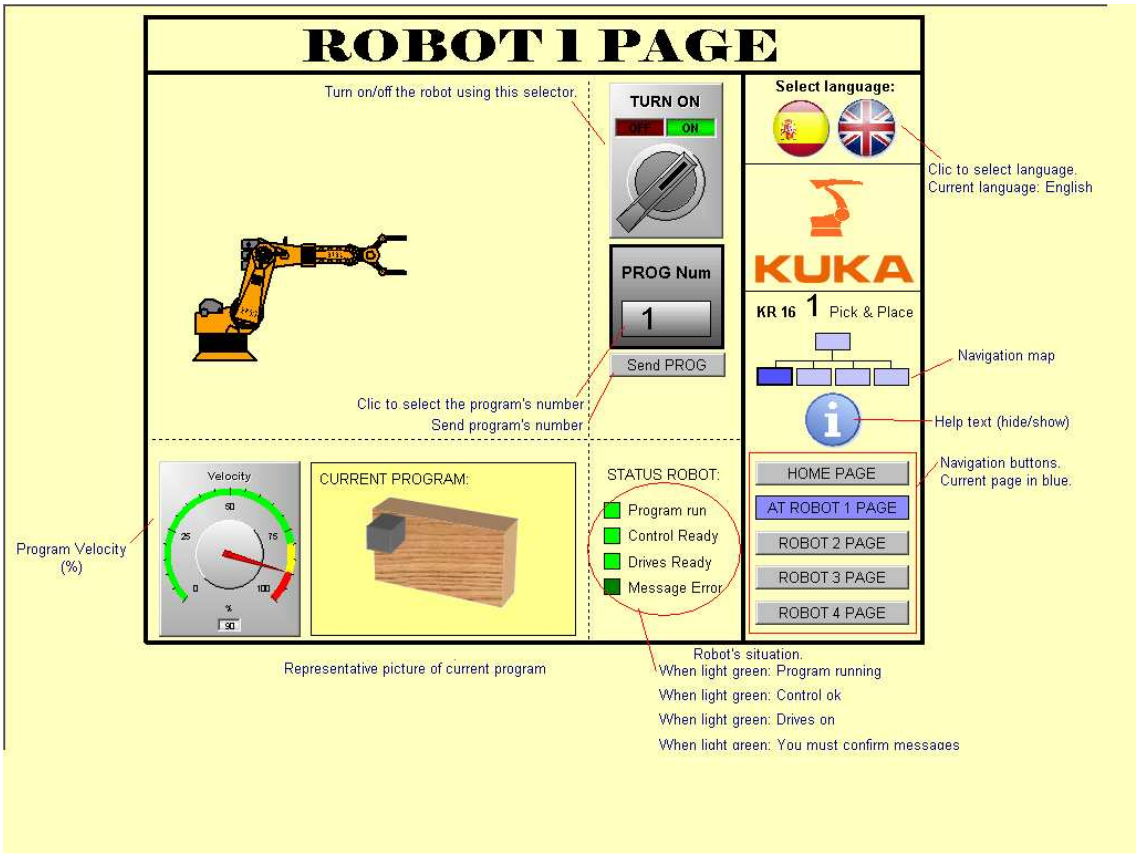


Fig. 3.32 - Pàgina de robot 1 en anglès amb text informatiu visible.

Com hem pogut observar tant en espanyol com en anglès el text informatiu apareix als marges de la pantalla de manera que no ens impedeix veure el normal funcionament de la pantalla ni ens tapa informació de cap tipus. Aquest aspecte de disseny també s'ha tingut en compte en la elaboració de les pantalles.

Un cop vist com es distribueixen les pantalles és moment de entrar més en detall amb les funcionalitats d'aquestes pantalles. Tot el que cal saber es troba a l'apartat 3.2.3 de Funcionalitats.

3.2.3 Funcionalitats

Les pantalles s'han realitzat, amb l'objectiu de ser funcionals, és a dir, que aportin informació i permetin incorporar canvis o decisions sobre els robots. Per a aquest motiu s'han configurat una sèrie de botons i accionaments que ens permeten actuar sobre el robot. A continuació descriurem la funcionalitat dels diferents dispositius que inclouen les pantalles, a mode de guia. Aquestes funcionalitats estaran desdoblades en els casos que tinguin nomenclatura diferent en espanyol i en anglès.

PANTALLA PRINCIPAL:

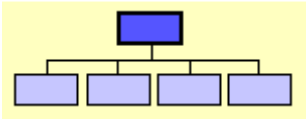
Indicadors lluminosos d'estat del robot: Dins de "Estado robot"/"Status robot" ens trobem amb un seguit d'indicadors lluminosos amb diferents rètols. Cada rètol indica quina variable o estat estem visualitzant i l'indicador verd estarà il·luminat (verd clar) quan el paràmetre del rètol sigui cert i estarà apagat (verd fosc) quan el paràmetre del rètol sigui fals.

Botó idioma: bandera Espanya: Aquest botó ens permet seleccionar l'idioma espanyol.
bandera Anglaterra: Aquest botó ens permet seleccionar l'idioma anglès.

Velocitat de programa "Velocidad Prog (%)"/"Prog Velocity (%)": Aquest indicador ens mostra el valor analògic de la velocitat de programa actual. Aquest valor es comprèn entre 0 i 100 ambdós inclosos. El valor no es pot editar des de la pantalla principal.

Botons de navegació "PÁGINA ROBOT X"/"ROBOT X PAGE": En prémer qualsevol botó de salt de pàgina s'obrirà una nova pantalla que reemplaçarà la que teníem. El títol dels botons ens indica la pàgina que obre.

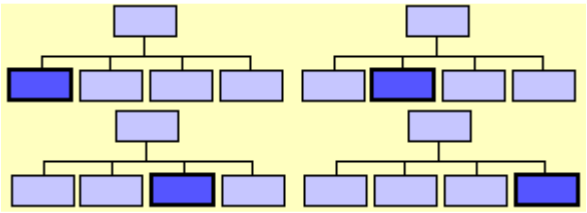
Indicador/navegació situació de la pantalla actual: Aquest element ens indica en blau més fosc quina pantalla es mostra actualment. A més a més, els quadres de blau més clar, en fer clic, actuen com els botons de navegació reemplaçant la pantalla actual per la seleccionada a la estructura de navegació.



PANTALLES ESPECÍFIQUES

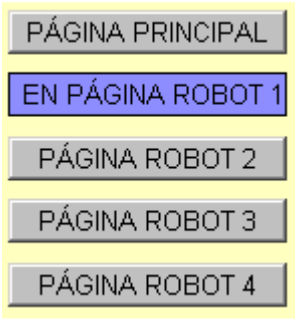
Botó idioma: bandera Espanya: Aquest botó ens permet seleccionar l'idioma espanyol.
bandera Anglaterra: Aquest botó ens permet seleccionar l'idioma anglès.

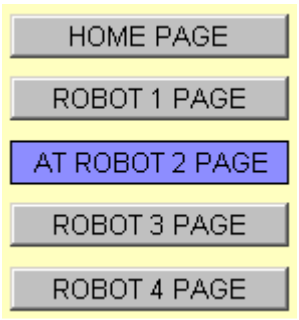
Indicador/navegació situació de la pantalla actual: Aquest element ens indica en blau més fosc quina pantalla es mostra actualment. A més a més, els quadres de blau més clar, en fer clic, actuen com els botons de navegació reemplaçant la pantalla actual per la seleccionada a la estructura de navegació.



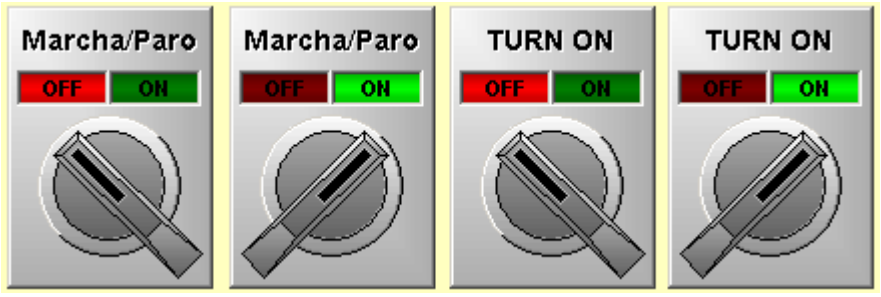
Text d'ajuda: Aquest botó rodó que recorda els símbols d'informació generals pretén mostrar i ocultar un text d'ajuda de la pantalla. Per defecte roman amb la informació oculta i quan es prem el botó apareix la informació per pantalla. En tornar a prémer el botó la informació es torna a ocultar (A mode d'exemple, veure les figures XX i YY del subcapítol anterior 3.2.2)

Botons de navegació "PÁGINA ROBOT X"/"ROBOT X PAGE" i "PÁGINA PRINCIPAL"/"HOME PAGE": En prémer qualsevol botó de salt de pàgina s'obrirà una nova pantalla que reemplaçarà la que teníem. El títol dels botons ens indica la pàgina que obre. Per a fer més visual la navegació entre pantalles aquestes estan ordenades en columna i mostren un quadre en blau especificant en quina pàgina ens trobem. Aquest quadre blau no és un botó.

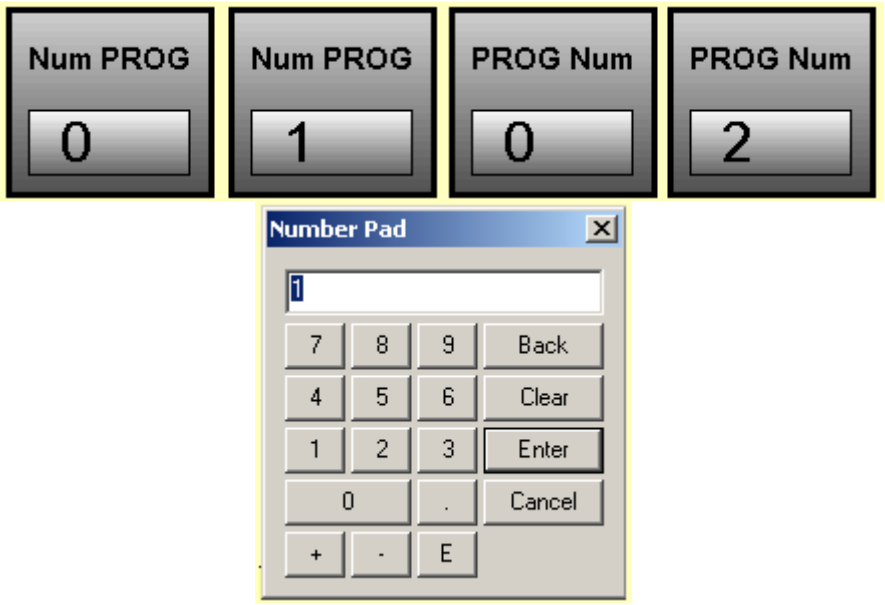




Botó "Marcha/Paro"/"Turn on": Aquest botó ens permet posar en marxa i aturar el robot de la mateixa manera que ho faria l'interruptor principal de l'armari del robot. Amb l'interruptor principal es connecta o desconnecta el sistema de robot i la unitat de control.



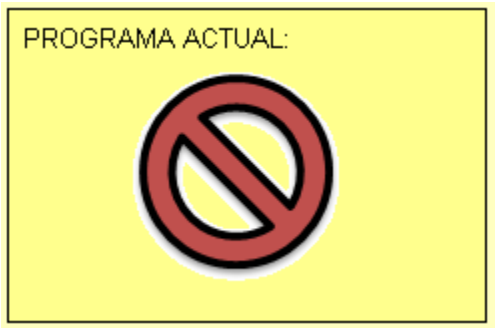
Selector numèric “Num PROG”/“PROG Num”: Aquest selector enter numèric ens permet indicar quin és el número de programa que volem executar al robot corresponent. Cal indicar un valor que tingui un programa associat al cell.src, en cas contrari, ens remetrà un error. Per a entrar el valor, es fa clic sobre el valor numèric i aquest desplegarà una finestreta emergent que ens permetrà entrar el valor tant mitjançant teclat com emprant ratolí.



Botó “Enviar PROG”/“Send PROG”: Aquest botó envia el número indicar al selector numèric PROG Num al robot de manera. Aquest botó doncs, envia la dada.

Indicadors lluminosos d’estat del robot: Dins de “Estado robot”/“Status robot” ens trobem amb un seguit d’indicadors lluminosos amb diferents rètols. Cada rètol indica quina variable o estat estem visualitzant i l’indicador verd estarà il·luminat (verd clar) quan el paràmetre del rètol sigui cert i estarà apagat (verd fosc) quan el paràmetre del rètol sigui fals.

Visor de programa en execució “PROGRAMA ACTUAL”/“CURRENT PROGRAM”: Aquest quadre mostra una animació que fa referència al tipus de programa que el robot està executant segons el número de programa. Quan no està executant cap programa apareix una imatge amb el símbol de prohibit que ens intenta explicar que no hi ha res en procés. Per a fer més entenedors els diferents programes ara mostrarem totes les opcions que hi ha:



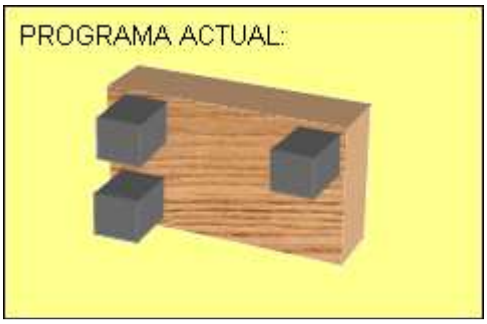
Tipus: Imatge.

Identificació: Símbol prohibició.

Programa Núm: 0 o no programats.

Significat: No hi ha cap programa en execució en aquests instants.

Situació: A tots els robots.



Tipus: Animació (5 frames)

Identificació: Posicionador de peces (potes).

Programa Núm: 1.

Significat: El programa en execució situa les potes allà on toca.

Situació: Al robot 1 (Pick & Place).

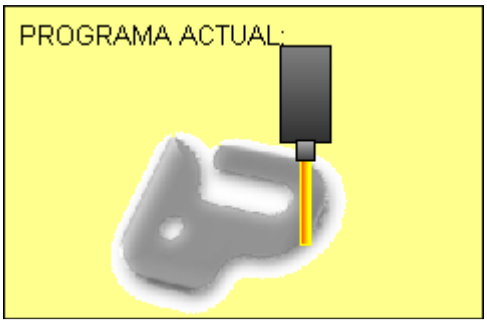
Tipus: Animació (4 frames)

Identificació: Posicionador de decoració.

Programa Núm: 2.

Significat: El programa en execució situa les peces platejades allà on toca.

Situació: Al robot 1 (Pick & Place).



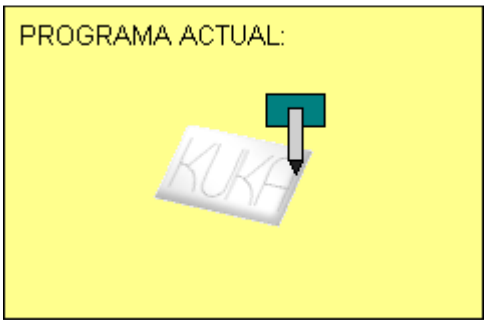
Tipus: Animació (contínua)

Identificació: Peça tallada a laser.

Programa Núm: 1.

Significat: El programa en execució talla amb làser una figura com la del dibuix.

Situació: Al robot 2 (Manipulador A).



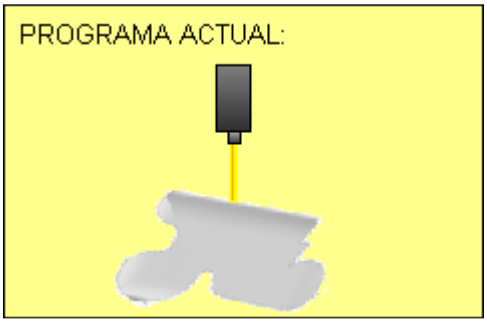
Tipus: Animació (contínua)

Identificació: Plaqueta KUKA

Programa Núm: 2.

Significat: El programa en execució serigrafia KUKA en una peça.

Situació: Al robot 2 (Manipulador A).



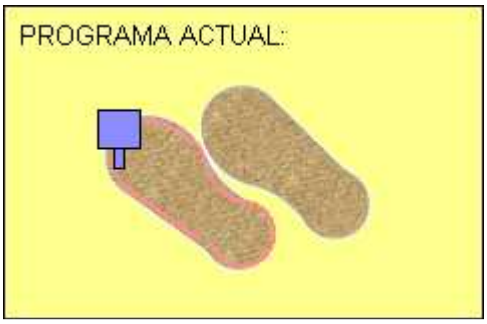
Tipus: Animació (contínua)

Identificació: Peça tallada a laser.

Programa Núm: 3.

Significat: El programa en execució talla amb làser una figura com la del dibuix.

Situació: Al robot 2 (Manipulador A).



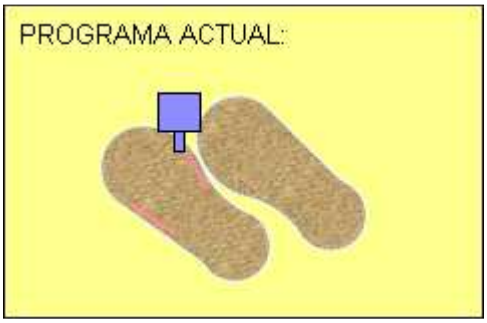
Tipus: Animació (contínua).

Identificació: Peces omplint palet.

Programa Núm: 1.

Significat: El programa en execució poda adhesiu a les vores d'una sola de sabata.

Situació: Al robot 3 (Manipulador B).



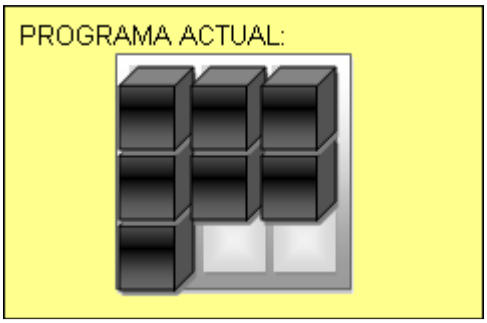
Tipus: Animació (10 contínua).

Identificació: Peces omplint palet.

Programa Núm: 2.

Significat: El programa en execució poda adhesiu a les zones on va la roba de les sandàlies.

Situació: Al robot 3 (Manipulador B).



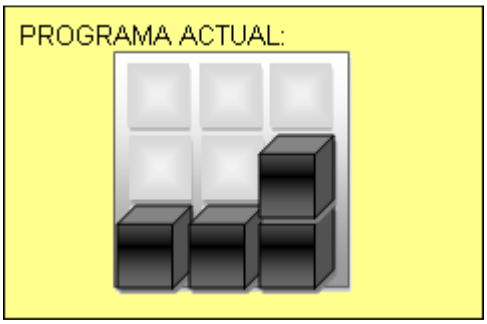
Tipus: Animació (10 frames).

Identificació: Peces omplint palet.

Programa Núm: 1.

Significat: El programa en execució és un paletitzat de 3x3.

Situació: Al robot 4 (Paletitzador).



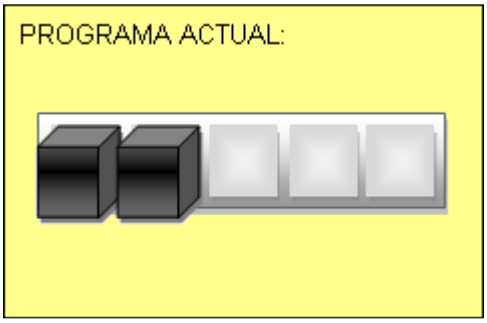
Tipus: Animació (10 frames).

Identificació: Peces desapareixent del palet.

Programa Núm: 2.

Significat: El programa en execució és un despaletitzat de 3x3.

Situació: Al robot 4 (Paletitzador).



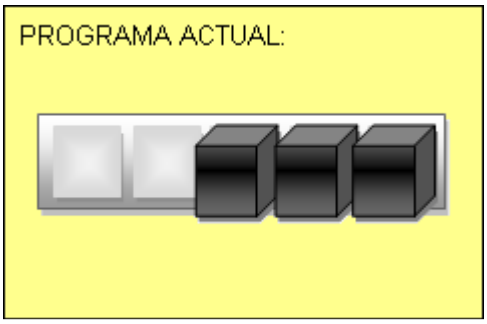
Tipus: Animació (6 frames).

Identificació: Peces omplint línia.

Programa Núm: 3.

Significat: El programa en execució és un paletitzat de 1x5.

Situació: Al robot 4 (Paletitzador).



Tipus: Animació (6 frames).

Identificació: Peces desapareixent de línia.

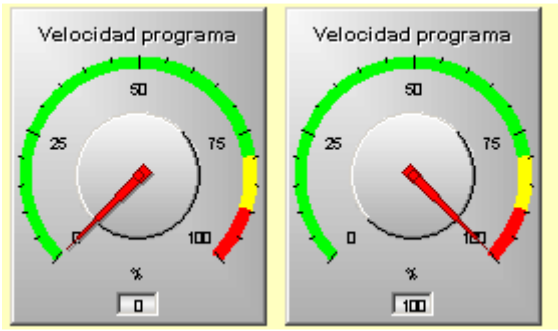
Programa Núm: 4.

Significat: El programa en execució és un despaletitzat de 1x5.

Situació: Al robot 4 (Paletitzador).

Selector analògic "Velocidad"/"Velocity": Aquest dial analògic ens permet visualitzar i modificar la velocitat general de programa en tant per cent (%). Canvis en aquest paràmetre variarà la velocitat d'execució de tot el procés que executi el robot però no sobre una instrucció específica. Els valors estaran compresos entre 0 i 100 i qualsevol altre valor generarà un error.

Si per exemple tenim una instrucció programada a una velocitat de 2m/s i una segona instrucció programada a 1m/s en aplicar una velocitat de programa del 50% els valors de velocitat es veurien afectats a 1m/s i 0,5m/s respectivament.



4. SEGURETAT

4.1 ATURADES D'EMERGÈNCIA

A través de l'endoll X11 es poden connectar dispositius d'aturada d'emergència addicionals o concatenar la instal·lació amb unitats de control superior (com ara PLC). Les senyals d'entrada/sortida i també les alimentacions externes han de garantitzar un estat segur en el cas d'una aturada d'emergència.

Si s'atura un programa en execució, es prem una aturada d'emergència o s'obre una porta de seguretat, el robot es deté. La unitat de control diferencia aquí entre els següents estats:

Aturada per rampa: El robot es deté mitjançant una rampa de frenada normal i es troba sobre la trajectòria programada. Això esdevé quan:

A) Durant el mode manual del robot es deixa anar la tecla "avançar el programa endavant" o "avançar el programa enrere".

B) Es prem la tecla d'aturada (Stop) durant el servei "Automàtic" o "Automàtic Extern" (Es tracta d'un Stop passiu).

C) Llevem la alliberació de moviment.

Aturada d'emergència amb detenció sobre la trajectòria: La unitat de control intenta detenir el robot seguint la trajectòria, per mitjà d'una rampa de frenada més aguda i un petit retard de temps. Això esdevé quan:

A) Durant el servei automàtic s'activa algun polsador d'aturada d'emergència; si la rampa d'aturada d'emergència no pot ser finalitzada, s'efectua una aturada de curtcircuit.

B) S'ha deixat anar el polsador d'home mort (en mode manual, T1 o T2); si no es possible complir amb la trajectòria, la unitat de control commuta automàticament al estat "aturada amb frenada dinàmica".

C) Durant el servei automàtic s'obre alguna porta de seguretat, o es lesiona la protecció de l'operari; si no es possible complir amb la trajectòria, la unitat de control commuta automàticament al estat "aturada amb frenada dinàmica".

D) Durant la execució d'un programa es desconnecten els accionaments; si no es possible complir amb la trajectòria, la unitat de control commuta automàticament al estat "aturada amb frenada dinàmica".

E) Durant la execució d'un programa es canvia de mode de servei; si no es possible complir amb la trajectòria, la unitat de control commuta automàticament al estat "aturada amb frenada dinàmica".

Aturada amb frenada dinàmica: El robot ja no es troba sobre la seva trajectòria. Aquest cas es presenta quan:

A) Un eix sobrepassa la seva velocitat nominal o bé, la acceleració nominal (en el mode de servei manual T1 la velocitat nominal és menor que en T2 o en els modes de servei automàtics)

B) S'ha assolit un límit de carrera software o s'ha sobrepassat una magnitud de comandament.

Frenada de curtcircuit (aturada per efecte generador): El robot ja no es troba sobre la trajectòria programada i pot haver abandonat la seva finestra de posicionament. Aquest cas esdevé quan:

A) En el mode de servei manual (T1 o T2) s'activa el polsador d'aturada d'emergència.

B) Fallada en el codificador

C) Es desconnecta la unitat de control o tenim un tall d'alimentació de tensió.

D) El cable entre DSE i RDW està obert.
Per tal de protegir els frens dels motors contra sobreescalfaments, es determinen les energies de frenada consumides i un temps de refredament depenent d'aquesta energia.

Si aquesta energia de frenada sobrepassa un valor determinat, s'efectua un enclavament dels accionaments i s'emet un missatge d'estat en la finestra de missatges. En quant que els frens dels motors s'han refrigerat suficientment, es permet la confirmació de fallada i el robot pot ser novament desplaçat.

En el cas de frenada de curtcircuit o bé, en un stop amb efecte generador, els frens de retenció de cada eix individual ja actuen durant el moviment. Si això passa freqüentment quan el robot encara s'està movent, causa un elevat desgast als frens de retenció.

Abans d'emprar la funció "aturada sobre la trajectòria en cas de dispar de protecció de l'operari", l'usuari ha de realitzar, per a cada cas en particular, un anàlisi i qualificació de riscos.

Els següents circuits elèctrics responen a la categoria 3 segons EN 954-1:

- Dispositius d'aturada d'emergència.
- Polsador d'home mort.
- Protecció de l'usuari.
- Modes de servei.
- Entrades qualificants.

Com a resum tenim una taula que ens mostra les reaccions de STOP en el sistema del robot produïts per efectes d'operacions o per reacció a controls i els missatges de falla.

Situació que origina	T1 / T2	AUT / AUT EXT
Polsar Aturada d'emergència	Frenada propera a trajectòria (STOP 0)	Frenada sobre la trajectòria (STOP 1)
Deixar anar la tecla "RUN"	Aturada per rampa (STOP 2)	-
Deixar anar polsador home mort.	Frenada propera a trajectòria (STOP 0)	-
Obrir porta de protecció	-	Frenada sobre la trajectòria (STOP 1)
Activar "Accionaments DESC"	Frenada propera a trajectòria (STOP 0)	
Canvi mode d'operació	Frenada propera a trajectòria (STOP 0)	
Fallada codificador (Unió DSE-RDW oberta)	Frenada de curtcircuit (STOP 0)	
Validació de marxa es desactiva	Aturada per rampa (STOP 2)	
Prémer tecla STOP	Aturada per rampa (STOP 2)	
Desconnectar la unitat de control del robot /Tall de tensió	Frenada de curtcircuit (STOP 0)	

4.2 TANCATS DE SEGURETAT I ZONES DE TREBALL

Els tancats de seguretat, o gàbies han de suportar accions previsibles de forces provinents del servei i de l'entorn, no han de presentar en sí mateixos un perill i també han de respectar les distàncies mínimes a la zona de perill.

La quantitat de portes de protecció en els tancats de protecció s'han de reduir a un mínim i totes elles han d'estar equipades d'una protecció del operari (connexió X11) de manera que en obrir-se una porta de protecció es dispari la funció d'aturada d'emergència (com hem vist al punt 4.1.1 d'aquesta memòria).

La entrada de protecció de l'operari serveix per al interbloqueig de dispositius seccionadors de protecció. Si a la entrada bicanal no es connecta res no es possible el mode de servei automàtic. Per als altres modes de servei (T1 i T2) la protecció de l'operari no es troba activa.

En cas de pèrdua de senyal durant el mode de servei automàtic (per exemple, la porta de protecció es oberta) els accionaments es desconecten després de 1s i el robot s'atura amb un STOP 1. Quan la senyal es presenta novament a la entrada (per exemple, la porta ha estat de nou tancada i es confirma el missatge), pot reanudar-se el mode de servei automàtic.

Un cop es tanca una porta de seguretat, el robot no pot ser arrencat immediatament en mode de servei automàtic. És necessari efectuar una validació del missatge en el panell d'operació.

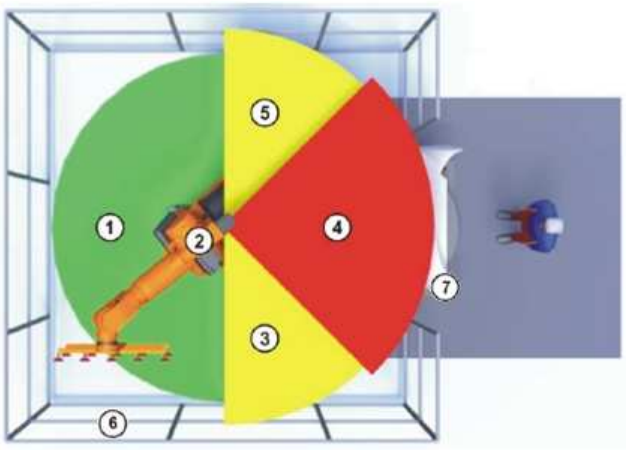
Com és lògic, no s'ha de posar mai en marxa el mode de servei d'automàtic fins a no estar segur que dins de la gàbia no hi ha ningú i que totes les portes de protecció han estat tancades. A més

a més, per a tenir més seguretat, durant un servei automàtic, les portes d'accés al tancant s'han de bloquejar mitjançant un dispositiu mecànic de seguretat.

Informacions addicionals sobre les gàbies de protecció han d'anar en concordança amb les normes i prescripcions corresponents a les característiques concretes del sistema del tancat.

Els camps de treball s'han de reduir a la mesura mínima possible necessària. Un camp de treball ha de protegir-se mitjançant els dispositius de seguretat corresponents (com més dispositius de seguretat, millor). La zona de perill la composen el camp de treball i les carreres de frenada. Han d'assegurar-se mitjançant tancats per a evitar perills a persones o danys materials.

Com podem observar a la figura Fig. 4.1 un bon exemple de seguretat seria un robot operant en un camp de treball que té aplicat una carrera de frenada i una zona protegida mitjançant hardware i software. Aquest sistema estaria dins d'un tancat amb la porta d'accés a l'interior situada (sempre que sigui possible) a la zona protegida amb hardware/software.



1 Camp de treball
2 Robot
3 Carrera de frenada
4 Zona de protecció
5 Carrera de frenada
6 Tancat de protecció
7 Porta de seguretat

Fig. 4.1 - Exemple sistema segur amb zones.

4.3 ALTRES ASPECTES DE SEGURETAT

En quant a seguretat es refereix, tot i respectar totes les observacions sobre seguretat no es pot garantir que el sistema del robot no ocasioni lesions o danys.

Està determinantment prohibit efectuar modificacions en el sistema del robot sense la autorització expressa de KUKA Robot Group. Es permet integrar components addicionals (eines, software, etc) en el sistema del robot que no pertanyin al volum de subministrament de KUKA. Si degut a la integració d'aquests components el sistema robot pateix danys, serà l'usuari qui assumeixi les responsabilitats.

La següent taula mostra en quins modes de servei els dispositius de seguretat es troben actius.

Dispositiu de seguretat	T1	T2	AUT	AUT EXT
Protecció de l'operari	-	-	Actiu	Actiu
Polsador aturada d'emergència (STOP 0)	Actiu	Actiu	-	-
Polsador aturada d'emergència (STOP 1)	-	-	Actiu	Actiu
Polsador de l'home mort	Actiu	Actiu	-	-
Velocitat reduïda	Actiu	-	-	-
Mode teclat	Actiu	Actiu	-	-
Finals de carrera software	Actiu	Actiu	Actiu	Actiu

Atenció, el robot sense dispositius de seguretat en condicions de funcionament, pot ocasionar lesions a persones o danys materials. Sempre i quan el robot estigui en servei, no està permès desmuntar o desactivar dispositius de seguretat.

Cal recordar que a la KCP es disposen de 3 polsadors de l'home mort de tres posicions (actius a T1 i T2, els modes manuals o de test). En els modes manuals el robot només es pot desplaçar quan el polsador de l'home mort es troba a la posició central. En deixar anar o polsar completament (posició de pànic) el polsador de l'home mort, els accionaments són desconnectats immediatament i el robot s'atura amb STOP 0.

Els rangs de moviment dels eixos A1, A2, A3 i A5 estan limitats mitjançant topalls mecànics amb amortidors. Els límits de carrera software serveixen a efectes de protecció màquina (mai per a protegir persones i està determinantment prohibit canviar una protecció mecànica per una software) i han de ser ajustats de manera que el robot no pugui xocar amb els topalls finals mecànics.

Si un robot impacta contra un obstacle, o bé un amortidor al límit mecànic o bé la limitació del camp de treball, pot ocasionar danys al robot. Abans de la seva posta en marxa de nou es necessari una consulta amb el Servei de KUKA. Cal reemplaçar immediatament l'amortidor afectat per un de nou. Si el robot impacta contra un amortidor a una velocitat superior a 250mm/s, cal canviar el robot o bé efectuar-se una resposta en marxa per part de KUKA.

És molt important que es tingui present en tot moment la integritat de les persones que puguin operar als voltants del robot quan aquest està en marxa. Avisar sempre que algú accedeixi dins d'una gàbia de seguretat per evitar tancar a ningú a dins i després executar el robot.

S'ha de tenir present que els accidents succeeixen, molt sovint, per no tenir presents el mal i les conseqüències que pot ocasionar un equip amb tanta força. Sovint es cau en l'error de pensar que es té la situació sota control i que no es pot donar lloc a un accident, en canvi, aquests continuen donant-se. Abans d'operar amb un robot cal tenir presents tots els riscos que comporta i totes les mesures de seguretat que cal seguir.

5. PROVES I RESULTATS

De cara a determinar quina idea pot tenir el futur usuari de l'aplicatiu s'ha fet un test del programa en HMI Studio acabat amb alguns tècnics i personal sense coneixements específics. La intenció és avaluar la utilitat, funcionalitat, distribució i practicitat de la interfície per a determinar fins a quin punt es pot millorar. Els passos de la prova han estat els següents:

1. Proposar una sèrie d'exercicis simples sense explicar-los-hi com funciona la interfície.
2. Determinar el temps en que s'assoleix els objectius dels exercicis.
3. En acabar els exercicis senzills, respondre les preguntes del formulari de forma anònima.

Per a efectuar la prova de temps s'han proposat els exercicis següents:

Ex 1. Canviar de pantalla principal a pantalla de robot 3 i canviar d'idioma
Ex 2. Visualitzar el text d'ajuda
Ex 3. Introduir al robot 3 el programa 1 i modificar velocitat de programa al 80%

Els resultats obtinguts de la prova de temps són:

	P1(s)	P2(s)	P3(s)	P4(s)	P5(s)	P6(s)	P7(s)	P8(s)	P9(s)	P10(s)	P11(s)
Ex. 1	25	34	15	43	28	32	36	39	27	29	32
Ex. 2	11	22	7	28	16	18	16	24	10	20	17
Ex. 3	38	47	31	67	39	44	56	61	36	41	46

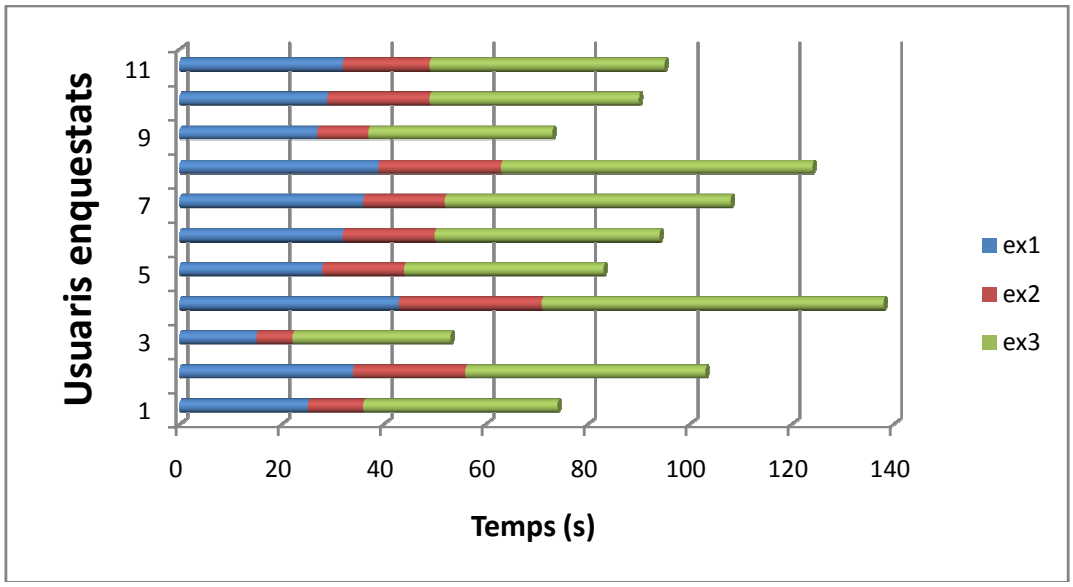


Fig. 5.1 - Diagrama temporal d'execució dels exercicis.

Com podem observar (Fig. 5.1) els onze usuaris se'n surten relativament ràpid dels exercicis proposats. Pràcticament tothom resol tots els exercicis en menys de 2 minuts, la qual cosa implica que es pot operar de manera efectiva sobre la interfície tot i no conèixer-la prèviament.

A continuació trobarem el test que s'ha passat als usuaris:

TEST ANÒMIM

NIVELL DE SATISFACCIÓ DE L'USUARI AMB LA INTERFÍCIE HMI

Marqui la casella corresponent segons el nivell de satisfacció en cada pregunta formulada.
Per a qualsevol dubte sobre les preguntes i les respostes, consultar al projectista.

1. Creus que la interfície resulta útil i funcional?
2. Trobes que el text d'ajuda aporta gaire informació?
3. Penses que la distribució dels diferents elements és correcte?
4. És fàcil la navegació per pantalles?
5. Has reconegut fàcilment on estaves situat en el mapa de navegació?
6. Es podria millorar la interfície?
7. Estaria disposat a plantejar-se la possibilitat d'adquirir el producte?

Observacions addicionals / Comentaris:

Moltes gràcies per la seva col·laboració

Les respostes es classifiquen de la manera següent

Respostes	☹☹	☹	😐	☺	☺☺
Pregunta 1	0	0	1	4	6
Pregunta 2	0	0	0	6	5
Pregunta 3	0	0	1	3	7
Pregunta 4	0	0	0	2	9
Pregunta 5	0	0	0	6	5
Pregunta 6	1	1	5	3	1
Pregunta 7	0	0	2	7	2

Gràficament (Fig. 5.2) es veu de seguida que els colors majoritaris són el blau més clar i el morat, de manera que els usuaris estan d'acord o molt d'acord amb les preguntes formulades a nivell general (exeptuant en la pregunta 6 en que hi ha una mica més de varietat en les respostes).

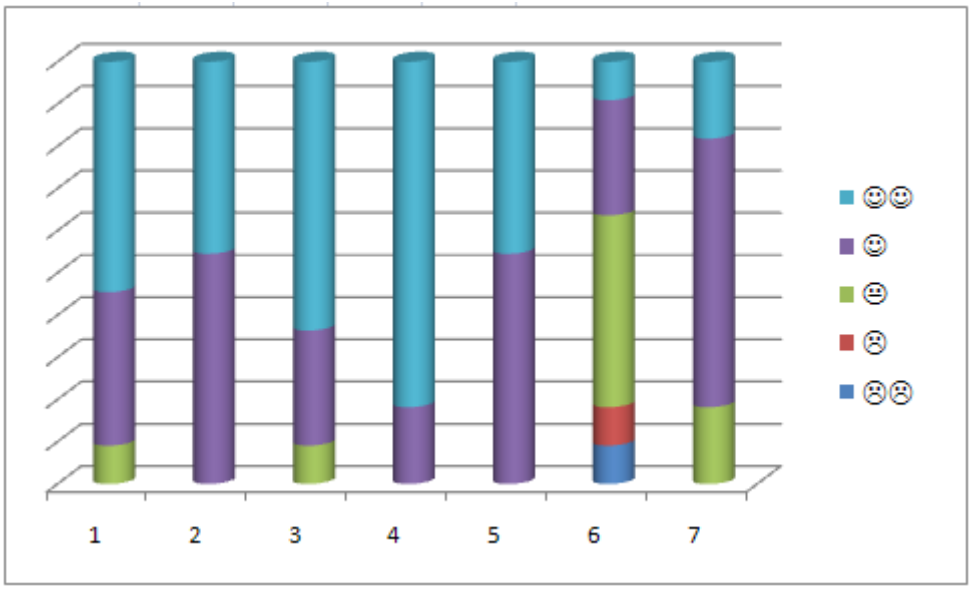


Fig. 5.2 – Respostes dels usuaris de la interfície HMI

Podem deduir doncs que la interfície és funcional, útil, pràctica i fàcil de fer anar.

En quant a la part de funcionament, s'han realitzat proves a la sala de formació amb els robots en mode automàtic extern funcionant correctament. Els leds indicatius de les sortides es posen en marxa de la forma prevista.

6. CONCLUSIONS

Aquest és un treball més complex del que aparenta donat que cal realitzar la major part del projecte a les instal·lacions de KUKA. Això ha fet que es perdi molt de temps, ja que, totes les proves de funcionament calia fer-les en horari d'oficina.

També he hagut de fer cursos d'aprenentatge sobre el funcionament dels robots, la seva programació, etc. De cara a estar qualificada per a treballar a les cel·les a aquest nivell. La documentació que he hagut d'emprar ha estat, sovint, extensa i poc clarificadora.

Alhora m'he anat trobant amb múltiples contratemps que ens han dificultat una mica algunes tasques aparentment més ràpides (com ara obres a la sala de formació). De manera que el temps s'ha recuperat realment com s'ha pogut fent un grandíssim esforç per part de tots.

Ha calgut fer moltes proves i canviar coses fins a arribar a un resultat adequat al gust de tots i que respongui les necessitats inicials. Estic orgullosa i contenta del resultat d'aquest projecte, que serà la base que ens obri les portes a una nova forma de treballar amb els robots en que l'operari no necessiti tants coneixements a nivell de programació i control de robots. Responem a la demanda, doncs, sense emprar cap software extra o d'altres institucions.

He après moltes coses sobre els robots KUKA i la manera en que treballen i he pogut posar en pràctica alguns dels coneixements adquirits a la universitat en el període d'aprenentatge.

Queda la possibilitat de afegir nous canvis, millores, inserció de mòduls nous, modificacions en el nombre de robots, canvis en els programes a executar, etc. S'ha fet molta feina, però podria no acabar-se mai.

Caldria tenir en compte les especificacions dels clients alhora de pensar en la possibilitat de crear aquests tipus d'aplicacions de manera comercial.

7. BIBLIOGRAFIA I REFERÈNCIES

[1] KUKA Roboter GmbH, KUKA College, "Documentación del robot, OPERADOR", *Software KUKA V5.x.*, Vilanova i la Geltrú, 2006.

[2] KUKA Roboter GmbH, KUKA College, "Documentación del robot, SEGURIDADES", *Para Software KUKA V5.x.*, Vilanova i la Geltrú, 2006.

[3] KUKA Roboter GmbH, KUKA College, "Programación experto de robots, PROGRAMADOR", *Para Software KUKA V5.x.*, Vilanova i la Geltrú, 2006.

[4] KUKA Roboter GmbH, "Training Documentation, Soft-PLC", *Specialist Seminar Software PLC* , Edition 11.03.99.

[5] KUKA Roboter GmbH, "Seminar workbook of HMI - Studio", *Software KRC* , June 2004.

[6] KUKA Roboter GmbH, "OPC Server Release 2.0", *Software KR C2* , version 01 June 2004.

[7] KUKA Roboter GmbH, "KR C2 edition05, Especificación", *Controller* , version 1.2 Edición 26.07.2007.

8. ANNEX

8.1 CODIS DE PROGRAMES

8.1.1 Robot KR 16 (A) – Pick&Place

Programa que es crida en Automàtic Extern Cell.src:

```
DEF Cell( )
  ;EXT KRA_PROG1 ( )
  ;EXT KRA_PROG2 ( )
INIT
BASISTECH INI
CHECK HOME
PTP HOME  Vel= 100 % DEFAULT
AUTOEXT INI
  LOOP
    POO (#EXT_PGNO,#PGNO_GET,DMY[],0 )
    SWITCH  PGNO ; Seleccio del programa segons el numero

    CASE 1
      POO (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Resset Progr.No.-Request
      KRA_PROG1 ( ) ; Realitzar el programa 1

    CASE 2
      POO (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Resset Progr.No.-Request
      KRA_PROG2 ( ) ; Realitzar el programa 2

    DEFAULT
      POO (#EXT_PGNO,#PGNO_FAULT,DMY[],0 ); Problema amb el Progr.No.
    ENDSWITCH
  ENDLOOP
END
```

Primer programa del Robot KR 16 (A) – Pick & Place:

```
;&COMMENT Prog 1 per KR16 A
DEF KRA_Prog1( )
INI

;Resetejem sortides (per si de cas):
FOR I = 1 TO 16
   $OUT[I] = FALSE
ENDFOR

PTP HOME
Agafar_P()
PTP P1  Vel=100 % PDAT1 Tool[1] Base[1]
;Moviment a la posicio 1 on deixar peca:
LIN P5  Vel=2 m/s CPDAT4 Tool[1] Base[1]
;deixem peca:
SET GRIPPER  State= OPEN  GDAT2
LIN P1  Vel=2 m/s CPDAT5 Tool[1] Base[1]
Agafar_P()
PTP P1  Vel=100 % PDAT1 Tool[1] Base[1]
;Moviment a la posicio 2 on deixar peca:
LIN P6  Vel=2 m/s CPDAT6 Tool[1] Base[1]
;deixem peca:
SET GRIPPER  State= OPEN  GDAT2
```

```
LIN P1  Vel=2 m/s CPDAT5 Tool[1] Base[1]
Agafar_P()
PTP P1  Vel=100 % PDAT1 Tool[1] Base[1]
;Moviment a la posicio 3 on deixar peca:
LIN P7  Vel=2 m/s CPDAT7 Tool[1] Base[1]
;deixem peca:
SET GRIPPER  State= OPEN  GDAT2
LIN P1  Vel=2 m/s CPDAT5 Tool[1] Base[1]
Agafar_P()
PTP P1  Vel=100 % PDAT1 Tool[1] Base[1]
;Moviment a la posicio 4 on deixar peca:
LIN P8  Vel=2 m/s CPDAT8 Tool[1] Base[1]
;deixem peca:
SET GRIPPER  State= OPEN  GDAT2
; Ja s'han deixat les 4 peces.

$OUT[0] = TRUE ; Indiquem que el programa ha finalitzat el paletitzat
PTP HOME

END

DEF Agafar_P( )
;Ens assegurem que la pinca es oberta:
SET GRIPPER  State= OPEN  GDAT2
PTP P2 CONT Vel=100 % PDAT2 Tool[1] Base[1]
PTP P3 CONT Vel=100 % PDAT3 Tool[1] Base[1]
LIN P4  Vel=1 m/s CPDAT2 Tool[1] Base[1]
SET GRIPPER  State= CLOSE  GDAT2
;Ja tenim la peca agafada i hem de sortir del sortidor de peces
LIN P3  Vel=1 m/s CPDAT3 Tool[1] Base[1]
PTP P2 CONT Vel=100 % PDAT4 Tool[1] Base[1]

END
```

Segon programa del Robot KR 16 (A) – Pick & Place:

```
DEF KRA_Prog2( )
INI

;Resetejem sortides (per si de cas):
FOR I = 1 TO 16
   $OUT[I] = FALSE
ENDFOR

PTP HOME
Agafar_P()
PTP P1  Vel=100 % PDAT1 Tool[1] Base[1]
;Moviment a la posicio 1 on deixar peca:
LIN P5  Vel=2 m/s CPDAT4 Tool[1] Base[1]
;deixem peca:
SET GRIPPER  State= OPEN  GDAT2
;FOLD LIN P1  Vel=2 m/s CPDAT5 Tool[1] Base[1]
Agafar_P()
PTP P1  Vel=100 % PDAT1 Tool[1] Base[1]
;Moviment a la posicio 2 on deixar peca:
LIN P6  Vel=2 m/s CPDAT6 Tool[1] Base[1]
;deixem peca:
SET GRIPPER  State= OPEN  GDAT2
LIN P1  Vel=2 m/s CPDAT5 Tool[1] Base[1]
```

```
24 Agafar_P()
25 PTP P1  Vel=100 % PDAT1 Tool[1] Base[1]
26 ;Moviment a la posicio 3 on deixar peca:
27 LIN P7  Vel=2 m/s CPDAT7 Tool[1] Base[1]
28 ;deixem peca:
29 SET GRIPPER  State= OPEN  GDAT2
30 ; Ja s'han deixat les 3 peces.
31 $OUT[0] = TRUE ; Indiquem que el programa ha finalitzat el paletitzat
32 PTP HOME
33
34 END
35
36 DEF Agafar_P( )
37 ;Ens assegurem que la pinca es oberta:
38 SET GRIPPER  State= OPEN  GDAT2
39 PTP P2 CONT Vel=100 % PDAT2 Tool[1] Base[1]
40 PTP P3 CONT Vel=100 % PDAT3 Tool[1] Base[1]
41 LIN P4  Vel=1 m/s CPDAT2 Tool[1] Base[1]
42 SET GRIPPER  State= CLOSE  GDAT2
43 ;Ja tenim la peca agafada i hem de sortir del sortidor de peces
44 LIN P3  Vel=1 m/s CPDAT3 Tool[1] Base[1]
45 PTP P2 CONT Vel=100 % PDAT4 Tool[1] Base[1]
46
47 END
48
```

8.1.2 Robot KR 16 (B) – Manipulador A

Programa que es crida en Automàtic Extern Cell.src:

```
 1 DEF Cell( )
 2   ;EXT KRB_PROG1 ( )
 3   ;EXT KRB_PROG2 ( )
 4   ;EXT KRB_PROG3 ( )
 5
 6 INIT
 7 BASISTECH INI
 8 CHECK HOME
 9 PTP HOME  Vel= 100 % DEFAULT
10 AUTOEXT INI
11   LOOP
12     POO (#EXT_PGNO,#PGNO_GET,DMY[],0 )
13     SWITCH  PGNO ; Seleccio del programa segons el numero
14
15     CASE 1
16       POO (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Resset Progr.No.-Request
17       KRB_PROG1 ( ) ; Realitzar el programa 1
18
19     CASE 2
20       POO (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Resset Progr.No.-Request
21       KRB_PROG2 ( ) ; Realitzar el programa 2
22
23     CASE 3
24       POO (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Resset Progr.No.-Request
25       KRB_PROG3 ( ) ; Realitzar el programa 3
26
27     DEFAULT
28       POO (#EXT_PGNO,#PGNO_FAULT,DMY[],0 ); Problema amb el Progr.No.
29     ENDSWITCH
30   ENDLOOP
31 END
```

Primer programa del Robot KR 16 (B) – Manipulador A:

```
DEF KRB_Prog1( )
PTP HOME
; No hi ha necessitat de definir cap variable.

;Resetejem sortides (per si de cas):
FOR I = 1 TO 16
   $OUT[I] = FALSE
ENDFOR

; Posicio coneguda de seguretat a l'espai
PTP P1  Vel=100 % PDAT1 Tool[0] Base[0]
;Posicio sobre del punt Start.
LIN P2 CONT Vel=0.2 m/s CPDAT1 Tool[1] Base[1]
;Aqui comen?aria el proces de tallar la pessa, accionem sortida 1 (Talla):
$OUT[1] = TRUE
;Fem el recorregut tallant
CIRC P3 P4 CONT Vel=2 m/s CPDAT2 Tool[1] Base[1]
CIRC P5 P6 CONT Vel=2 m/s CPDAT3 Tool[1] Base[1]
LIN P7 CONT Vel=2 m/s CPDAT4 Tool[1] Base[1]
CIRC P8 P9 CONT Vel=2 m/s CPDAT7 Tool[1] Base[1]
LIN P10 CONT Vel=2 m/s CPDAT8 Tool[1] Base[1]
CIRC P11 P12 CONT Vel=2 m/s CPDAT9 Tool[1] Base[1]
LIN P13 CONT Vel=2 m/s CPDAT10 Tool[1] Base[1]
CIRC P14 P15 CONT Vel=2 m/s CPDAT11 Tool[1] Base[1]
CIRC P16 P17 CONT Vel=2 m/s CPDAT12 Tool[1] Base[1]
CIRC P18 P19 CONT Vel=2 m/s CPDAT13 Tool[1] Base[1]
LIN P20 CONT Vel=2 m/s CPDAT14 Tool[1] Base[1]
CIRC P21 P22 CONT Vel=2 m/s CPDAT15 Tool[1] Base[1]
LIN P23  Vel=2 m/s CPDAT16 Tool[1] Base[1]
$OUT[1] = FALSE ; En arribar aqui deixem de tallar.

;Moviment a posicio coneguda de seguretat a l'espai
PTP P1  Vel=100 % PDAT2 Tool[0] Base[0]
;Posicio sobre 1 punt del cercle a tallar en mig de la pessa:
LIN P24  Vel=2 m/s CPDAT20 Tool[1] Base[1]

; Aqui comensaria el proces de tallar:
$OUT[1] = TRUE
;Fem el recorregut tallant
CIRC P25 P26  Vel=2 m/s CPDAT23 Tool[1] Base[1]
CIRC P27 P25  Vel=2 m/s CPDAT24 Tool[1] Base[1]
$OUT[1] = FALSE ; En arribar aqui deixem de tallar.
; Pessa acabada, anar home i acabar programa.

PTP HOME
END
```

Segon programa del Robot KR 16 (B) – Manipulador A:

```
DEF KRB_Prog2( )
INI

; No hi ha necessitat de definir cap variable.
; Resetejem sortides (per si de cas):
FOR I = 1 TO 16
   $OUT[I] = FALSE
ENDFOR
```

```
 9
10 PTP HOME
11
12 ; SERIGRAFIAR PRIMERA K:
13 ; P1 situat a 200mm per sobre de la xapa.
14 PTP P1  Vel=100 % PDAT1 Tool[1] Base[1]
15 LIN_REL {Z -200} ;Moviment relatiu en z de la base.
16 OUT[2] = TRUE ; Posem en marxa la eina.
17 LIN P2  Vel=2 m/s CPDAT3 Tool[0] Base[0]
18 OUT[2] = FALSE ; Aturem la eina
19 LIN_REL {z +200} ; Moviment relatiu en z de la base.
20 ;Punt P3 situat a 200mm per sobre del segon punt d'inici
21 LIN P3  Vel=2 m/s CPDAT4 Tool[1] Base[1]
22 LIN_REL {Z -200} ;Moviment relatiu en z de la base.
23 OUT[2] = TRUE ; Posem en marxa la eina.
24 CIRC P4 P5  Vel=2 m/s CPDAT5 Tool[1] Base[1]
25 OUT[2] = FALSE ; Aturem la eina
26 LIN_REL {z +200} ; Moviment relatiu en z de la base.
27 ;De 200mm sobre el punt P5 al P4 tocant xapa:
28 LIN P4  Vel=1 m/s CPDAT6 Tool[1] Base[1]
29 OUT[2] = TRUE ; Posem en marxa la eina.
30 LIN P6  Vel=2 m/s CPDAT7 Tool[1] Base[1]
31 OUT[2] = FALSE ; Aturem la eina
32 LIN_REL {z +200} ; Moviment relatiu en z de la base.
33
34 ; SERIGRAFIAR LA U:
35 ; P7 situat a 200mm per sobre de la xapa.
36 PTP P7  Vel=100 % PDAT2 Tool[1] Base[1]
37 LIN_REL {Z -200} ;Moviment relatiu en z de la base.
38 OUT[2] = TRUE ; Posem en marxa la eina.
39 LIN P8  Vel=2 m/s CPDAT8 Tool[1] Base[1]
40 CIRC P9 P10  Vel=2 m/s CPDAT9 Tool[1] Base[1]
41 LIN P11  Vel=2 m/s CPDAT10 Tool[1] Base[1]
42 OUT[2] = FALSE ; Aturem la eina
43 LIN_REL {z +200} ; Moviment relatiu en z de la base.
44
45 ; SERIGRAFIAR SEGONA K:
46 ; P12 situat a 200mm per sobre de la xapa.
47 PTP P12  Vel=100 % PDAT3 Tool[1] Base[1]
48 LIN_REL {Z -200} ;Moviment relatiu en z de la base.
49 OUT[2] = TRUE ; Posem en marxa la eina.
50 LIN P13  Vel=2 m/s CPDAT11 Tool[1] Base[1]
51 OUT[2] = FALSE ; Aturem la eina
52 LIN_REL {z +200} ; Moviment relatiu en z de la base.
53 ;Punt P14 situat a 200mm per sobre del següent punt
54 LIN P14  Vel=2 m/s CPDAT12 Tool[1] Base[1]
55 LIN_REL {Z -200} ;Moviment relatiu en z de la base.
56 OUT[2] = TRUE ; Posem en marxa la eina.
57 CIRC P15 P16  Vel=2 m/s CPDAT13 Tool[1] Base[1]
58 OUT[2] = FALSE ; Aturem la eina
59 LIN_REL {z +200} ; Moviment relatiu en z de la base.
60 ;De 200mm sobre el punt P16 al P15 tocant xapa:
61 LIN P15  Vel=2 m/s CPDAT14 Tool[1] Base[1]
62 OUT[2] = TRUE ; Posem en marxa la eina.
63 LIN P17  Vel=2 m/s CPDAT15 Tool[1] Base[1]
64 OUT[2] = FALSE ; Aturem la eina
65 LIN_REL {z +200} ; Moviment relatiu en z de la base.
```

```
66
67 ; SERIGRAFIAR LA A:
68 ; P18 situat a 200mm per sobre de la xapa.
69 PTP P18  Vel=100 % PDAT4 Tool[1] Base[1]
70 LIN_REL {Z -200} ;Moviment relatiu en z de la base.
71 OUT[2] = TRUE ; Posem en marxa la eina.
72 LIN P19  Vel=2 m/s CPDAT16 Tool[1] Base[1]
73 CIRC P20 P21  Vel=2 m/s CPDAT17 Tool[1] Base[1]
74 LIN P22  Vel=2 m/s CPDAT18 Tool[1] Base[1]
75 OUT[2] = FALSE ; Aturem la eina
76 LIN_REL {z +200} ; Moviment relatiu en z de la base.
77 ;De 200mm sobre el punt P22 al P23 tocant xapa:
78 LIN P23  Vel=2 m/s CPDAT19 Tool[1] Base[1]
79 OUT[2] = TRUE ; Posem en marxa la eina.
80 LIN P24  Vel=2 m/s CPDAT20 Tool[1] Base[1]
81 OUT[2] = FALSE ; Aturem la eina
82
83 PTP HOME
84
85 END
```

Tercer programa del Robot KR 16 (B) – Manipulador A:

```
 1 DEF KRB_Prog3( )
 2 INI
 3
 4 ; No hi ha necessitat de definir cap variable.
 5 ; Resetejem sortides (per si de cas):
 6 FOR I = 1 TO 16
 7    $OUT[I] = FALSE
 8 ENDFOR
 9
10 PTP HOME
11 ; Pl punt d'inici, farem la trajectòria completa.
12 $OUT[1] = TRUE ; Posem en marxa eina de tall
13 LIN P1  Vel=1 m/s CPDAT1 Tool[1] Base[1]
14 LIN P2  Vel=2 m/s CPDAT2 Tool[1] Base[1]
15 LIN P3  Vel=2 m/s CPDAT3 Tool[1] Base[1]
16 LIN P4  Vel=2 m/s CPDAT4 Tool[1] Base[1]
17 LIN P5  Vel=2 m/s CPDAT5 Tool[1] Base[1]
18 CIRC P6 P7  Vel=2 m/s CPDAT6 Tool[1] Base[1]
19 CIRC P8 P9  Vel=2 m/s CPDAT7 Tool[1] Base[1]
20 LIN P10  Vel=2 m/s CPDAT8 Tool[1] Base[1]
21 CIRC P11 P12  Vel=2 m/s CPDAT9 Tool[1] Base[1]
22 CIRC P13 P14  Vel=2 m/s CPDAT10 Tool[1] Base[1]
23 CIRC P15 P16  Vel=2 m/s CPDAT11 Tool[1] Base[1]
24 CIRC P17 P18  Vel=2 m/s CPDAT12 Tool[1] Base[1]
25 CIRC P19 P20  Vel=2 m/s CPDAT13 Tool[1] Base[1]
26 CIRC P21 P1  Vel=2 m/s CPDAT14 Tool[1] Base[1]
27 ; Acabem tornant al punt 1.
28 $OUT[1] = FALSE ; Aturem eina
29 $OUT[3] = TRUE ; Indiquem que el programa ha finalitzat
30
31 PTP HOME
32
33 END
```

8.1.3 Robot KR 16 (C) – Manipulador B

Programa que es crida en Automàtic Extern Cell.src:

```
DEF Cell( )
  ;EXT KRC_PROG1 ( )
  ;EXT KRC_PROG2 ( )

INIT
BASISTECH INI
CHECK HOME
PTP HOME  Vel= 100 % DEFAULT
AUTOEXT INI
  LOOP
    POO (#EXT_PGNO,#PGNO_GET,DMY[],0 )
    SWITCH  PGNO ; Seleccio del programa segons el numero

    CASE 1
      POO (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Resset Progr.No.-Request
      KRC_PROG1 ( ) ; Realitzar el programa 1

    CASE 2
      POO (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Resset Progr.No.-Request
      KRC_PROG2 ( ) ; Realitzar el programa 2

    DEFAULT
      POO (#EXT_PGNO,#PGNO_FAULT,DMY[],0 ); Problema amb el Progr.No.
    ENDSWITCH
  ENDLOOP
END
```

Primer programa del Robot KR 16 (C) – Manipulador B:

```
DEF KRC_Prog1( )
INI

; No hi ha necessitat de definir cap variable.
; Resetejem sortides (per si de cas):
FOR I = 1 TO 16
   $OUT[I] = FALSE
ENDFOR

PTP HOME

PTP P1  Vel=100 % PDAT1 Tool[0] Base[0]
; punt d'inici
; Execucio de la ruta de la sabata esquerra:
; En mig de la recta es posa en marxa l'adhesiu!!
LIN P2 CONT Vel=1 m/s CPDAT1 Tool[1] Base[1]
SYN OUT 3 ''  State = TRUE  PATH = 10mm  Delay= 3 ms
LIN CONT P3  Vel=1 m/s CPDAT2 Tool[1] Base[1]
CIRC CONT P4 P5  Vel=1 m/s CPDAT3 Tool[1] Base[1]
CIRC CONT P6 P7  Vel=1 m/s CPDAT4 Tool[1] Base[1]
CIRC CONT P8 P9  Vel=1 m/s CPDAT5 Tool[1] Base[1]
CIRC P10 P11  Vel=1 m/s CPDAT6 Tool[1] Base[1]
; En acabar la ruta aturem l'adhesiu:
$OUT[3] = FALSE
```

```
25
26 ;Execució de la ruta de la sabata dreta:
27
28 PTP P12  Vel=100 % PDAT2 Tool[1] Base[1]
29 ; En mig de la recta es posa en marxa l'adhesiu!!
30 LIN P13 CONT Vel=1 m/s CPDAT7 Tool[1] Base[1]
31 SYN OUT 3 ''  State= TRUE  PATH = 10mm  Delay= 3 ms
32 LIN P14 CONT Vel=1 m/s CPDAT8 Tool[1] Base[1]
33 CIRC P15 P16 CONT Vel=1 m/s CPDAT9 Tool[1] Base[1]
34 CIRC P17 P18 CONT Vel=1 m/s CPDAT11 Tool[1] Base[1]
35 CIRC P19 P20 CONT Vel=1 m/s CPDAT12 Tool[1] Base[1]
36 CIRC P21 P22 CONT Vel=1 m/s CPDAT13 Tool[1] Base[1]
37 ; En acabar la ruta aturem l'adhesiu:
38 $OUT[3] = FALSE
39
40 PTP HOME
41
42 END
```

Segon programa del Robot KR 16 (C) – Manipulador B:

```
 1 DEF KRC_Prog2( )
 2 INI
 3
 4 ; No hi ha necessitat de definir cap variable.
 5 ; Resetejem sortides (per si de cas):
 6 FOR I = 1 TO 16
 7    $OUT[I] = FALSE
 8 ENDFOR
 9
10 PTP HOME
11
12 PTP P1  Vel=100 % PDAT1 Tool[0] Base[0]
13 ; punt d'inici
14 ; Execucio de la ruta de la sabata esquerra:
15 ; En mig de la recta es posa en marxa l'adhesiu!!
16 LIN P2 CONT Vel=1 m/s CPDAT1 Tool[1] Base[1]
17 SYN OUT 3 ''  State = TRUE  PATH = 9mm  Delay= 3 ms
18 LIN CONT P3  Vel=1 m/s CPDAT2 Tool[1] Base[1]
19 SYN OUT 3 ''  State= TRUE  PATH = 16mm  Delay= 4 ms
20 CIRC CONT P4 P5  Vel=1 m/s CPDAT3 Tool[1] Base[1]
21 ; En acabar la ruta aturem l'adhesiu:
22 $OUT[3] = FALSE
23
24 ;Execucio de la ruta de la sabata dreta:
25 PTP P6  Vel=100 % PDAT2 Tool[1] Base[1]
26 ; En mig de la recta es posa en marxa l'adhesiu!!
27 LIN P7 CONT Vel=1 m/s CPDAT4 Tool[1] Base[1]
28 SYN OUT 3 ''  State= TRUE  PATH = 16mm  Delay= 3 ms
29 LIN P8 CONT Vel=1 m/s CPDAT5 Tool[1] Base[1]
30 SYN OUT 3 ''  State= TRUE  PATH = 10mm  Delay= 4 ms
31 CIRC P9 P10 CONT Vel=1 m/s CPDAT6 Tool[1] Base[1]
32 ; En acabar la ruta aturem l'adhesiu:
33 $OUT[3] = FALSE
34
35 PTP HOME
36
37 END
```

8.1.4 Robot KR 16 (D) – Pal·letitzador

Programa que es crida en Automàtic Extern Cell.src:

```
DEF Cell( )
  ;EXT KRD_PROG1 ( )
  ;EXT KRD_PROG2 ( )
  ;EXT KRD_PROG3 ( )
  ;EXT KRD_PROG4 ( )

INIT
BASISTECH INI
CHECK HOME
PTP HOME  Vel= 100 % DEFAULT
AUTOEXT INI
  LOOP
    POO (#EXT_PGNO,#PGNO_GET,DMY[],0 )
    SWITCH  PGNO ; Seleccio del programa segons el numero

    CASE 1
      POO (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Resset Progr.No.-Request
      KRD_PROG1 ( ) ; Realitzar el programa 1

    CASE 2
      POO (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Resset Progr.No.-Request
      KRD_PROG2 ( ) ; Realitzar el programa 2

    CASE 3
      POO (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Resset Progr.No.-Request
      KRD_PROG3 ( ) ; Realitzar el programa 3

    CASE 4
      POO (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Resset Progr.No.-Request
      KRD_PROG4 ( ) ; Realitzar el programa 4

    DEFAULT
      POO (#EXT_PGNO,#PGNO_FAULT,DMY[],0 ); Problema amb el Progr.No.
    ENDSWITCH
  ENDLOOP
END
```

Primer programa del Robot KR 16 (D) Paletitzador:

```
DEF KRD_Prog1( )
INI
; --------------- Seccio declarativa ---------------
INT PAL, I
POS Pseguretat {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0,S 0,T 0}
; --------------- Seccio d'instruccions ---------------
PTP HOME
;Resetejem sortides (per si de cas):
FOR I = 1 TO 16
   $OUT[I] = FALSE
ENDFOR

```

```
; Palet de 3x3= 9 peces:
FOR PAL=1 TO 9 STEP 1
   ;Punt respectiu a base i tool nullframe:
   PTP P1  Vel=100 % PDAT1 Tool[0] Base[0]
   ; En aquest punt respecte la base(1) i la eina (1):
   LIN P2  Vel=2 m/s CPDAT1 Tool[1] Base[1]
   ; Agafem peca i acte seguit calculem el punt on deixar-la.
   Agafar_p()
   IF PAL == 1 THEN
         LIN P3  Vel=2 m/s CPDAT3 Tool[1] Base[1]
         ; Aquest es el primer punt del palet; el guardem.
         Pseguretat = $POS_INT
         ;Obrim pinca per deixar la primera peca del palet:
         SET GRIPPER  State= OPEN  GDAT6

   ELSE ; PAL no es 1:
      IF PAL == 2 OR 3 OR 5 OR 6 OR 8 OR 9 THEN
         XPPalet.x = XPPalet.x + 60 ; Incrementem 60mm en X
         ;Interessa que PPalet es situi 200mm sobre el palet.
         XPPalet.Z = XPSeguretat.z + 200
         ; Moviment lineal al punt PPalet:
         LIN PPalet  Vel=2 m/s CPDAT4 Tool[1] Base[1]
         Deixar()
      ELSE ;PAL val o 4 o 7:
         XPPalet.x = XPSeguretat.x
         XPPalet.y = XPPalet.y + 60
         LIN PPalet  Vel=2 m/s CPDAT5 Tool[1] Base[1]
         Deixar()
      ENDIF
   ENDIF
ENDFOR

$OUT[0] = TRUE ; Indiquem que el programa ha finalitzat el paletitzat
PTP HOME

END
```

```
DEF Agafar_p( )
; Ens assegurem que la pinca esta oberta:
SET GRIPPER  State= OPEN GDAT2
PTP P4 CONT Vel=100 % PDAT4 Tool[1] Base[1]
PTP P5 CONT Vel=100 % PDAT5 Tool[1] Base[1]
;P6 amb orientacio constant!
LIN P6  Vel=1 m/s CPDAT6 Tool[1] Base[1]
;Tanquem pinca:
SET GRIPPER  State= CLOSE  GDAT3
;Ara tenim peca agafada.
LIN P5  Vel= 1 m/s CPDAT7
PTP P4  Vel= 100 % PDAT6

END
```

```
64
65 DEF Deixar( )
66
67 ; Moviment a la posicio PPalet
68 LIN PPalet  Vel=1 m/s CPDAT8 Tool[1] Base[1]
69 LIN_REL {Z -200} ; moviment relatiu a la Z de la base
70 SET GRIPPER  State= OPEN  GDAT5
71 LIN_REL {Z +200} ; moviment relatiu a la Z de la base
72
73 END
74
```

Segon programa del Robot KR 16 (D) Paletitzador:

```
 1 DEF KRD_Prog2( )
 2 INI
 3 ; --------------- Seccio declarativa ---------------
 4 INT PAL, I
 5 POS Pseguretat {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0,S 0,T 0}
 6 ; --------------- Seccio d'instruccions ---------------
 7 PTP HOME
 8 ;Resetejem sortides (per si de cas):
 9 FOR I = 1 TO 16
10    $OUT[I] = FALSE
11 ENDFOR
12
13 ; Palet de 3x3= 9 peces:
14 FOR PAL=1 TO 9 STEP 1
15    ;Punt respectiu a base i tool nullframe:
16    PTP P1  Vel=100 % PDAT1 Tool[0] Base[0]
17    ; En aquest punt respecte la base(1) i la eina (1):
18    LIN P2  Vel=2 m/s CPDAT1 Tool[1] Base[1]
19    ; Obrim pinca per no xocar amb peca al palet
20   SET GRIPPER  State= OPEN  GDAT6
21
22    ; Calcul de la peca a agafar:
23    IF PAL == 1 THEN
24          LIN P3  Vel=2 m/s CPDAT3 Tool[1] Base[1]
25          ; Aquest es el primer punt del palet; el guardem.
26          Pseguretat = $POS_INT
27          ;Tancar pinca per agafar la primera peca del palet:
28          SET GRIPPER  State= CLOSE  GDAT6
29
30    ELSE ; PAL no es 1:
31       IF PAL == 2 OR 3 OR 5 OR 6 OR 8 OR 9 THEN
32          XPPalet.x = XPPalet.x + 60 ; Incrementem 60mm en X
33          ;Interessa que PPalet es situi 200mm sobre el palet.
34          XPPalet.Z = XPSeguretat.z + 200
35          ; Moviment lineal al punt PPalet:
36          LIN PPalet  Vel=2 m/s CPDAT4 Tool[1] Base[1]
37          Agafar()
38       ELSE ;PAL val o 4 o 7:
39          XPPalet.x = XPSeguretat.x
40          XPPalet.y = XPPalet.y + 60 ; Incrementem 60mm en Y
41          LIN PPalet  Vel=2 m/s CPDAT5 Tool[1] Base[1]
42          Agafar()
43       ENDIF
44       ;Un cop agafada la peca la podem deixar a la pila:
45       Deixar()
46    ENDIF
47 ENDFOR
```

```
48
49 $OUT[0] = TRUE ; Indiquem que el programa ha finalitzat el paletitzat
50 PTP HOME
51
52 END
53
54 DEF Agafar( )
55 ; Ens assegurem que la pinca esta oberta (TRUE):
56 SET GRIPPER  State= OPEN GDAT2
57 ; Moviment a la posició PPalet
58 LIN PPalet  Vel=1 m/s CPDAT8 Tool[1] Base[1]
59 LIN_REL {Z -200}
60 SET GRIPPER  State= CLOSE  GDAT5
61 LIN_REL {Z +200}
62
63 END
64
65 DEF Deixar( )
66
67 PTP P4 CONT Vel=100 % PDAT4 Tool[1] Base[1]
68 PTP P5 CONT Vel=100 % PDAT5 Tool[1] Base[1]
69 ;Obrim pinca:
70 SET GRIPPER  State= OPEN  GDAT3
71 ;Ara ja hem deixat la peça a la pila.
72 PTP P4  Vel= 100 % PDAT6
73 PTP HOME
74
75 END
```

Tercer programa del Robot KR 16 (D) Paletitzador:

```
 1 DEF KRD_Prog3( )
 2 INI
 3 ; --------------- Seccio declarativa ---------------
 4 INT PAL, I
 5 POS Pseguretat {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0,S 0,T 0}
 6 ; --------------- Seccio d'instruccions ---------------
 7 PTP HOME
 8 ;Resetejem sortides (per si de cas):
 9 FOR I = 1 TO 16
10    $OUT[I] = FALSE
11 ENDFOR
12
13 ; Linia de 5 peces = 1x5:
14 FOR PAL=1 TO 5 STEP 1
15    ;Punt respectiu a base i tool nullframe:
16    PTP P1  Vel=100 % PDAT1 Tool[0] Base[0]
17    ; En aquest punt respecte la base(1) i la eina (1):
18    LIN P2  Vel=2 m/s CPDAT1 Tool[1] Base[1]
19    ; Agafem peca i acte seguit calculem el punt on deixar-la.
20    Agafar_p()
21    IF PAL == 1 THEN
22          LIN P3  Vel=2 m/s CPDAT3 Tool[1] Base[1]
23          ; Aquest es el primer punt del palet; el guardem.
24          Pseguretat = $POS_INT
25          ;Obrim pinca per deixar la primera peca del palet:
26          SET GRIPPER  State= OPEN  GDAT6
27
```

```
   ELSE ; PAL no es 1:
        XPPalet.x = XPPalet.x + 60 ; Incrementem 60mm en X
        ;Interessa que PPalet es situi 200mm sobre el palet.
        XPPalet.Z = XPSeguretat.z + 200
        ; Moviment lineal al punt PPalet:
        LIN PPalet  Vel=2 m/s CPDAT4 Tool[1] Base[1]
        Deixar()
     ENDIF
   ENDIF
ENDFOR

$OUT[0] = TRUE ; Indiquem que el programa ha finalitzat el paletitzat
PTP HOME

END
```

```
DEF Agafar_p( )
; Ens assegurem que la pinca esta oberta:
SET GRIPPER  State= OPEN GDAT2
PTP P4 CONT Vel=100 % PDAT4 Tool[1] Base[1]
PTP P5 CONT Vel=100 % PDAT5 Tool[1] Base[1]
;P6 amb orientacio constant!
LIN P6  Vel=1 m/s CPDAT6 Tool[1] Base[1]
;Tanquem pinca:
SET GRIPPER  State= CLOSE  GDAT3
;Ara tenim peca agafada.
LIN P5  Vel= 1 m/s CPDAT7
PTP P4  Vel= 100 % PDAT6

END
```

```
DEF Deixar( )

; Moviment a la posicio PPalet
LIN PPalet  Vel=1 m/s CPDAT8 Tool[1] Base[1]
LIN_REL {Z -200} ; moviment relatiu a la Z de la base
SET GRIPPER  State= OPEN  GDAT5
LIN_REL {Z +200} ; moviment relatiu a la Z de la base

END
```

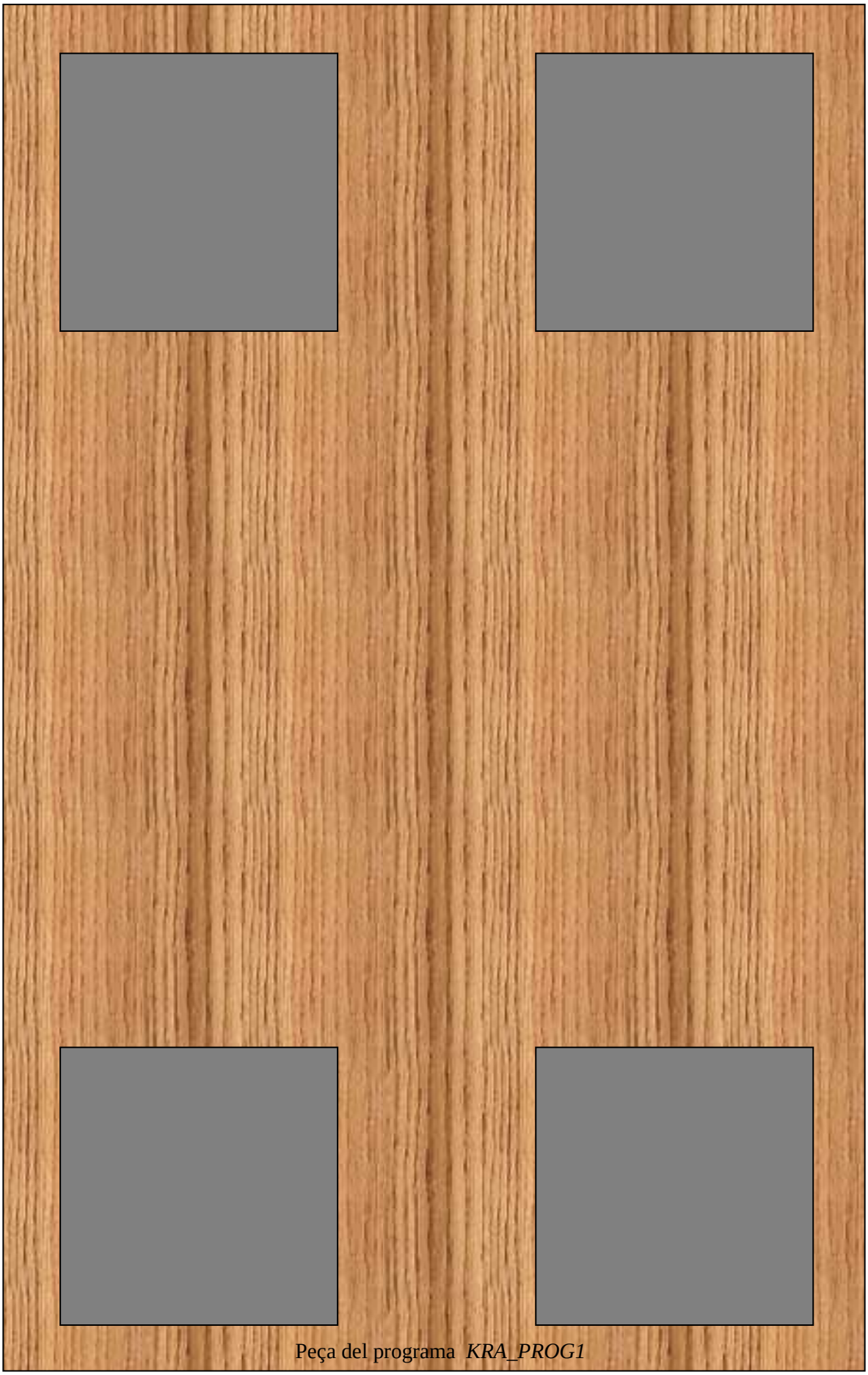
Quart programa del Robot KR 16 (D) Paletitzador:

```
DEF KRD_Prog4( )
INI
; --------------- Seccio declarativa ---------------
INT PAL, I
POS Pseguretat {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0,S 0,T 0}
; --------------- Seccio d'instruccions ---------------
PTP HOME
;Resetejem sortides (per si de cas):
FOR I = 1 TO 16
   $OUT[I] = FALSE
ENDFOR

```

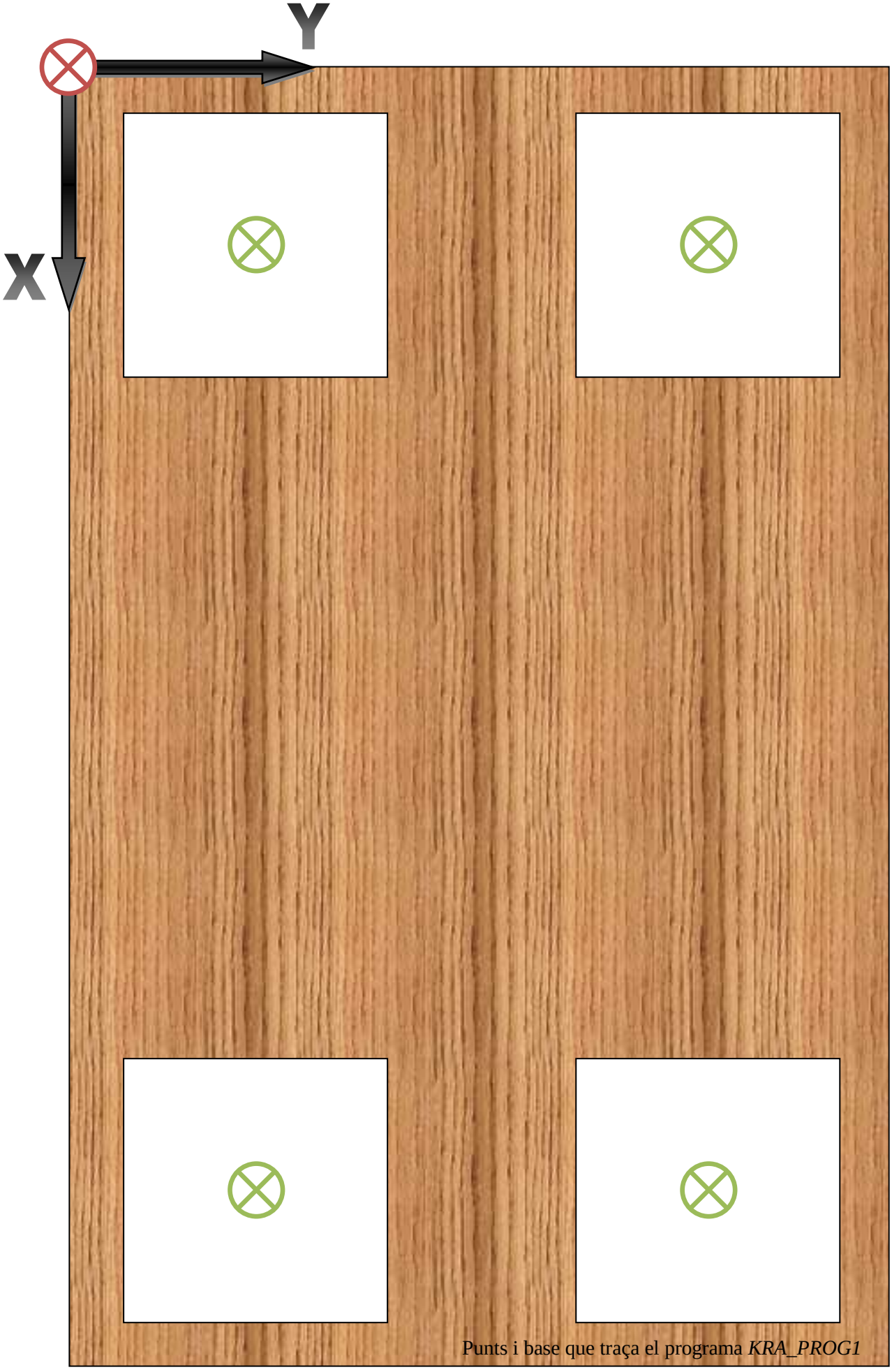
```
; Linia de 5 peces = 1x5:
FOR PAL=1 TO 5 STEP 1
   ;Punt respectiu a base i tool nullframe:
   PTP P1  Vel=100 % PDAT1 Tool[0] Base[0]
   ; En aquest punt respecte la base(1) i la eina (1):
   LIN P2  Vel=2 m/s CPDAT1 Tool[1] Base[1]
   ; Obrim la pinca:
   SET GRIPPER  State= OPEN  GDAT6

   ; Fem el càlcul de la peca a agafar:
   IF PAL == 1 THEN
         LIN P3  Vel=2 m/s CPDAT3 Tool[1] Base[1]
         ; Aquest es el primer punt del palet; el guardem.
         Pseguretat = $POS_INT
         ;Tanquem pinca per recollir la primera peca del palet:
         SET GRIPPER  State= CLOSE  GDAT6

   ELSE ; PAL no es 1:
         XPPalet.x = XPPalet.x + 60 ; Incrementem 60mm en X
         ;Interessa que PPalet es situi 200mm sobre el palet.
         XPPalet.Z = XPSeguretat.z + 200
         ; Moviment lineal al punt PPalet:
         LIN PPalet  Vel=2 m/s CPDAT4 Tool[1] Base[1]
         Agafar()
      ENDIF
      ;Un cop tenim la peca la podem deixar al acumulador:
      Deixar()
   ENDIF
ENDFOR

$OUT[0] = TRUE ; Indiquem que el programa ha finalitzat el paletitzat
PTP HOME

END
-----------------------------------------------------------------------

DEF Agafar( )
; Ens assegurem que la pinca esta oberta (TRUE):
SET GRIPPER  State= OPEN GDAT2
; Moviment a la posicio PPalet
LIN_REL {Z -200}
SET GRIPPER  State= CLOSE  GDAT5
LIN_REL {Z +200}

END
-----------------------------------------------------------------------

DEF Deixar( )

PTP P4 CONT Vel=100 % PDAT4 Tool[1] Base[1]
PTP P5 CONT Vel=100 % PDAT5 Tool[1] Base[1]
;Obrim pinca:
SET GRIPPER  State= OPEN  GDAT3
;Ara ja hem deixat la peca a la pila.
PTP P4  Vel= 100 % PDAT6
PTP HOME

END
```

8.2 DISSENY I PUNTS DE LES PECES CREADES:

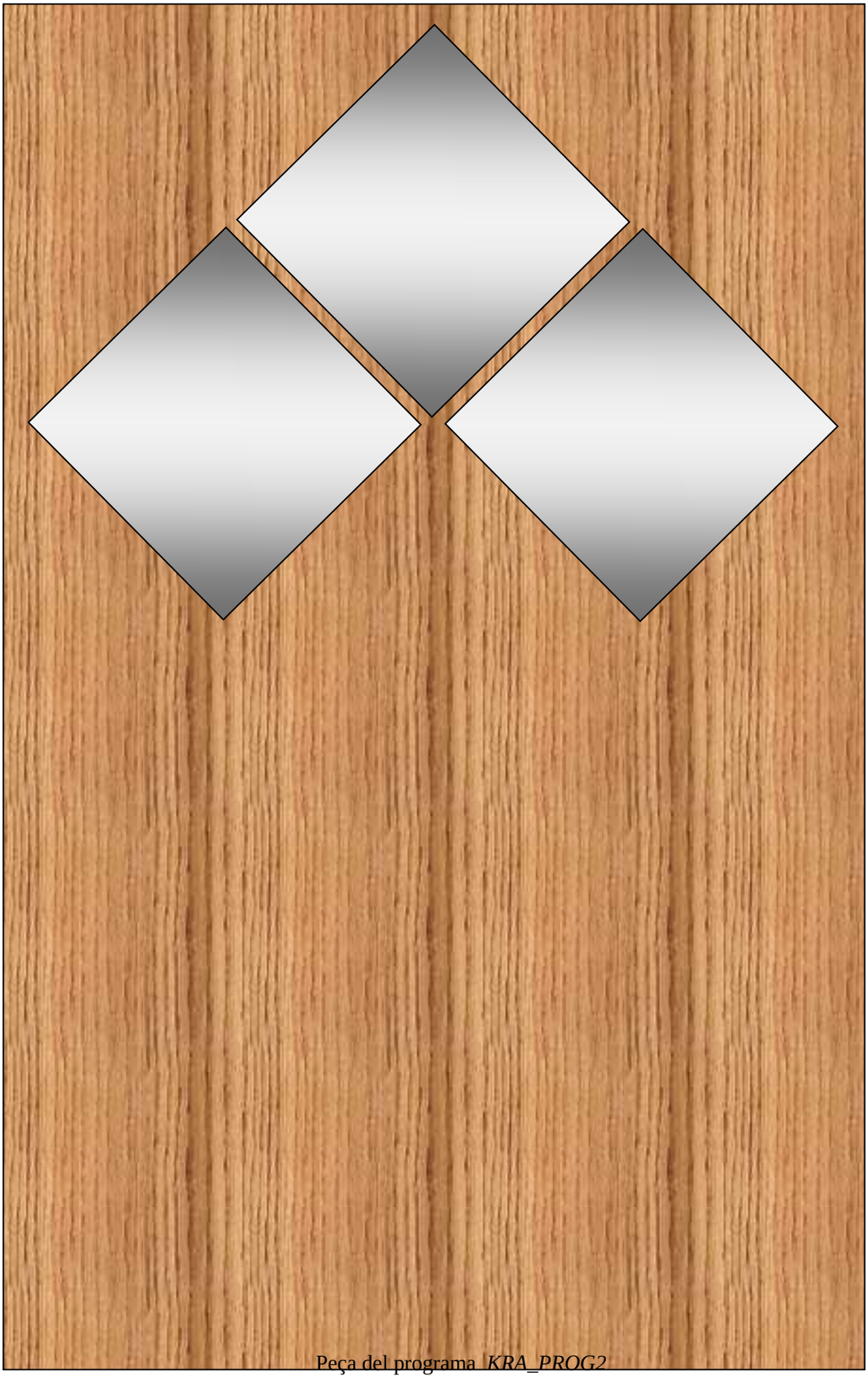
A continuació veurem com són les peces (a tamany real) amb les que opera el robot i els punts programats que cal fer touch up des de la KCP del robot.



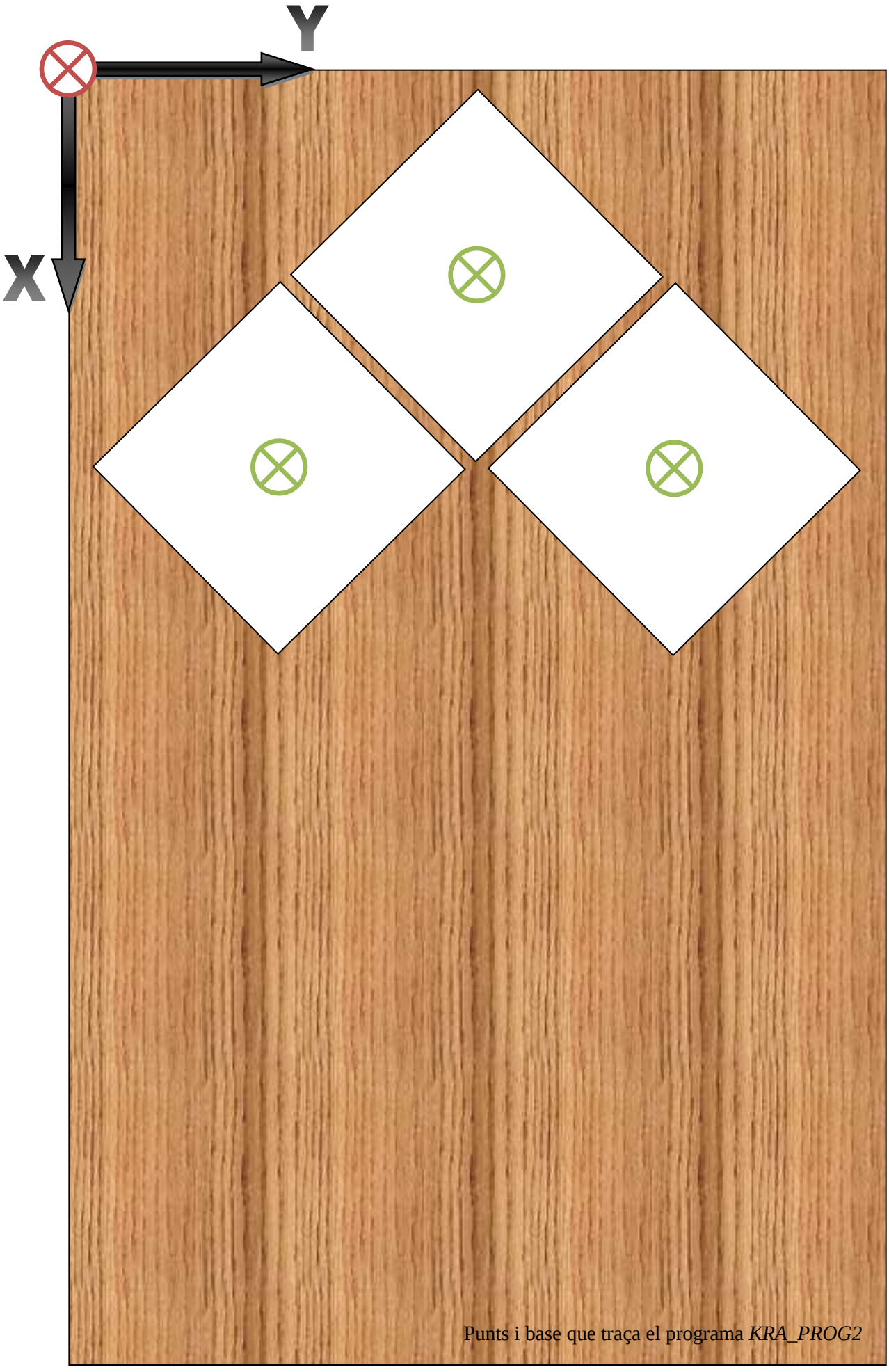
Peça del programa *KRA_PROG1*



Punts i base que traça el programa *KRA_PROG1*

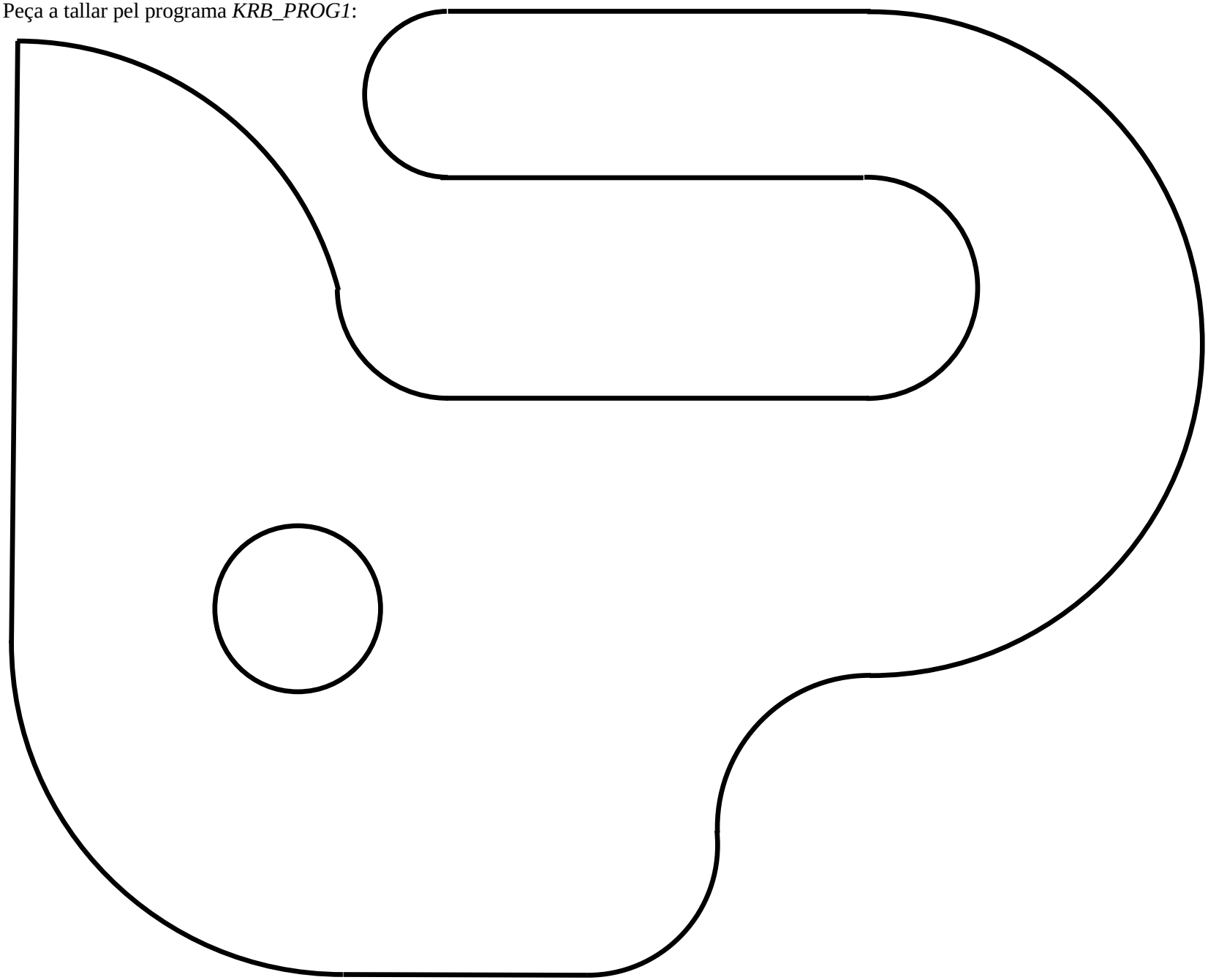


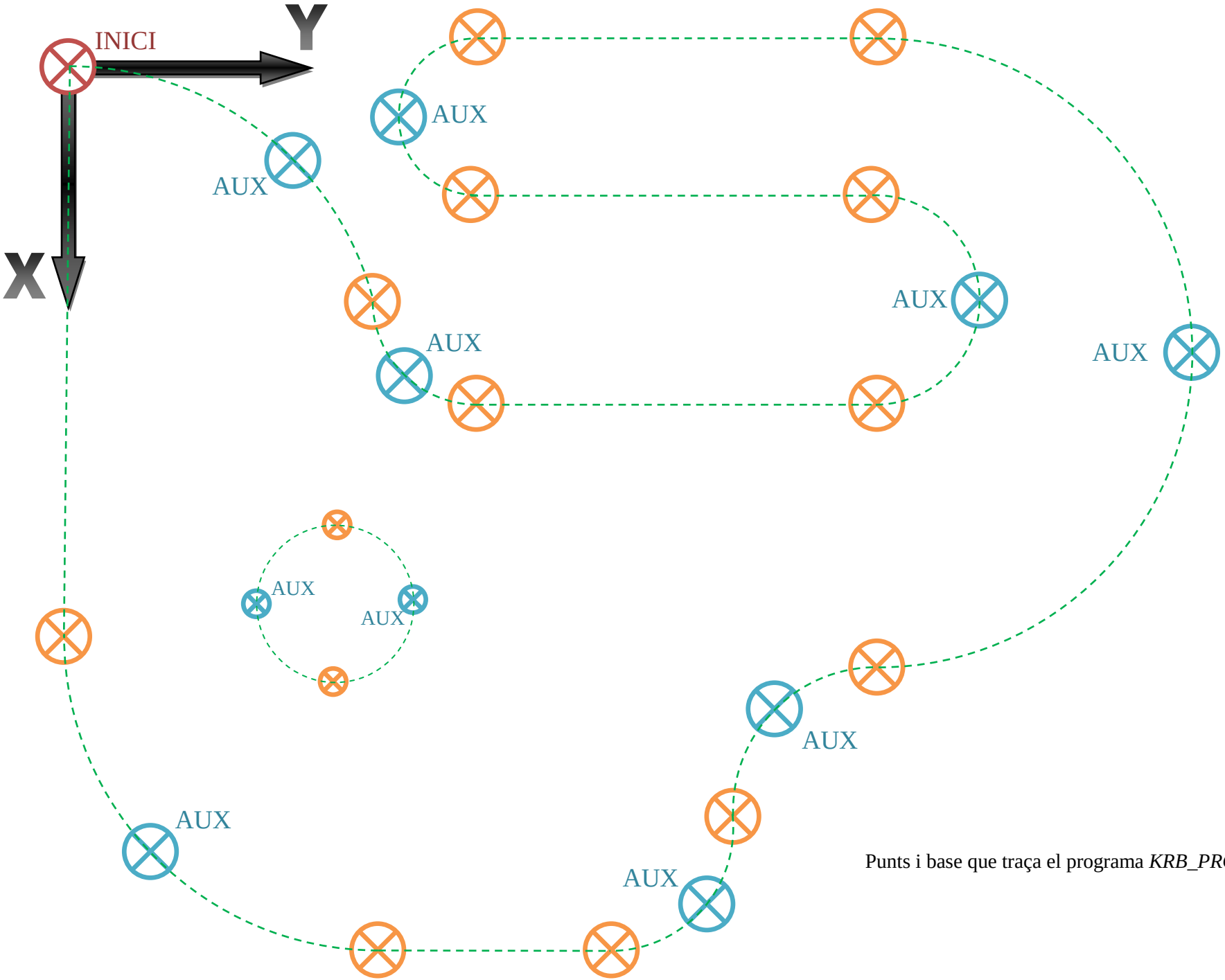
Peça del programa *KRA_PROG2*



Punts i base que traça el programa *KRA_PROG2*

Peça a tallar pel programa *KRB_PROG1*:

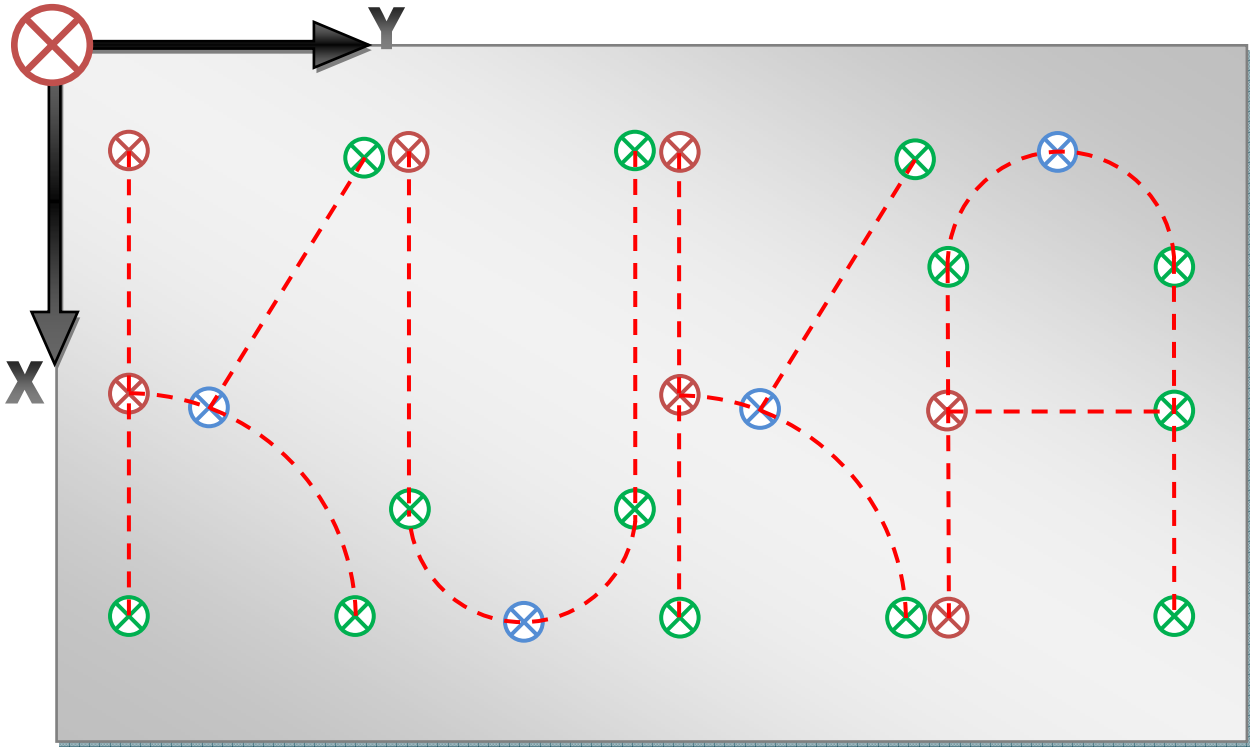




Punts i base que traça el programa *KRB_PROG1*

Peça a serigrafiar pel programa *KRB_PROG2*:

Punts i base que traça el programa *KRB_PROG2*:

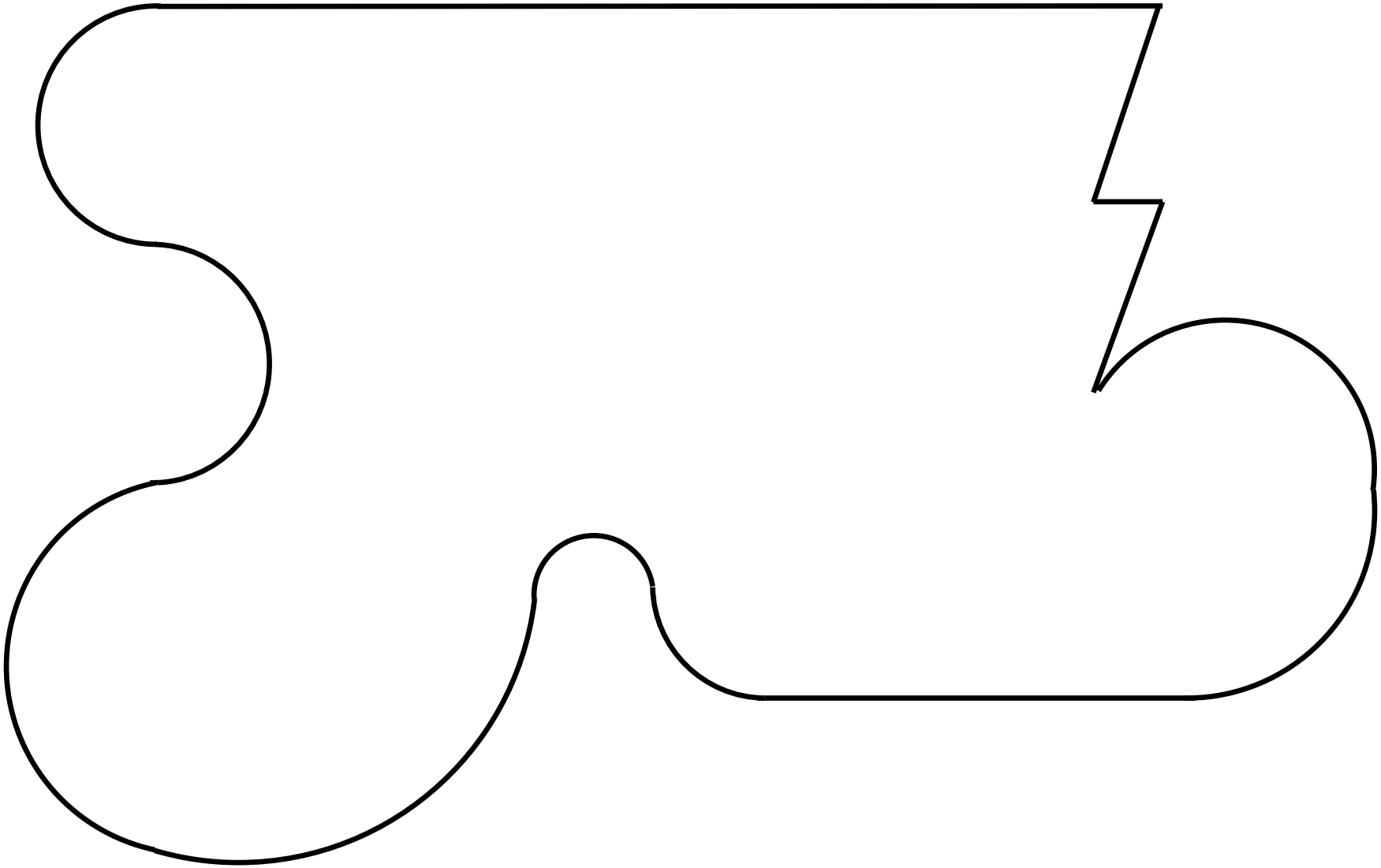


Punts auxiliars en moviments circulars.

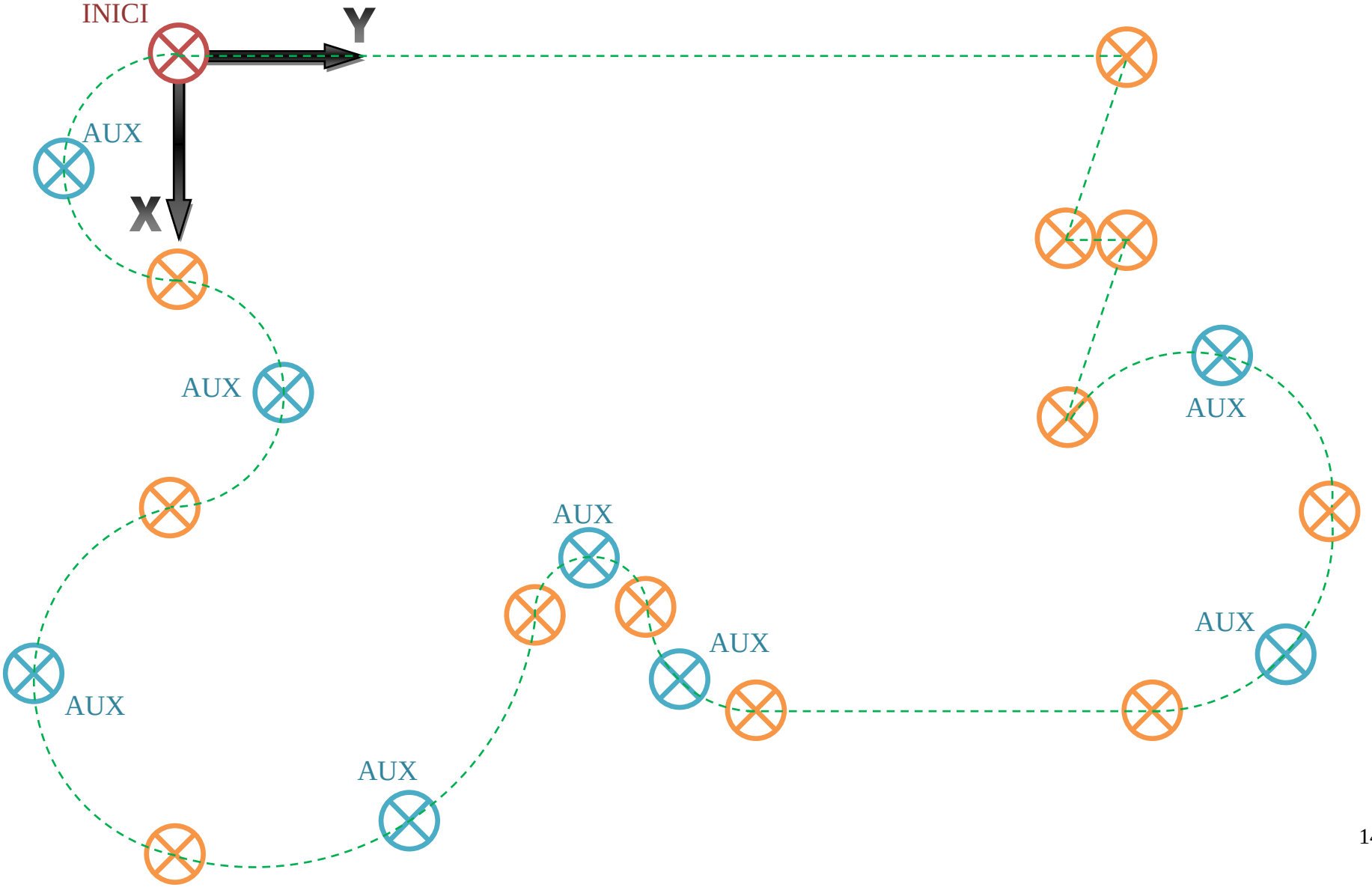
Punts inicials

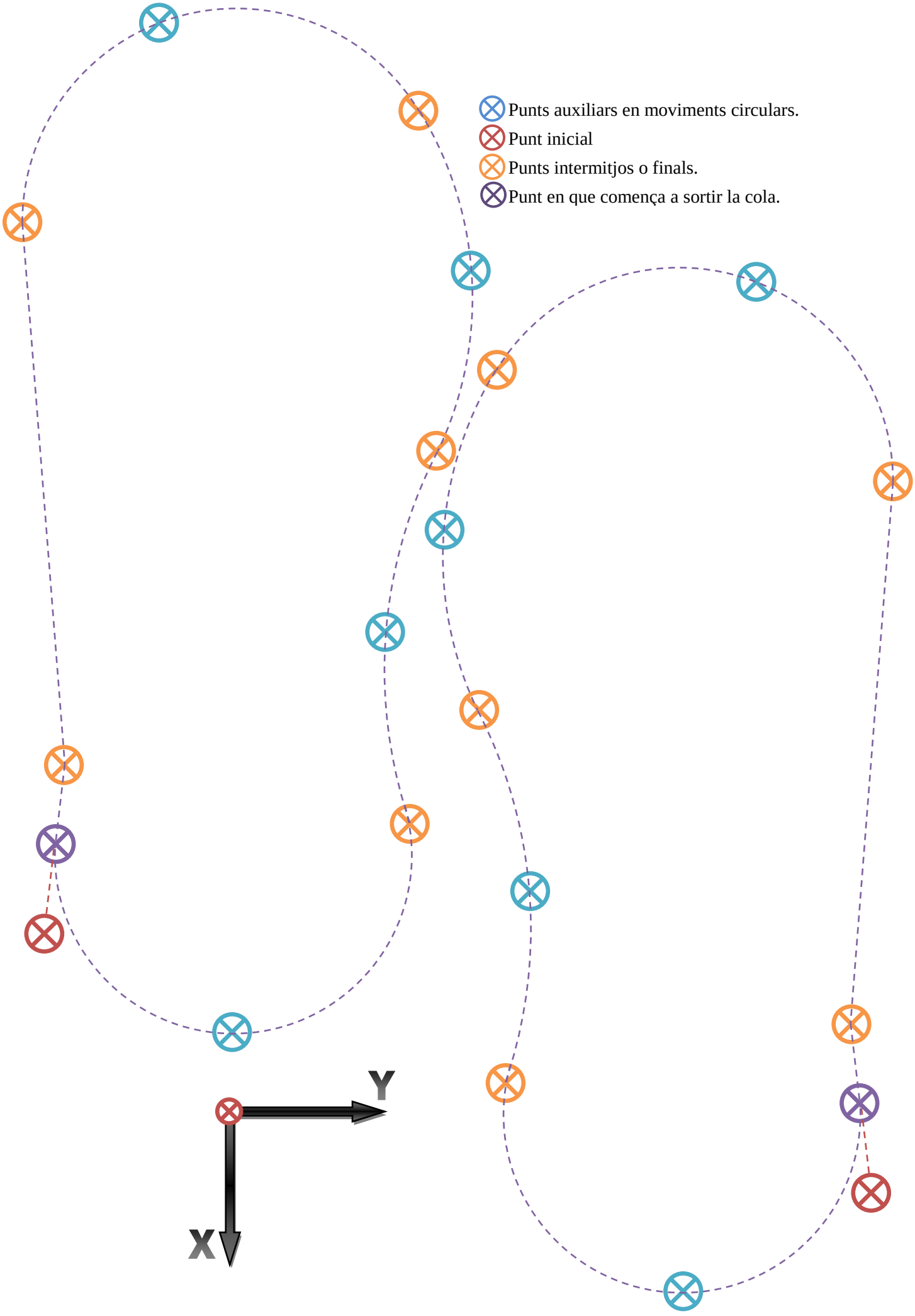
Punts intermitjos o finals.

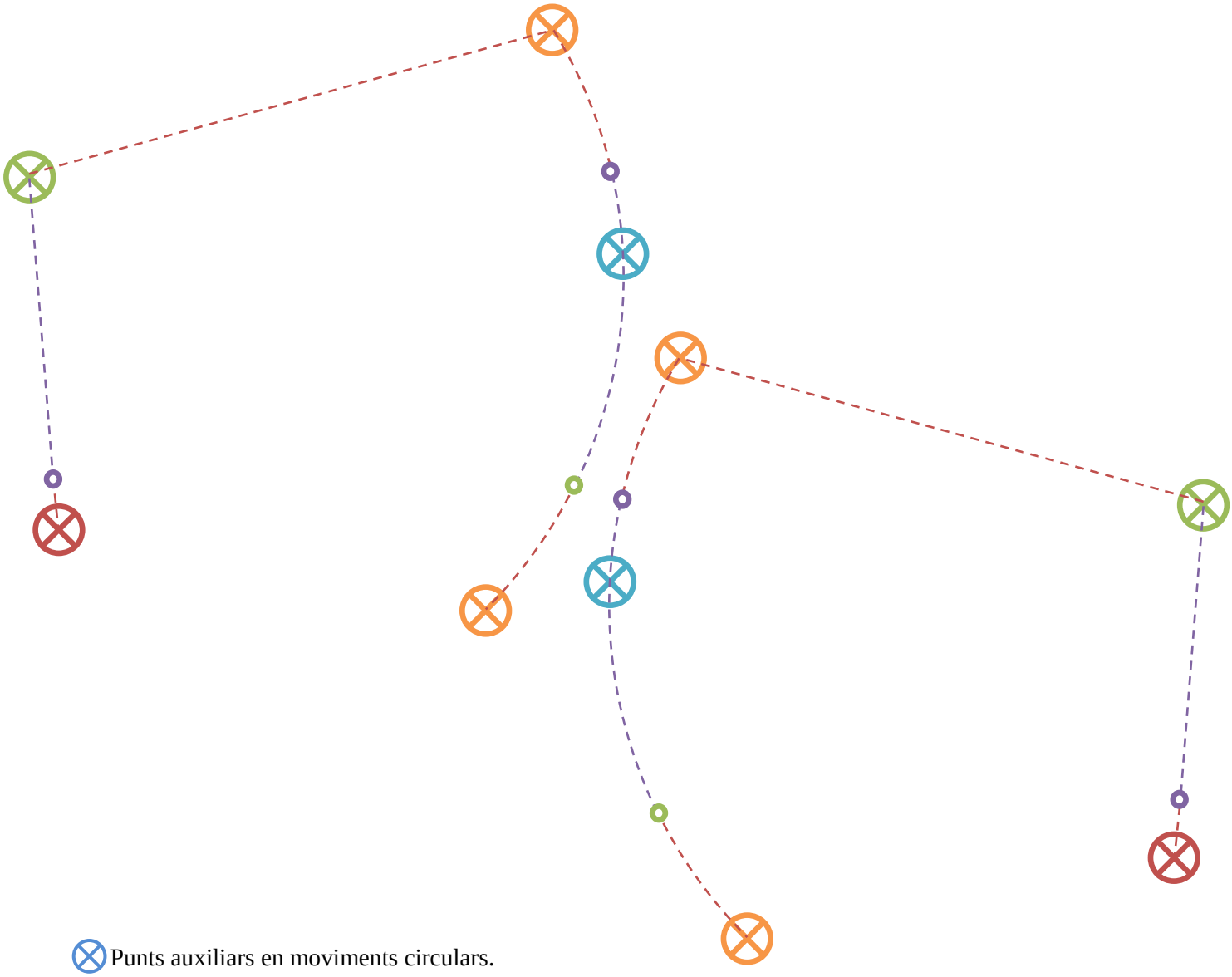
Peça a tallar pel programa *KRB_PROG3*:



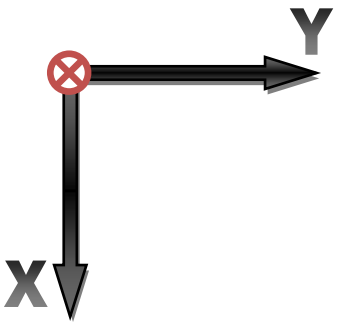
Punts i base que traça el programa *KRB_PROG3*:



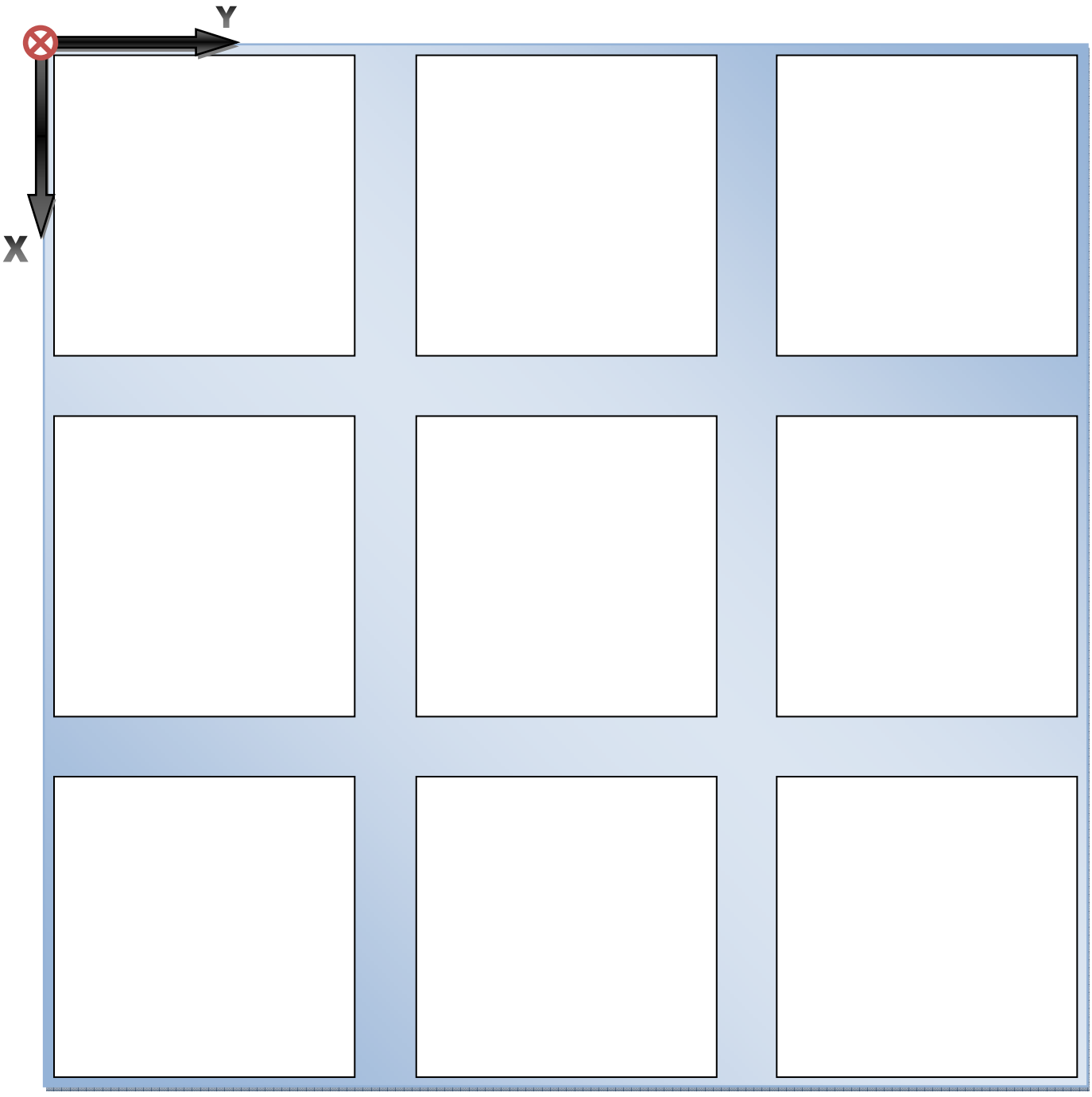

Punts auxiliars en moviments circulars.
Punt inicial
Punts intermitjos o finals.
Punt en que comença a sortir la cola.
Y
X



Punts auxiliars en moviments circulars.
Punt inicial
Punts intermitjos o finals.
Punt en que es deixa de posar la cola.
Punt en que surt la cola.
Punt en que es deixa de posar la cola.



Paletitzat i despaletitzat pels programes *KRD_PROG1* i *KRD_PROG2* :



Vull agrair el suport que he rebut de part de la meva família i amics.

Gràcies també a KUKA robots IBÉRICA, S.A. i tot el seu personal per a facilitar les tasques del projecte, donar-me formació addicional i prestar-me tot el material necessari.

Gràcies als professors de la EPSEVG que m'han mostrat el camí i m'han ensenyat a aprendre.

Gràcies als meus companys que han mirat sempre de fer-me sentir part d'un grup.

I gràcies a la meva parella, que m'ha donat suport durant la realització de tot el projecte i m'ha hagut de recolzar quan les coses es posaven difícils.

GRÀCIES A TOTS