Come muovere ARI V3.

Ari non si muove da sola, si muove su commando. I comandi vengono inviati tramite una riga di commando (Command Line Interface), una GUI oppure un sw.  
Per iniziare a capire usare la CLI è una buona soluzione.

## Elenco comandi

Axxx: Alfa. assegna Alfa, la direzione del robot. xxx è l'angolo in radianti, es 3.14. La direzione zero è definita all'accensione del robot. E' la direzione in avanti. Coincide con l'asse x.

Alfa è positivo in senso antiorario.

Bx: enableFrontSensor. 1 Enable, 0 disable

Cxxx: assegna "raggiorSterzo" in [m]. La variabile è usata nella movimento R2, stabilisce il raggio della circonferenza su cui ruota il robot.

Dxxx: Distance. La distanza che viene percorsa nel prossimo Run espressa in [mm]. La distanza è sempre incrementale.

Ex: EEprom. Esegue operazioni su dei parametri di taratura. Vedi procedura DataEEprom.

E0 SCRIVI i parametri in E2prom,

E1 LEGGI i parametri in E2prom,

E2 rispristina in valori di DEFAULT,

E3 mostra i parametri CORRENTI,

E4 carica i valori di DEFAULT per il modello ARI02,

E5 carica i valori di DEFAULT per il modello ARI03,

Fnxxx: imposta dei parametri del robot. "n" indica quale parametro, "xxx" è il valore.

F0xx ED

F1xx ED\_BASE

F2xx BASELINE mm

F3xx GIRO\_RUOTA mm = sviluppo ruota[mm]/(4\*ppr)

N.B. questi valori vanno attivati con un 3E3

Gxxx: "Guide Mode". definisce il modo di guida nel run "R4". Il teta di feedback, tetaMisura, può provenire dall'odometria, dalla bussola, o altri mix possibili.

Nel codice sotto teta è l'angolo ricavato dagli encoder.

Il modo zero si affida alle misure fatte dagli encoder.

Il modo 1 e 3 ricavano il teta direttamente dalla bussola. Nel modo 3 anche le posizioni x e y

sono calcolate dal teta della bussola.

con il modo 2 è possibile imporre l'angolo corrente come angolo zero. Altrimenti la direzione è

assoluta come determinata dalla bussola.

if (modoGuida == 0) tetaMisura = teta; // ID\_001

if (modoGuida == 1) tetaMisura = tetaCompass; // ID\_009

if (modoGuida == 3) tetaMisura = tetaCompass; // ID\_009

if (modoGuida == 2) {

tetaCompass = 0; // definisco la direzione corrente come zero.

modoGuida = 1;

}

in updatePosition integro posizioni in funzione del modoGuida attivo

if (modoGuida == 3) {

xpos += deltaC\*cos(tetaCompass);

ypos += deltaC\*sin(tetaCompass);

}

else{

xpos += deltaC\*cos(teta);

ypos += deltaC\*sin(teta);

}

Knxx: assegna i guadagni

K0 kpTeta il guadagno proporzionale kp usato nel modo Run 4

K1 kiTeta

K2 kp\_guida il guadagno proporzionale kp usato nel modo Run 1 e 3 (sensore di distanza laterale)

K3 kd\_guida il guadagno derivativo kd usato nel modo Run 1 e 3 (sensore di distanza laterale)

H0: Homing, assegna lo posizione corrente (x, y, teta) = (0, 0, 0)

Ixxx; Integral part. assegna il guadagno della parte integrale kiTeta usato nel modo Run 4.

Lx: Led. puntatore a Led acceso, x=1, o spento x=0.

Mx: Monitor. Attiva, 1, o disattiva, 0, l'invio continuo di misure da parte del robot.

le msiure sono inviate nella routine "updatePosition" ogni 100 ms.

La stringa ha l'header "mon:"

if (monitorDati && (counter >=4) ){

counter = 0;

risposta = "mon;";

risposta += (dDxCnt);

risposta += (";");

risposta += (dSxCnt);

risposta += (";");

risposta += (deltaC);

risposta += (";");

risposta += (teta);

risposta += (";");

risposta += (xpos);

risposta += (";");

risposta += (ypos);

risposta += (";");

risposta += (i\_part);// raggiorSterzo

risposta += (";");

risposta += event.magnetic.x;

risposta += ";";

risposta += event.magnetic.y;

risposta += ";";

risposta += event.magnetic.z;

risposta += ";";

sendAnswer2(port);

}

Nnxxx: assegna i coefficienti di correzione della bussola.

N0: ox

N1: oy

N2: ky

Onxx:

O0: distRef [cm]. riferimento per guida a distanza fissa nei modi R1 e R2

O0: distOstacolo [cm]. distanza minima sotto la quale ARI si arresta

Pxxx: orienta il servo PAN della testa. xxx è in gradi, 90° guarda in avanti, 0 a sinistra e 180 a destra.

in alcuni casi, a seconda del servo usato, è bene limitare l'escursione ad esempio tra 10 e 170°.

Rx: Run. Definisce in che modo effettuare la corsa "Run". Il run è definito con un obbiettivo (target).

Il raggiungimento del target termina il run.

x=4: R4. Il robot si muove sino a percorrere la distanza impostata con "D" nella direzione alfa impostata con "A".

x=5: R5. Il robot si muove sino a orientarsi nella direzione alfa impostata con "A". La rotazione fa perno su una ruota che rimane ferma.

x=6: R6. Il robot si muove sino a orientarsi nella direzione alfa impostata con "A". La rotazione avviene con perno al centro del robot con ruote controrotanti.

x=2: R2. Il robot effettua un percorso circolare con raggio pari al parametro "C". Il raggiorSterzo può anche essere

imposto direttamente con il comando "S". In questo caso sarà lo scorrimento tra le ruote dx e sx.

R1 e 2. i modi 1 e 3 fanno viaggiare il robot a una distanza fissa dalla parete laterale. la misura è fatta orinetando il LIDAR.

i due modi sono in test.

x=99: provoca l'arresto del movimento

Il movimento si arresta quando il lidar vede un ostacolo troppo vicino.

if (lidardist<50 && (statoRun != 5) &&(statoRun != 6) &&(statoRun != 0))

{

statoRun = 99; // senza rampa

motorSpeed = 0; // annullo rampa

motorSpeedRef = 0;

risposta = "ostacolo";

sendAnswer2(port);

}

Sxxx: Scorrimento. Imposta direttamente la varibile "raggiDiSterzo". è un numero tra -1 e 1.

Zero significa scorrimento nullo, cioè le due ruote hanno la stessa velocità comandata.

vedi procedura "differenziale"

ponendo s = raggiorSterzo abbiamo che le velocita' alle ruote valgono:

VA = motorSpeed\*(1-s)

VB = motorSpeed\*(1+s)

Txxx: orienta il servo TILT della testa. xxx è in gradi, 90° guarda in avanti, 0 a sinistra e 180 a destra.

in alcuni casi, a seconda del servo usato, è bene limitare l'escursione ad esempio tra 10 e 170°.

Vxxx: Velocità. Definisce la velocità del robot. é il valore assegnato al PWM.

Zxxx: definisce MAX\_S. è il limite dello scorrimento (positivo e negativo) applicabile dal controllo.