UNIVERSITÀ DI BRESCIA
FACOLTÀ DI INGEGNERIA
Dipartimento di Elettronica per l'Automazione

Laboratorio di Robotica Avanzata
Advanced Robotics Laboratory

Corso di Robotica Mobile
(Prof. Riccardo Cassinis)

Controllo della traiettoria di un robot umanoide

Elaborato di esame di:

Daniele Vido, Simone Romis

Consegnato il: **6 luglio 2007**

Sommario

Il Laboratorio di Robotica Avanzata dell'Università di Brescia ha acquistato un robot umanoide ROBONOVA-1. Il kit standard prevede che l'utente possa controllare il robot tramite il telecomando ad infrarossi fornito in dotazione; il programma allegato, tuttavia, permette solamente di eseguire a comando semplici movimenti dimostrativi. Scopo del presente elaborato è di fornire una nuova modalità di pilotaggio del robot, che consenta di effettuare una camminata continua e di modificare la sua direzione tramite un potenziometro, in modo che il robot curvi in funzione della tensione applicata in ingresso.

Introduzione

Il kit ROBONOVA-1 utilizzato per l'elaborato è costituito da:

- Robot umanoide ROBONOVA-1;
- Software Robobasic, RoboScript e RoboRemocon per la programmazione e il controllo del robot;
- Telecomando a infrarossi Remocon;
- Cavo seriale per la connessione con il calcolatore;
- Caricabatteria;
- Documentazione e istruzioni per il montaggio.

Figura 1: il kit ROBONOVA-1

Di seguito è riportata una breve descrizione dei due componenti principali del kit: il robot ROBONOVA-1 e il software Robobasic.

1.1. Il ROBONOVA-1

Il ROBONOVA-I è un robot umanoide prodotto dalla Hitec Robotics; è alimentato da una batteria NiMH da 1000mA/h a 6V, che garantisce un'autonomia di circa un'ora. I movimenti sono generati da 16 servomotori digitali HSR-8498HB, controllati da una scheda di controllo Micom MR-C3024. La posizione dei servomotori ricalca quella delle articolazioni presenti nel corpo umano; questa soluzione consente al ROBONOVA-1 una grande libertà di movimento: esso infatti può camminare, correre e girarsi, ma anche compiere movimenti più articolati quali capriole o flessioni. Il controllore possiede diversi ingressi, sia di tipo analogico che digitale, fornendo quindi la possibilità di collegare diverse periferiche; esistono diversi moduli accessori forniti direttamente dalla casa produttrice quali, ad esempio, giroscopi, accelerometri, moduli di sintesi vocale o sistemi di comunicazione Bluetooth. In dotazione è fornito un telecomando a infrarossi, che costituisce il principale strumento di interazione con il robot; tramite di esso l'utente può invocare le differenti routine caricate nel microcontrollore.

1.2. Il linguaggio ROBOBASIC

Il Robobasic è un linguaggio di programmazione appositamente ideato per il controllo del ROBONOVA-1; deriva dal BASIC standard, a cui sono state aggiunte istruzioni specifiche per il controllo del robot. Tuttavia, dal momento che il codice dovrà essere eseguito su hardware dalle risorse piuttosto limitate (microcontrollore), sono presenti alcune limitazioni da tenere in considerazione:

- Esistono solo due tipi di dati: interi positivi a un byte (BYTE) e a due byte (INTEGER). Il Robobasic, quindi, non supporta numeri negativi;
- Non sono considerate regole di precedenza degli operatori e non è consentito l'uso di parentesi;
- È bene suddividere computazioni complesse in calcoli più semplici, onde evitare errori imprevedibili;
- Le variabili vanno dichiarate prima di essere utilizzate;
- Tutte le variabili sono visibili globalmente, dal momento che non esistono variabili locali.

2. Il problema affrontato

Come accennato nel sommario, l'obiettivo dell'elaborato consiste nel programmare il robot in modo che esso possa camminare in avanti, modificando la propria direzione di marcia in funzione di una tensione fornita in ingresso da un potenziometro.

Il primo problema che si è dovuto affrontare riguarda la definizione dei movimenti necessari alla deambulazione del robot. Questa necessità è emersa in seguito all'analisi del programma di esempio, fornito in dotazione con il robot. Esso, infatti, risultava inadeguato per due aspetti: per prima cosa permetteva di effettuare solamente singoli passi in avanti e non una camminata continua; secondariamente, esso prevedeva un unico movimento di rotazione, che modifica la direzione del robot di un angolo prefissato. I movimenti sopra descritti, come si può intuire, risultano inadatti per l'obiettivo che ci si proponeva di perseguire; si è pertanto delineata la necessità di modificarli opportunamente o, qualora fosse necessario, di realizzarne di nuovi.

Il secondo problema affrontato è relativo all'interazione con il potenziometro: si è infatti dovuto procedere alla creazione del collegamento fisico tra il robot e lo strumento e, successivamente, all'identificazione dei valori di fondo scala, dal momento che essi non erano noti a priori.

3. La soluzione adottata

3.1. Analisi del programma dimostrativo

Durante la prima fase di lavoro è stata effettuata in un'analisi del programma dimostrativo fornito con il robot, in modo da comprenderne i meccanismi di programmazione e acquisire familiarità con il linguaggio Robobasic.

Il programma `telecomando` è pensato fondamentalmente per dimostrare le capacità di movimento del ROBONOVA-1; l'interazione, come suggerisce il nome del file, avviene tramite il telecomando a infrarossi: ad ogni tasto è associata una routine che fa compiere al robot movimenti più o meno complessi, che spaziano dall'assunzione della posizione di riposo (accovacciato) a capriole laterali, in avanti o indietro. Una routine è tipicamente costituita da una sequenza di comandi di tipo MOVE, che consentono di modificare la posizione di uno o più motori del robot. In questa categoria rientrano due istruzioni utilizzate molto frequentemente: MOVE24, che permette di specificare una posizione per ciascuno dei 16 servomotori, e MOVE, che invece consente di muovere un gruppo di motori, dove per gruppo si intende l'insieme dei servomotori che costituiscono un arto del robot (ad esempio, il gruppo D è associato alla gamba sinistra). Un movimento elementare del robot può quindi essere ottenuto in due modi: con un singolo comando MOVE24, che determina il movimento di tutti i 16 servomotori, oppure con una serie di comandi MOVE, ciascuno dei quali impone il movimento di un arto. La combinazione di più movimenti elementari consente poi di eseguire movimenti più articolati, quali, ad esempio, quelli citati precedentemente.

La fase di analisi ha consentito inoltre di chiarire le modalità di interazione con il telecomando a infrarossi; a questa periferica è associato un ingresso analogico, il numero 7, che fornisce in output un valore numerico che varia a seconda di quale tasto sia premuto in un determinato istante. La lettura del valore avviene tramite un'apposita istruzione, `remocon(1)`, che restituisce un valore numerico di tipo `BYTE`; a partire dal valore acquisito sarà poi possibile ricostruire quale tasto è stato premuto, invocando poi la (eventuale) routine ad esso associata. Si noti che l'interazione del telecomando richiede un certo intervallo di tempo, stimato in circa 0,5 secondi; la sua esecuzione, quindi, introduce un certo ritardo nei movimenti del robot, di cui è necessario tenere conto.

L'obiettivo principale della fase di analisi, tuttavia, consisteva nel ricercare eventuali movimenti preprogrammati che potessero essere adatti per il lavoro che ci si proponeva di svolgere. I movimenti individuati sono stati tre: `forward_walk, left_turn, right_turn.`

Il primo consente di eseguire un singolo passo in avanti; il movimento inizia a partire dalla posizione eretta di riposo (invocabile tramite la routine `standard_pose`) e si svolge all'incirca in questo modo: per prima cosa viene spostato in avanti il piede destro; poi viene portato in avanti quello sinistro così da appoggiarlo più avanti del destro e, infine, il robot si riporta in posizione eretta riallineando i due piedi.

I movimenti `left_turn` e `right_turn` consentono invece di far ruotare il robot di un numero di gradi prefissato, che è stato stimato essere di circa 14,5°[1]. Il movimento di rotazione a destra avviene sfruttando l'attrito del piede sinistro con il pavimento e si svolge come segue: il robot porta in avanti il piede sinistro e lo appoggia qualche centimetro più in avanti del destro; successivamente, lo riporta indietro facendolo strisciare sul terreno e facendo leva su di esso. In questo modo, quindi, il piede destro funge da perno e l'attrito del piede sinistro con il terreno causa la rotazione vera e propria, facendo girare il robot verso destra. La rotazione verso sinistra avviene sfruttando lo stesso principio, ma facendo leva sulla gamba destra e utilizzando come perno la sinistra.

3.2. Deambulazione del robot

Terminata l'analisi, si è passati ad affrontare il problema principale, ovvero quello della deambulazione del robot. L'obiettivo era quello di creare le routine/movimenti che permettessero di ottenere una camminata in avanti continua ed una rotazione di un numero desiderato di gradi. In questa fase, quindi,

[1] Valore ottenuto sperimentalmente, calcolando il numero medio di passi che erano necessari al robot per eseguire una rotazione di circa 90°.

ci si è basati soprattutto sui movimenti identificati durante l'analisi, combinandoli e modificandoli opportunamente in modo da ottenere i comportamenti desiderati. Di seguito verranno esposte in dettaglio le modalità con cui questi risultati sono stati raggiunti.

3.2.1. Camminata in avanti

Il primo problema affrontato è stato quello della creazione della camminata continua. Come detto precedentemente, la routine `forward_walk` fa compiere al robot un singolo passo in avanti; una prima proposta di soluzione, perciò, è stata quella di inserire tale movimento in un ciclo infinito. Tuttavia, la camminata ottenuta in questo modo non era né fluida né naturale: il robot infatti, alla fine di ogni passo ritornava in posizione eretta e si fermava per un istante, interrompendo momentaneamente l'avanzamento e introducendo un piccolo ritardo.

Stabilita quindi l'inadeguatezza di tale soluzione, si è deciso di cambiare approccio: è risultata infatti evidente la necessità di comprendere meglio quali movimenti avrebbero dovuto costituire la camminata vera e propria, in modo da renderla il più possibile naturale e continua. La soluzione più ovvia, vista anche la struttura umanoide del robot, è sembrata quindi l'osservazione della camminata di una persona, in modo da individuare i movimenti elementari di cui è composta.

Da quest'analisi è emerso che i movimenti che il robot avrebbe dovuto eseguire sono fondamentalmente tre:

- un passo iniziale, per cominciare la camminata. Ad esempio, partendo dalla posizione eretta, sarà necessario portare il piede sinistro in avanti.
- un movimento da ripetere ciclicamente, che costituisce la camminata vera e propria e che chiameremo *passo continuo*. Ipotizzando di avere eseguito il passo iniziale come descritto al punto precedente, quindi, sarà necessario portare in avanti il piede destro e successivamente riportare in avanti il sinistro. Si noti che in questo modo la posizione iniziale del movimento coincide con quella finale; di conseguenza esso potrà essere ripetuto ciclicamente senza discontinuità o interruzioni.
- un passo finale, che consenta di terminare la camminata. Ipotizzando di avere il piede sinistro in avanti (posizione raggiunta al termine sia del passo iniziale che di quello continuo), sarà necessario riallineare il piede destro al sinistro, riportandosi in posizione eretta.

La routine che permette al robot di effettuare il passo iniziale, denominata `passo_in`, è stata generata isolando le prime istruzioni di `forward_walk`. Il codice risultante è stato quindi il seguente.

```
passo_in:
  SPEED 5
  'spostare il peso a destra
  MOVE2485,71,152,91,112,60,100,40,80,,,,100,40,80,,,,112,76,145,93,92,60,

  SPEED 10
  'sollevare il piede sinistro
  MOVE2490,107,105,105,114,60,90,40,80,,,,100,40,80,,,,114,76,145,93,90,60,

  'portare in avanti il piede sinistro
  MOVE2490,56,143,122,114,60,80,40,80,,,,105,40,80,,,,113,80,145,90,90,60,
  MOVE2490,46,163,112,114,60,80,40,80,,,,105,40,80,,,,112,80,145,90,90,60,

  'appoggiare a terra il piede sinistro
  MOVE G6A,91,42,163,113,112,60
  MOVE G6B,80,40,80,100,100,100
  MOVE G6C,105,40,80,100,100,100
  MOVE G6D,110,84,138,95,87,60
  WAIT
  RETURN
```

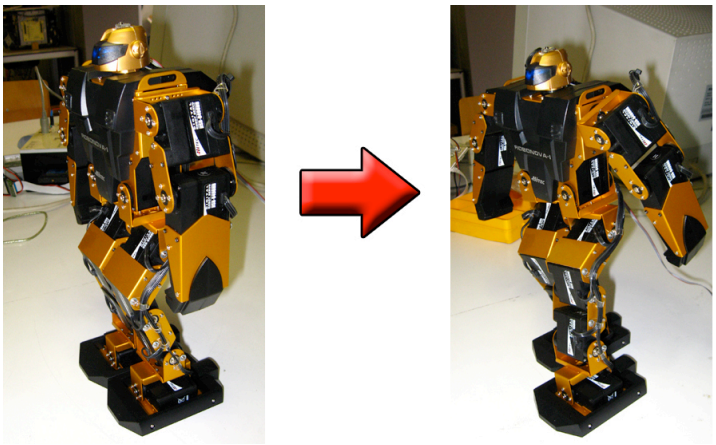

Figura 2: le varie fasi del movimento passo_in

Per quanto riguarda il movimento di passo continuo, invece, è stato necessario ricombinare alcune delle parti del movimento `forward_walk`; in particolare è stata mantenuta inalterata la sezione di codice che fa spostare il piede destro in avanti, mentre sono state spostata al termine della routine le istruzioni che fanno muovere in avanti il piede sinistro. Il codice risultante è stato quindi il seguente:

```
passo_con:
  SPEED 12
  'avanzare facendo leva sul piede sinistro
  MOVE24 100,66,141,113,100,100,90,40,80,,,,100,40,80,,,,100,83,156,80,100,100,
  MOVE24 113,78,142,105,90,60,100,40,80,,,,100,40,80,,,,90,102,136,85,114,60,


  SPEED 16
  'alzare il piede destro
  MOVE24 113,76,145,93,90,60,100,40,80,,,,90,40,80,,,,90,107,105,105,114,60,
  'spostare il piede destro in avanti
  MOVE24 113,80,145,90,90,60,105,40,80,,,,80,40,80,,,,90,56,143,122,114,60,
  MOVE24 112,80,145,90,90,60,105,40,80,,,,80,40,80,,,,90,46,163,112,114,60,

  SPEED 12
  'avanzare facendo leva sul piede destro
  MOVE24 100,83,156,80,100,100,100,40,80,,,,90,40,80,,,,100,66,141,113,100,100,
  MOVE2490,102,136,85,114,60,100,40,80,,,,100,40,80,,,,113,78,142,105,90,60,

  SPEED 16
  'alzare il piede sinistro
  MOVE2490,107,105,105,114,60,90,40,80,,,,100,40,80,,,,114,76,145,93,90,60,

  'portare in avanti il piede sinistro
  MOVE2490,56,143,122,114,60,80,40,80,,,,105,40,80,,,,113,80,145,90,90,60,
  MOVE2490,46,163,112,114,60,80,40,80,,,,105,40,80,,,,112,80,145,90,90,60,

  'appoggiare il piede sinistro a terra
  MOVE G6A,91,42,163,113,112,60
  MOVE G6B,80,40,80,100,100,100
  MOVE G6C,105,40,80,100,100,100
  MOVE G6D,110,84,138,95,87,60
  WAIT
  RETURN
```

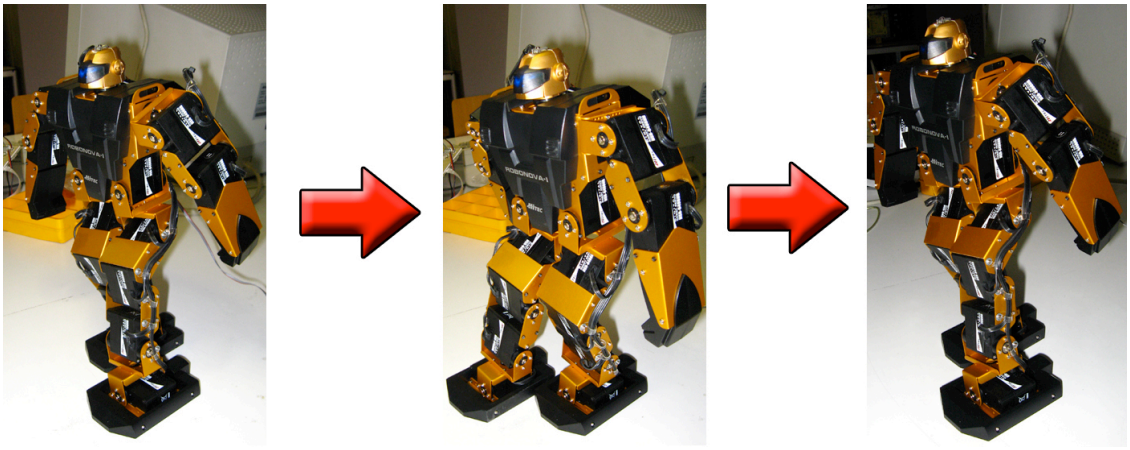

Figura 3: le varie fasi del movimento passo_con

La realizzazione del passo finale, infine, è stata più complessa, dal momento che i movimenti elementari per eseguire questo movimento non sono stati individuati in nessuna delle routine presenti nel programma. Si è quindi deciso di definirli ex-novo; ciò è stato possibile tramite l'utilizzo di un'apposita utility fornita con il robot: la funzione ServoMotorRealtimeControl.

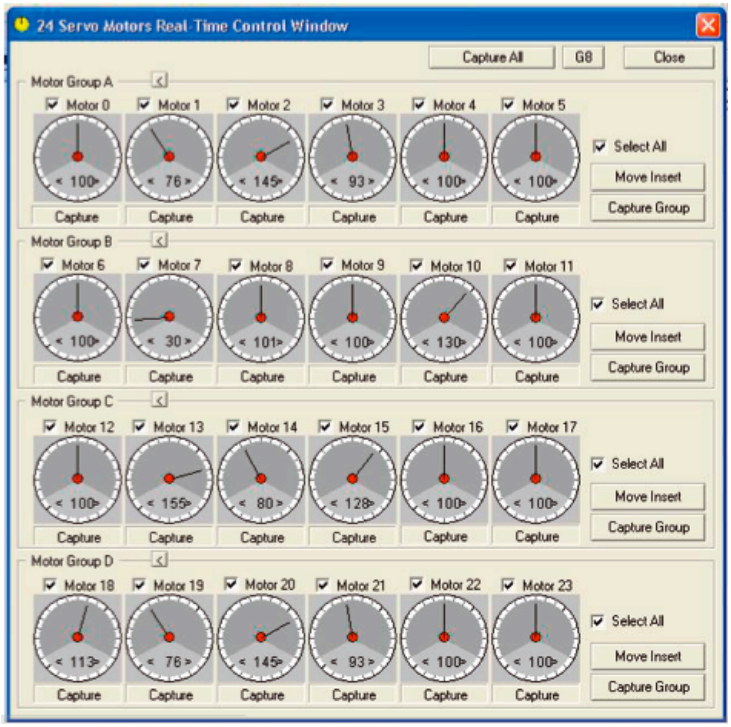

Figura 4: La schermata dell'utility ServoMotorRealtimeControl

Questa applicazione consente sia di leggere la posizione di ciascuno dei servomotori del robot, sia di controllarli direttamente: è possibile quindi impostarne la posizione, oppure bloccarli e sbloccarli a piacere. La programmazione dei movimenti può avvenire quindi tramite il procedimento "leading-by-nose", che tipicamente si svolge in questo modo: per prima cosa si sblocca il motore (o il gruppo di motori) di cui si vuole modificare la posizione, così da poterlo muovere a piacere. Successivamente si procede a muovere le parti interessate, fino a far assumere al robot la posizione desiderata; è poi necessario bloccare nuovamente i motori del robot ed eseguire una lettura delle posizioni degli stessi. Infine, tramite i pulsanti "move insert", sarà possibile convertire le posizioni dei motori di ciascun

gruppo in una corrispondente istruzione MOVE, che verrà automaticamente inserita nel programma. Con questo procedimento è stato quindi generata la routine `passo_fin`, che si compone di tre movimenti elementari: supponendo che il robot abbia il piede sinistro in avanti, esso per prima cosa sposta il peso sulla gamba sinistra, sollevando nel frattempo la gamba destra; successivamente porta in avanti il piede destro e, infine, assume la posizione eretta. Il codice della routine `passo_fin` è riportato di seguito:

```
passo_fin:
  SPEED 4

  'portare il peso sulla gamba sinistra
  MOVE G6A,109,46,162,110,91,60
  MOVE G6D,94,89,131,98,107,60
  SPEED 6
  MOVE G6B,123,40,80,100,100,100
  MOVE G6C,80,45,82,100,100,100
  WAIT

  SPEED 9

  'portare in avanti il piede destro
  MOVE G6A,110,48,174,102,101,60
  MOVE G6B,124,55,80,100,100,100
  MOVE G6C,80,45,82,100,100,100
  MOVE G6D,93,76,131,117,104,60
  WAIT

  'posizione eretta (standard pose)
  MOVE G6A,100,76,145,93,100,100
  MOVE G6D,100,76,145,93,100,100
  MOVE G6B,100,30,80,100,100,100
  MOVE G6C,100,30,80,100,100,100
  WAIT
  RETURN
```

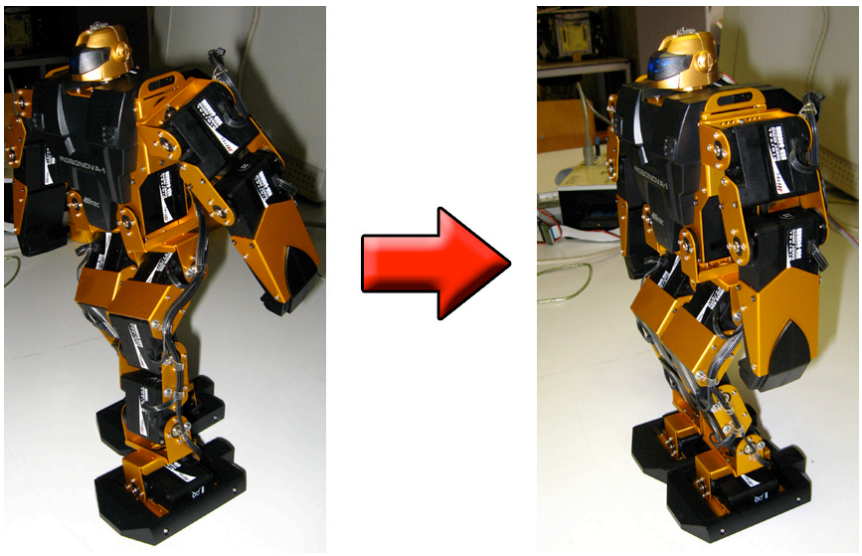

Figura 5: le varie fasi del movimento passo_fin

Una volta definiti i movimenti necessari, si è provveduto a inserirli in una routine che permettesse al robot di camminare in maniera continua. La routine principale, chiamata `cammina`, consiste quindi in un ciclo, che consente di ripetere il passo continuo (`passo_con`) più volte. Il codice che la costituisce è riportato di seguito.

```
cammina:
  GOSUB testa_pot
  GOSUB passo_con
```

```
  GOSUB check_stop
  GOTO cammina
```

Come si può notare viene qui invocata la routine `testa_pot`; essa, dal momento che si occupa di gestire l'interazione con il potenziometro, verrà esposta e commentata in dettaglio nei paragrafi successivi. La routine `check_stop`, invece, è stata creata per consentire di arrestare il movimento del robot tramite il telecomando a infrarossi; essa viene eseguita all'interno del ciclo principale, al termine di ogni passo continuo. Scopo di questa funzione è quindi di controllare se sul telecomando in quel momento è premuto il tasto stop o meno; nel caso in cui lo sia, viene richiamato il passo finale e viene arrestata la camminata. Il codice che la costituisce è il seguente:

```
check_stop:
  A = REMOCON(1)
  IF A=26 THEN
          GOSUB passo_fin
          GOSUB standard_pose
          GOTO main_exit
  ENDIF
  RETURN
```

Infine è stata creata una routine di supporto, denominata `inizia_cammino`, che permette di dare inizio alla camminata: essa esegue il passo introduttivo, invocando poi la routine `cammina` che avvierà il ciclo principale. Il codice di `inizia_cammino` è riportato di seguito

```
inizia_cammino:
  GOSUB passo_in
  GOTO cammina
```

3.2.2. Controllo della traiettoria del robot

Terminata la realizzazione del movimento di camminata, si è affrontato il problema del controllo della traiettoria del robot. L'obiettivo di questa fase era di fare si che il robot potesse ruotare di un angolo variabile; in questo caso il movimento preprogrammato è risultato essere adeguato allo scopo, a differenza di quanto accaduto per la camminata. La routine `turn_lef`, infatti, permetteva di ruotare il robot di un angolo fisso, stimato in circa 14,5°; si è quindi pensato di creare un ciclo che consentisse di ripeterlo un numero arbitrario di volte, in modo da ottenere un angolo di rotazione variabile.

Ad una prima analisi, potrebbe sembrare che una rotazione eseguita a passi discreti di circa 15 gradi sia troppo approssimativa; tuttavia si è optato per questa soluzione anche alla luce delle intrinseche difficoltà che il robot ha nel modificare la propria direzione. Per prima cosa, infatti, esso tende a sbilanciarsi durante la fase di rotazione; le oscillazioni che si generano influenzano molto l'efficacia del movimento, facendo si che l'effettivo angolo di rotazione vari anche di molto da un'esecuzione all'altra. Secondariamente, anche le caratteristiche della superficie su cui avviene il movimento influenzano di molto l'efficacia dello stesso: l'effettiva ampiezza dell'angolo di rotazione varia anche in dipendenza della ruvidezza, rigidità e regolarità del piano di appoggio. Di conseguenza è stata ritenuta superflua una maggiore precisione nel controllo della direzione, dal momento che essa sarebbe stata comunque vanificata da fattori esterni non sempre controllabili.

Si è quindi stabilito che la rotazione sarebbe stata costituita da una sequenza di movimenti `turn_left` e `turn_right`; a questo punto, però, si è presentata una difficoltà: il movimento di rotazione a sinistra risultava più efficace di quello a destra, probabilmente a causa di piccole irregolarità nella simmetria del robot: esso, infatti, per ruotare di 90° a sinistra richiedeva circa 6-7 movimenti `turn_left`, mentre per ruotare dello stesso angolo verso destra erano necessari dai 9 ai 12 movimenti `turn_right`. Si è quindi provveduto a modificare il movimento `turn_right`, così da da aumentarne l'efficacia: in particolare si è modificata leggermente la posizione della caviglia destra, in modo che durante la rotazione il robot esercitasse più forza sul piede sinistro; così facendo l'azione del piede sinistro sul terreno risultava più energica, aumentando l'angolo di rotazione ottenuto.

Una volta adattati e regolarizzati i due movimenti di rotazione, si è creata la routine che a convertisse il valore dell'angolo di rotazione desiderato nel numero di passi di rotazione a destra o sinistra necessari. Il codice risultante è il seguente:

```
POT = 24
```

```
calcola_passi:
  TEMP = VARIAZIONE * 90
  ANGOLO = TEMP /POT
  PASSI = ANGOLO / 15
  WAIT
  RETURN
```

Per prima cosa si noti che questa procedura presuppone l'interazione con il potenziometro; le modalità con cui ciò avviene verranno descritte in dettaglio nel paragrafo successivo. Per ora è sufficiente sapere che la variabile POT, impostata a 24, rappresenta la variazione del valore di input del potenziometro corrispondente ad un cambio di direzione desiderato di circa 90°. La routine, quindi, calcola l'ampiezza della rotazione desiderata, implementando questa proporzione.

variazione input : 24 = angolo : 90

Si noti che è stato necessario introdurre una variabile temporanea `TEMP`, a causa delle limitazioni imposte dal linguaggio Robobasic: il manuale infatti consiglia spezzare i calcoli più complessi in tanti passaggi elementari, per non rischiare che si verifichino errori casuali e del tutto imprevedibili.

Una volta calcolata l'ampiezza desiderata della rotazione, si procede a determinare il numero di passi che dovranno essere eseguiti dal robot per cambiare effettivamente direzione: l'angolo totale viene quindi diviso per l'angolo associato ad un singolo movimento di `left_turn`/`right_turn`. Si noti che questo valore è stato arrotondato da 14,5° a 15°, dal momento che il linguaggio Robobasic supporta solamente valori interi.

In seguito è stata creata la routine che esegue il movimento di rotazione vero e proprio; di seguito, a titolo di esempio, viene riportato il codice della procedura `gira_sx`, che permette di fare ruotare il robot verso sinistra. La routine `gira_dx`, necessaria per ruotare verso destra, è del tutto simile.

```
gira_sx:
  FOR I = 1 TO PASSI
         GOSUB left_turn
         WAIT
         GOSUB standard_pose
         WAIT
  NEXT I
  RETURN
```

Come si può vedere il ciclo viene ripetuto un numero di volte pari al valore di `PASSI`; al termine di ogni singolo movimento `left_turn` il robot si riporta in posizione eretta (`standard_pose`), in modo da stabilizzarsi ed assorbire le oscillazioni generate dal movimento stesso.

3.3. Interazione con il potenziometro

Una volta risolta la problematica relativa alla corretta deambulazione del robot, si è passati alla seconda parte del lavoro, ovvero l'interazione con il potenziometro.

Il dispositivo vero e proprio è stato creato riadattando un vecchio pannello di controllo; esso era infatti dotato di un joystick analogico che risultava particolarmente adatto per il pilotaggio del robot. In particolare si è sfruttato il movimento della leva lungo l'asse X per modulare la tensione in uscita: l'azione di spostare la leva totalmente a sinistra avrebbe quindi fornito in output la tensione minima, imponendo al robot una rotazione di 90° a sinistra; la leva totalmente spostata verso destra avrebbe invece determinato il valore di tensione massimo, facendo ruotare il robot di 90° verso destra.

Procuratici il dispositivo, si è poi provveduto a realizzare un cavo per la connessione con il microcontrollore: il collegamento avviene tramite uno degli 8 ingressi analogici di cui esso è dotato; ciascuno ingresso è costituito da 3 pin: massa (il più esterno), alimentazione e segnale d'ingresso (il più interno). Il cavo creato ha il filo che trasmette il segnale d'ingresso marcato in modo da renderlo riconoscibile e poterlo di conseguenza collegare in maniera corretta. Lo schema che descrive il circuito risultante è riportato di seguito:

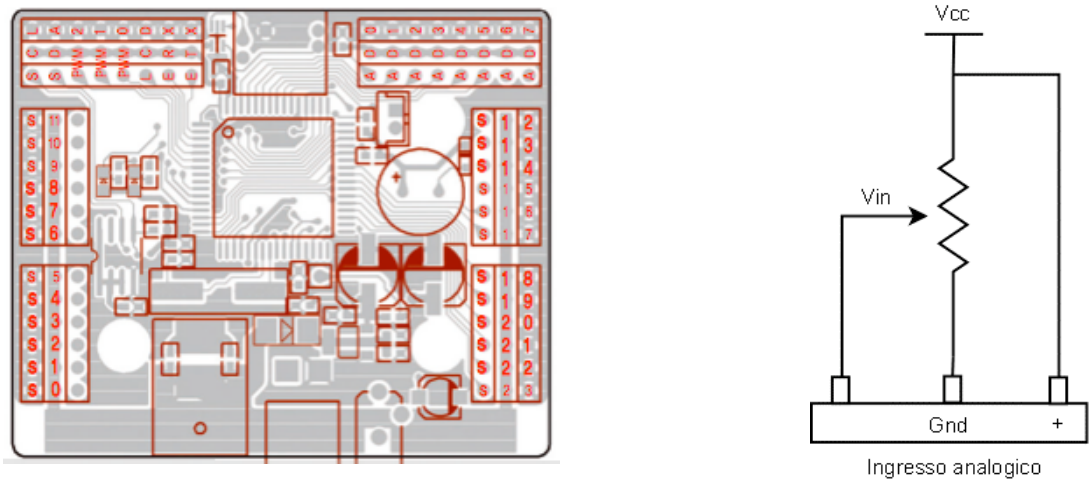

Figura 6: schema del microcontrollore e del circuito risultante

Figura 7: il pannello di controllo del robot

Figura 8: primo piano del microcontrollore e del collegamento con il potenziometro

Realizzato il collegamento fisico si è passati ad individuare i valori di uscita del potenziometro, dal momento che le caratteristiche del dispositivo non erano note a priori. Tuttavia, l'analisi del tipo di dato fornito dalle operazioni di lettura ha permesso di effettuare una considerazioni preliminare: l'istruzione `AD(x)`, che permette di leggere il valore di un determinato ingresso analogico x, restituisce infatti un valore di tipo `BYTE`; si è quindi potuto supporre che il valore numerico fornito in output fosse compreso nell'intervallo 0-255.

Si è poi proceduti alla ricerca vera e propria dei valori di fondo scala; in particolare si è cercato di identificare le seguenti quantità:

- il fondo scala inferiore, fornito in output quando il joystick è spostato totalmente a sinistra.
- il fondo scala superiore, ottenuto muovendo la leva a destra il più possibile.
- il valore mediano, fornito in output quando il joystick è in posizione centrale (a riposo).

Questi valori avrebbero potuto essere stimati misurando tramite tester le tensioni applicate all'ingresso ed eseguendo poi una opportuna proporzione, in modo da rapportarle al valore massimo fornito in output dall'ingresso (255). Tuttavia, per una stima corretta e precisa di queste quantità, si sarebbe comunque dovuto procedere ad un ulteriore controllo via software, in modo da compensare le eventuali imprecisioni della misurazione. Di conseguenza, per motivi puramente pratici, si è deciso di non effettuare alcuna misurazione fisica e di procedere direttamente alla stima dei valori tramite un'apposita routine.

I procedimenti seguiti per individuare i due valori di fondo scala sono molto simili; di seguito, quindi, descriveremo le modalità con cui abbiamo individuato il valore minimo, sottintendendo che l'attività per trovare il valore massimo è del tutto analoga. La ricerca si è svolta in questo modo: è stata definita una routine di test, attivabile con il telecomando, che effettua la lettura del valore ricevuto dal potenziometro; nel caso in cui il valore acquisito fosse inferiore ad una certa soglia, fissata a priori, essa avrebbe fatto eseguire al robot un movimento prestabilito, mentre, in caso contrario, il robot sarebbe rimasto totalmente immobile. La scelta dell'azione da far eseguire al robot è del tutto indifferente; l'obiettivo era infatti esclusivamente quello di ottenere un feedback sull'esito del controllo. In questo caso si è quindi deciso di utilizzare un movimento del braccio del robot come indicatore; tuttavia, per lo scopo che ci si proponeva, una qualsiasi altra azione (ad esempio, l'accensione della luce situata nella testa del robot) sarebbe risultata perfettamente equivalente. In seguito si è impostato il valore di soglia in modo che fosse pari ad un valore ipotetico di fondo scala sinistro, ad esempio 64; si è poi proceduti, tramite una ricerca binaria, a determinare il valore di fondo scala esatto. La procedura seguita consta dei seguenti passi:

1. spostare la leva del potenziometro totalmente verso sinistra, in modo da farle raggiungere la posizione di fondo scala
2. attivare la routine di test con il telecomando.
3. osservare la reazione del robot, modificando di conseguenza il valore di soglia.

La modifica avveniva secondo questo principio: il fatto che il robot si muovesse all'attivazione della routine significava che il valore fornito in output dal potenziometro era inferiore al fondo scala, situazione ovviamente impossibile; si poteva quindi dedurre logicamente che il fondo scala impostato fosse troppo alto, e si provvedeva a ridurlo. L'immobilità del robot, invece, permetteva di desumere che il valore del fondo scala fosse troppo ridotto, e che era necessario aumentarlo. Ripetendo iterativamente il procedimento sopra descritto si è arrivati ad individuare il valore di fondo scala sinistro, che è risultato essere pari a 118. Il valore di fondo scala destro è stato poi ottenuto secondo un procedimento del tutto simmetrico, ed è risultato essere pari a 166.

La ricerca del valore mediano è stata molto semplificata dal lavoro descritto al punto precedente: si è infatti potuto ipotizzare che tale valore fosse pari alla media dei due fondo scala. Supponendo quindi che esso fosse uguale a 142, si è provveduto a verificare la correttezza di tale congettura ripetendo la procedura prima descritta. Si è infine pensato di introdurre una piccola "zona morta", ovvero un intorno del valore mediano all'interno del quale il movimento del joystick non avrebbe dovuto causare alcuna reazione del robot; questo per evitare movimenti non desiderati dovuti a piccoli spostamenti accidentali della leva: il valore mediano è stato quindi "convertito" in due valori distinti: `centrosx`, che rappresenta il limite inferiore della zona morta ed è pari a 139, e `centrodx`, che invece ne costituisce il limite superiore ed è pari a 145.

È stata infine creata la routine software che permette di interagire con il potenziometro; il codice che la costituisce è riportato di seguito:

```
testa_pot:
  STERZO=AD(0)
  IF STERZO < CENTROSX OR STERZO > CENTRODX THEN
```

```
        GOSUB passo_fin
        GOSUB standard_pose
        GOSUB cambia_direzione
        WAIT
        GOTO inizia_cammino
ENDIF
RETURN
```

Come si può vedere essa procede per prima cosa ad effettuare una lettura del valore fornito dal potenziometro. Viene poi effettuato un controllo per verificare che tale valore sia esterno alla zona morta e, nel caso in cui ciò si verifichi, procede a riportare il robot in posizione eretta ed a invocare la routine `cambia_direzione`, che esegue la rotazione vera e propria.

3.4. Programma finale

Risolte le problematiche principali, si è proceduto a assemblare quanto creato durante le fasi precedenti all'interno del programma. A questo punto, effettuata l'esposizione del lavoro eseguito durante le varie fasi, è possibile comprendere pienamente il funzionamento della routine `cammina`, che costituisce il cuore del programma:

```
cammina:
  GOSUB testa_pot
  GOSUB passo_con
  GOSUB check_stop
  GOTO cammina
```

Come si può osservare essa all'inizio di ogni passo che costituisce il ciclo principale invoca la routine `testa_pot`, che si occupa sia di eseguire la lettura del valore fornito in ingresso dal potenziometro[2], sia di adeguare di conseguenza la direzione del robot. All'interno di `testa_pot`, infatti, viene invocata la routine `cambia_direzione`, il cui codice è riportato di seguito:

```
cambia_direzione:
  IF STERZO < CENTROSX THEN
        VARIAZIONE = CENTRO - STERZO
        GOSUB calcola_passi
        GOSUB gira_sx
        WAIT
  ELSEIF STERZO > CENTRODX THEN
        VARIAZIONE = STERZO - CENTRO
        GOSUB calcola_passi
        GOSUB gira_dx
        WAIT
        ENDIF
        WAIT

  RETURN
```

Essa calcola la variazione del valore di input ed effettua un controllo che permette di determinare se il joystick del potenziometro sia stato sposto a destra (caso STERZO > CENTRODX) o sinistra (caso STERZO < CENTROSX). È stato necessario implementare due confronti distinti a causa, ancora una volta, delle limitazioni del linguaggio Robobasic: esso, infatti, non supporta numeri negativi, rendendo di fatto impossibile l'esecuzione di un unico controllo basato sulla positività o negatività del valore della variazione. Viene poi eseguita la conversione per ottenere il numero di passi di rotazione che dovranno essere effettuati dal robot; questo valore viene calcolato nello stesso modo in entrambi i casi, visto che l'effettiva direzione di rotazione dipenderà dall'esito del controllo prima descritto. La rotazione viene poi materialmente effettuata dalle routine `gira_dx` e `gira_sx`, descritte nei paragrafi precedenti.

Ad alto livello, quindi, il funzionamento del programma principale si può riassumere in questo modo: la camminata viene avviata con il telecomando e procede indefinitamente fino a quando non viene arrestata, sempre tramite questa periferica. Durante il movimento è poi possibile modificare la direzione di marcia, spostando più o meno la leva del potenziometro nella direzione desiderata.

[2] il valore in ingresso viene assegnato alla variabile STERZO, come descritto nel paragrafo precedente.

4. Modalità operative

4.1. Componenti necessari

- Un calcolatore con sistema operativo Windows installato
- Kit ROBONOVA-1 (composizione descritta nel paragrafo introduttivo)
- Il file `elaborato.bas`, attualmente situato sul desktop dell'utente `smydor` del calcolatore `ROBOT`
- Potenziometro con il relativo cavo di collegamento

4.2. Modalità di installazione

Per quanto riguarda l'installazione del software necessario, la modalità di collegamento del robot al calcolatore, la creazione, compilazione e caricamento sul microcontrollore di un programma, si rimanda al manuale in dotazione.

Il potenziometro deve essere collegato all'ingresso analogico `AD0`. La massa va collegata al piedino più esterno, evidenziato dalla scritta "GND" sulla piastra madre, il filo di alimentazione al piedino centrale, mentre il filo che trasmette il segnale d'ingresso va connesso a quello più interno. Si noti inoltre che, nel caso in cui si desideri connettere il potenziometro ad un ingresso analogico differente rispetto a quello da noi utilizzato, sarà necessario modificare il programma in modo che l'istruzione di lettura del valore si riferisca all'ingresso opportuno.

4.3. Modalità di taratura

Il programma allo stato attuale non richiede alcuna taratura, a meno che non venga sostituito il potenziometro che consente di guidare il robot. In questo caso sarà necessario identificare nuovamente i valori di fondo scala dello strumento; tale procedura è stata descritta nel paragrafo 3.3. Si rimanda pertanto a questa sezione per istruzioni più dettagliate.

4.4. Modalità di esecuzione del programma

Le modalità di esecuzione del programma principale sono le seguenti

- Avviare il programma premendo il tasto "avanti" sul telecomando; in questo modo il robot inizia a camminare;
- Per modificare la direzione del robot, spostare il potenziometro a destra o a sinistra, sapendo che un escursione completa in una delle 2 direzioni comporta una rotazione di 90° nello stesso verso. Tenere il potenziometro in questa posizione fino a quando il robot non inizia a girare; questo perché la lettura avviene in istanti discreti (più specificatamente alla fine del `passo_con`) ed è pertanto necessario mantenere il valore in ingresso fino a quando non è avvenuta la sua acquisizione;
- Per fermare il robot, continuare a premere "stop" sul telecomando. Questo perché a lettura dei valori del telecomando avviene con le stesse modalità dell'acquisizione dell'ingresso del potenziometro; è pertanto necessario agire nello stesso modo.

4.5. Avvertenze

La messa in opera dell'elaborato richiede di prendere i seguenti accorgimenti:

- Durante la fase di caricamento del programma sul microcontrollore il robot rilascia tutti i servomotori; in questo lasso di tempo esso non è quindi in grado di mantenere la posizione eretta e tende perciò a cadere. Di conseguenza, in modo da evitare danni accidentali, si

consiglia di posizionarlo in maniera opportuna prima di programmarlo, o quantomeno di tenerlo ben saldo durante questa fase.

- Come si è accennato precedentemente, le caratteristiche del piano di appoggio influenzano di molto la stabilità del robot e l'efficacia della rotazione; è quindi opportuno assicurarsi che la superficie su cui avverrà il movimento sia il più possibile regolare e uniforme.

5. Conclusioni e sviluppi futuri

Il più naturale sviluppo di questo elaborato potrebbe consistere nel fornire il robot di una fotocellula differenziale, in modo che esso acquisti la capacità di camminare seguendo una sorgente luminosa mobile.

Un altro possibile sviluppo, ipotizzato durante il lavoro, consiste nel migliorare le capacità di pilotaggio del potenziometro utilizzato attualmente, in modo da poter regolare anche la velocità di camminata del robot. In particolare si è pensato di poter sfruttare l'asse Y del joystick, attualmente inutilizzato; in questo modo si fornirebbe la possibilità di poter controllare il robot tramite un unico dispositivo, in maniera più precisa, completa e naturale.

6. Appendice

In questa sezione si riporta per intero il codice del programma creato.

```
'================================================
' elaborato Robotica Mobile
'
' RR : internal parameter variable / ROBOREMOCON / Action command
' A: temporary variable/ REMOCON
' A16,A26 : temporary variable
'
'== auto_main ====================================
GOTO AUTO
FILL 255,10000

DIM RR AS BYTE
DIM A AS BYTE
DIM STERZO AS BYTE
DIM A16 AS BYTE
DIM A26 AS BYTE
DIM v AS BYTE
DIM VARIAZIONE AS BYTE
DIM PASSI AS BYTE
DIM ANGOLO AS INTEGER
DIM I AS BYTE
DIM TEMP AS INTEGER


CONST ID = 0 ' 1:0, 2:32, 3:64, 4:96,
CONST POT = 24
CONST CENTRO = 142
CONST CENTROSX= 139
CONST CENTRODX= 145

'== Action command check (50 - 82)
IF RR > 50 AND RR < 83 THEN GOTO action_proc

RR = 0

PTP SETON
PTP ALLON

'== motor diretion setting ======================
DIR G6A,1,0,0,1,0,0
```

```
DIR G6B,1,1,1,1,1,1
DIR G6C,0,0,0,0,0,0
DIR G6D,0,1,1,0,1,0


'== motor start position read ===================
TEMPO 230
MUSIC "CDE"
GETMOTORSET G24,1,1,1,1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,1,1,1,1,1,0
'== motor power on=============================
SPEED 5
MOTOR G24
GOSUB standard_pose
'================================================
MAIN:
'GOSUB robot_voltage
'GOSUB robot_tilt
IF A16 <> 0 THEN GOTO salta

  v=v-1
  IF v <> 0 THEN GOTO salta
  v=50
  OUT 52,0
  DELAY 200
  OUT 52,1

salta:

'------------------------------
IF RR = 0 THEN GOTO MAIN1

ON RR GOTO
MAIN,nop,nop,nop,nop,nop,nop,nop,nop,nop,nop,inizia_cammino,nop,nop,nop,nop,inter
ruttore
GOTO main_exit
'------------------------------
MAIN1:


A = REMOCON(1)
A = A - ID

ON A GOTO
MAIN,nop,nop,nop,nop,nop,nop,nop,nop,nop,nop,inizia_cammino,nop,nop,nop,nop,inter
ruttore
GOTO MAIN
'------------------------------------------------
action_proc:
A = RR - 50
ON A GOTO
MAIN,nop,nop,nop,nop,nop,nop,nop,nop,nop,nop,inizia_cammino,nop,nop,nop,nop,inter
ruttore
RETURN
'------------------------------
main_exit:
  IF RR > 50 THEN RETURN
  RR = 0
  GOTO MAIN


'================================================
'================================================
'================================================


nop: GOTO main_exit


inizia_cammino:
  GOSUB passo_in
  GOTO cammina
```

```
'controllo se è stato premuto il tasto di stop sul telecomando
check_stop:
  A = REMOCON(1)
  IF A=26 THEN
          GOSUB passo_fin
          GOSUB standard_pose
          GOTO main_exit
  ENDIF
  RETURN


'loop principale: esegue una sequenza di passi continui
cammina:
  GOSUB testa_pot
  GOSUB passo_con
  GOSUB check_stop
  GOTO cammina


'legge il valore di ingresso del potenziometro ed eventualmente
'esegue il cambio di direzione
testa_pot:
  STERZO=AD(0)
  IF STERZO < CENTROSX OR STERZO > CENTRODX THEN
          GOSUB passo_fin
          GOSUB standard_pose
          GOSUB cambia_direzione
          WAIT
          GOTO inizia_cammino
  ENDIF
  RETURN


cambia_direzione:
  IF STERZO < CENTROSX THEN
          VARIAZIONE = CENTRO - STERZO
          GOSUB calcola_passi
          GOSUB gira_sx
          WAIT
  ELSEIF STERZO > CENTRODX THEN
          VARIAZIONE = STERZO - CENTRO
          GOSUB calcola_passi
          GOSUB gira_dx
          WAIT
          ENDIF
          WAIT

  RETURN


'converte la variazione del valore di input del potenziometro
'nel numero di passi che costituiranno la rotazione
calcola_passi:
  TEMP = VARIAZIONE * 90
  ANGOLO = TEMP /POT
  PASSI = ANGOLO / 15
  WAIT
  RETURN


gira_sx:
  FOR I = 1 TO PASSI
          GOSUB left_turn
          WAIT
          GOSUB standard_pose
          WAIT
  NEXT I
  PASSI = 0
  WAIT
  RETURN


gira_dx:
```

```
  FOR I = 1 TO PASSI
         GOSUB right_turn
         WAIT
         GOSUB standard_pose
         WAIT
  NEXT I
  PASSI = 0
  RETURN


interruttore:
  GOSUB sit_down_pose16
  GOTO main_exit


'================================================

robot_voltage:                  ' [ 10 x Value / 256 = Voltage]


  A = AD(6)

  IF A < 148 THEN                                ' 5.8v

  FOR v = 0 TO 2
  OUT 52,1
  DELAY 200
  OUT 52,0
  DELAY 200
  NEXT v

  RETURN


standard_pose16:
  OUT 52,0
  TEMPO 230
  MUSIC "CDE"
  GETMOTORSET G24,1,1,1,1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,1,1,1,1,1,0
  'motor power on
  MOTOR G24
  A16 = 1
  SPEED 10
  GOSUB standard_pose
  RETURN


sit_down_pose16:
  IF A16 = 0 THEN GOTO standard_pose16
  A16 = 0
  SPEED 10
  MOVE G6A,100,151,23,140,101,100
  MOVE G6D,100,151,23,140,101,100
  MOVE G6B,100,30,80,100,100,100
  MOVE G6C,100,30,80,100,100,100
  WAIT
  'motor power off
  MOTOROFF G24
  TEMPO 230
  MUSIC "FEDC"
  RETURN


standard_pose:
  MOVE G6A,100,76,145,93,100,100
  MOVE G6D,100,76,145,93,100,100
  MOVE G6B,100,30,80,100,100,100
  MOVE G6C,100,30,80,100,100,100
  WAIT
  RETURN


left_turn:
```

```
  SPEED 6
  MOVE G6D,85,71,152,91,112,60
  MOVE G6A,112,76,145,93,92,60
  MOVE G6C,100,40,80,,,,
  MOVE G6B,100,40,80,,,,
  WAIT

  SPEED 9
  MOVE G6A,113,75,145,97,93,60
  MOVE G6D,90,50,157,115,112,60
  MOVE G6B,105,40,70,,,,
  MOVE G6C,90,40,70,,,,
  WAIT

  MOVE G6A,108,78,145,98,93,60
  MOVE G6D,95,43,169,110,110,60
  MOVE G6B,105,40,70,,,,
  MOVE G6C,80,40,70,,,,
  WAIT
  RETURN


right_turn:

  'PIEGA GAMBE
  SPEED 6
  MOVE G6A,85,71,152,91,112,60
  MOVE G6D,112,76,145,93,92,60
  MOVE G6B,100,40,80,,,,
  MOVE G6C,100,40,80,,,,
  WAIT

  'AVANTI GAMBA SX
  SPEED 9

  'gamba dx
  MOVE G6D,113,75,145,97,90,60 '(riga originale)

  'gamba sx
  MOVE G6A,90,50,157,115,112,60 '(riga originale)

  MOVE G6C,105,40,70,,,,
  MOVE G6B,90,40,70,,,,
  WAIT

  'INDIETRO GAMBA SX
  'gamba dx
  MOVE G6D,108,82,145,98,93,60

  MOVE G6A,95,43,169,110,110,60
  MOVE G6C,105,40,70,,,,
  MOVE G6B,80,40,70,,,,
  WAIT
  RETURN


'partendo dalla standard pose,sposta il piede sinistro in avanti
'per cominciare il movimento di camminata
passo_in:
          SPEED 5
  'spostare il peso a destra
  MOVE2485,71,152,91,112,60,100,40,80,,,,100,40,80,,,,112,76,145,93,92,60,

          SPEED 10
  'sollevare il piede sinistro
  MOVE2490,107,105,105,114,60,90,40,80,,,,100,40,80,,,,114,76,145,93,90,60,

  'portare in avanti il piede sinistro
  MOVE2490,56,143,122,114,60,80,40,80,,,,105,40,80,,,,113,80,145,90,90,60,
  MOVE2490,46,163,112,114,60,80,40,80,,,,105,40,80,,,,112,80,145,90,90,60,

  'appoggiare a terra il piede sinistro
  MOVE G6A,91,42,163,113,112,60
```

```
  MOVE G6B,80,40,80,100,100,100
  MOVE G6C,105,40,80,100,100,100
  MOVE G6D,110,84,138,95,87,60
  WAIT
  RETURN


'movimento pensato per terminare la camminata
'partendo con il piede sinistro in avanti
'viene spostato in avanti il destro in modo
'da tornare nella standard pose
passo_fin:
  SPEED 4

  'portare il peso sulla gamba sinistra
  MOVE G6A,109,46,162,110,91,60
  MOVE G6D,94,89,131,98,107,60
  SPEED 6
  MOVE G6B,123,40,80,100,100,100
  MOVE G6C,80,45,82,100,100,100
  WAIT

  SPEED 9

  'portare in avanti il piede destro
  MOVE G6A,110,48,174,102,101,60
  MOVE G6B,124,55,80,100,100,100
  MOVE G6C,80,45,82,100,100,100
  MOVE G6D,93,76,131,117,104,60
  WAIT

  'posizione eretta (standard pose)
  MOVE G6A,100,76,145,93,100,100
  MOVE G6D,100,76,145,93,100,100
  MOVE G6B,100,30,80,100,100,100
  MOVE G6C,100,30,80,100,100,100
  WAIT
  RETURN


'movimento pensato per una sequenza continua di passi
'partendo con il piede sinistro in avanti (passo_in)
'sposta in avanti il piede destro e poi quello sinistro
passo_con:
  SPEED 12
  'avanzare facendo leva sul piede sinistro
  MOVE24 100,66,141,113,100,100,90,40,80,,,,100,40,80,,,,100,83,156,80,100,100,
  MOVE24 113,78,142,105,90,60,100,40,80,,,,100,40,80,,,,90,102,136,85,114,60,


  SPEED 16
  'alzare il piede destro
  MOVE24 113,76,145,93,90,60,100,40,80,,,,90,40,80,,,,90,107,105,105,114,60,
  'spostare il piede destro in avanti
  MOVE24 113,80,145,90,90,60,105,40,80,,,,80,40,80,,,,90,56,143,122,114,60,
  MOVE24 112,80,145,90,90,60,105,40,80,,,,80,40,80,,,,90,46,163,112,114,60,

  SPEED 12
  'avanzare facendo leva sul piede destro
  MOVE24 100,83,156,80,100,100,100,40,80,,,,90,40,80,,,,100,66,141,113,100,100,
  MOVE2490,102,136,85,114,60,100,40,80,,,,100,40,80,,,,113,78,142,105,90,60,

  SPEED 16
  'alzare il piede sinistrfo
  MOVE2490,107,105,105,114,60,90,40,80,,,,100,40,80,,,,114,76,145,93,90,60,

'portare in avanti il piede sinistro
  MOVE2490,56,143,122,114,60,80,40,80,,,,105,40,80,,,,113,80,145,90,90,60,
  MOVE2490,46,163,112,114,60,80,40,80,,,,105,40,80,,,,112,80,145,90,90,60,

  ' appoggiare il piede sinistro a terra
  MOVE G6A,91,42,163,113,112,60
  MOVE G6B,80,40,80,100,100,100
```

```
MOVE G6C,105,40,80,100,100,100
MOVE G6D,110,84,138,95,87,60
WAIT
RETURN
```

Bibliografia

[1] ROBONOVA-1 English User Manual(Version 1.00 20051115)

[2] Robobasic English Command Instruction Manual(Version 2.10 20051115)

Indice