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

Laboratorio di Robotica Avanzata
Advanced Robotics Laboratory

Corso di Robotica
(Prof. Riccardo Cassinis)

Sensorizzazione di Gongolo

Elaborato di esame di: **Ivan Pareja Larios, Anna Bellini**

Consegnato il: **18 giugno 2003**

Sommario

Il presente elaborato è una continuazione del lavoro iniziato nel "Progetto Neogongolo".

Nel progetto Neogongolo, realizzato da Marco Merigo, Luca Alberici e Nicola Gatta è stata sviluppata una scheda per il microcontrollore Motorola 68HC11 ed il software per muovere il robot tramite i suoi motori a passo.

Ma il grado di sensorizzazione di Gongolo era abbasastanza basso, infatti Gongolo aveva solo due sensori di contatto.

Lo scopo del nostro progetto è quello di aggiungere a Gongolo 5 sensori di distanza e sviluppare il software necessario per l'acquisizione ed elaborazione dei dati provenienti da essi.

Inoltre Gongolo sarà programmato per trovare una parate e seguirla ad una distanza di circa 30 centimetri.

Ma l'aggiunta dei 5 sensori di distanza consentirà di ricevere molte più informazioni sul mondo esterno, permettendo lo sviluppo futuro di comportamenti molto più articolati.

1. Introduzione

1.1. Caratteristiche di Gongolo

Gongolo era un robot in grado di muoversi, mediante ruote, in ambienti con ostacoli, era dotato di un sistema sensoriale a "baffi" che permetteva di percepire un ostacolo prima di una possibile collisione e di un sistema di controllo con il quale era possibile prendere un'iniziativa adeguata per evitare un ostacolo frontale.

Il sistema di controllo era implementato mediante il microcontrollore della Motorola 68HC11. Il microcontrollore generava un segnale ad onda quadra in grado di pilotare i motori a passo per il movimento rettilineo del robot. Per la partenza da fermo era prevista una rampa di accelerazione (onde quadre a periodo uniformemente decrescente) che permetteva ai motori di vincere l'inerzia del robot. I motori a passo, infatti, necessitano di rampe di accelerazione per partire da posizione stazionaria.

Il sistema sensoriale a "baffi" era costituito da due fili di ferro ricurvi posti davanti al robot, in modo tale da ricoprirne la parte frontale e connessi a un sensore di vibrazione, collegato ad un circuito elettronico, il quale era in grado di fornire un segnale logico, in corrispondenza di un urto, e quindi di una leggera vibrazione (ad alta frequenza), da parte del filo di ferro. In risposta al segnale proveniente da uno dei due baffi anteriori (sintomo della presenza di uno o più ostacoli) il microcontrollore 68HC11 faceva indietreggiare il robot modificando il bit di direzione, in ingresso agli integrati di interfaccia con i motori, successivamente prevedeva una manovra di rotazione assiale di circa 60 gradi e una ripartenza.

Mantenendo inalterate tutte le precedenti caratteristiche di Gongolo, sono stati aggiunti cinque sensori di distanza che consentono al robot di trovare e seguire una parete.

1.2. Struttura della relazione

Lo schema con cui è scritta questa relazione è molto semplice.

Nel capitolo successivo è descritto il problema affrontato che costituisce l'obiettivo del progetto. In questo capitolo sono elencate le specifiche richieste per la modifica del sistema di controllo del robot e non si parla, volutamente, della soluzione adottata.

La descrizione del metodo risolutivo usato costituisce l'oggetto del terzo capitolo.

Il quarto capitolo è invece un manuale d'uso per l'utente, ovvero una descrizione dettagliata di come valutare i risultati ottenuti dal progetto.

Una sintesi dei possibili sviluppi futuri è raccolta nel capitolo conclusivo di questo elaborato.

2. Il problema affrontato

Il problema da risolvere consisteva, come già anticipato nel sommario, nel dotare Gongolo di 5 sensori di distanza per renderlo capace di comportamenti più autonomi ed intelligenti.

Ciò implica la risoluzione di vari problemi:

- In primo luogo si deve scegliere i sensori più adatti al nostro scopo, valutando la vasta gamma di sensori che esistono oggi sul mercato. Per la scelta si deve tenere conto di diverse caratteristiche: del **microcontrollore** (numero di ingressi analogici o digitali liberi, memoria, etc), dei **sensori** (range di distanza che vogliamo misurare, metodo di funzionamento, tipo di output fornito, etc).
- In secondo luogo si deve affrontare il problema del montaggio dei sensori su Gongolo, che consiste nel collegamento elettrico dei sensori alla scheda del microcontrollore e nel dover fornire i sensori di un adeguata alimentazione.
- Infine si deve affrontare il problema del software, che consiste nello sviluppo di una routine per l'acquisizione dei dati e nello sviluppo di un programma per seguire una parete a una distanza fissa. Le funzioni per ricevere informazioni dai sensori di contatto e la routine di generazione delle onde quadre a fase variabile per le rampe di accelerazione dei motori a passo, saranno quelle sviluppate nel "Progetto Neogongolo".

3. La soluzione adottata

3.1. Scelta Sensori

Attualmente esiste sul mercato una vastissima gamma di sensori in grado di misurare la distanza, questi sensori sono basati su diversi principi fisici di funzionamento (intensità del fascio riflesso, triangolazione, interferenza, tempo di volo, etc).

Ma date le diverse circostanze (come ad esempio: l'ambiente in cui deve muoversi il robot o la relazione qualità prezzo dei sensori) concentreremo la nostra attenzione sulla famiglia di sensori a infrarossi della SHARP. Il loro funzionamento è basato sul fenomeno della triangolazione. In linea di massima hanno un diodo che emette un fascio di luce infrarossa, un foto detector e diversi circuiti di elaborazione del segnale. A seconda dell'angolo con cui è riflesso il fascio di luce dall'oggetto, si misura la distanza dal sensore.

I diversi modelli di sensori SHARP sono:

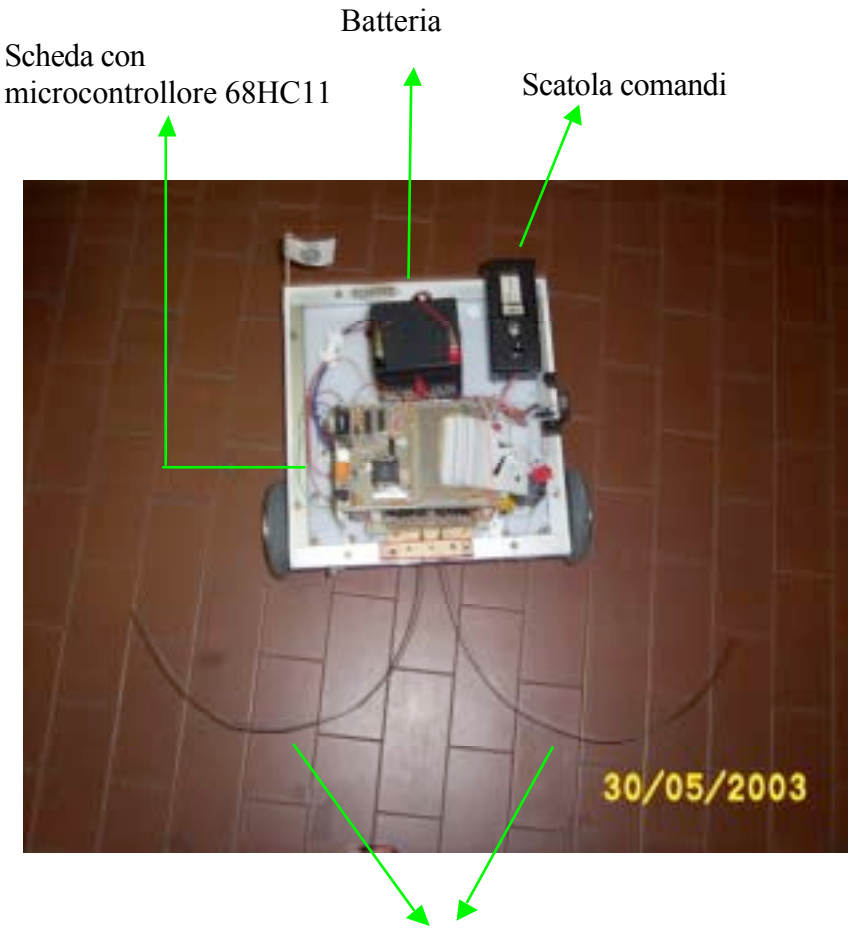
Modello	Output	Minima distanza (cm)	Massima distanza (cm)
GP2D02	Serial	10	80
GP2D05	Digitale	-	fissa a 24
GP2D12	Analogica	10	80
GP2D15	Digitale	-	fissa a 24 +/- 3 cm
GP2D120	Analogica	4	30
GP2Y0A02YK	Analogica	20	150
GP2Y0D02YK	Digitale	-	fissa a 80

I modelli più adatti al nostro progetto sono il GP2D12, GP2D120, GP2Y0A02YK perché forniscono buone prestazione a bassi costi. Il microcontrollore Motorola 68HC11 è fornito di una porta di ingresso (PortE) con otto linee, configurabile come analogiche o digitali, quindi non occorre nessun tipo di circuito esterno per il loro uso. La scelta finale cadrà sul modello **GP2D12** perché il range di distanza misurato (10 - 80 cm) è il più adeguato all'ambiente che circonda al robot (il Laboratorio di Robotica Avanzata della Facoltà di Ingegneria della università di Brescia).

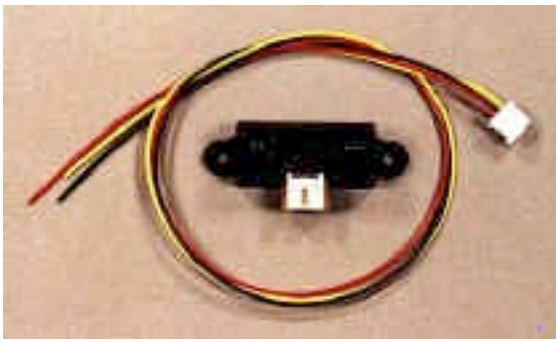
I rimanenti modelli risultano essere meno appropriati al nostro scopo. I sensori GP2D05, GPD2D15 e GP2Y0D02YK perché sono semplicemente sensori di presenza (rivelano la presenza di un ostacolo a una distanza fissa, fornendo in uscita un valore digitale). Il GP2D02 perché l'acquisizione del segnale fornito in uscita (di tipo seriale) è più complessa;e inoltre necessita di un segnale di clock all'ingresso e di una circuiteria esterna.

3.2. Montaggio Sensori

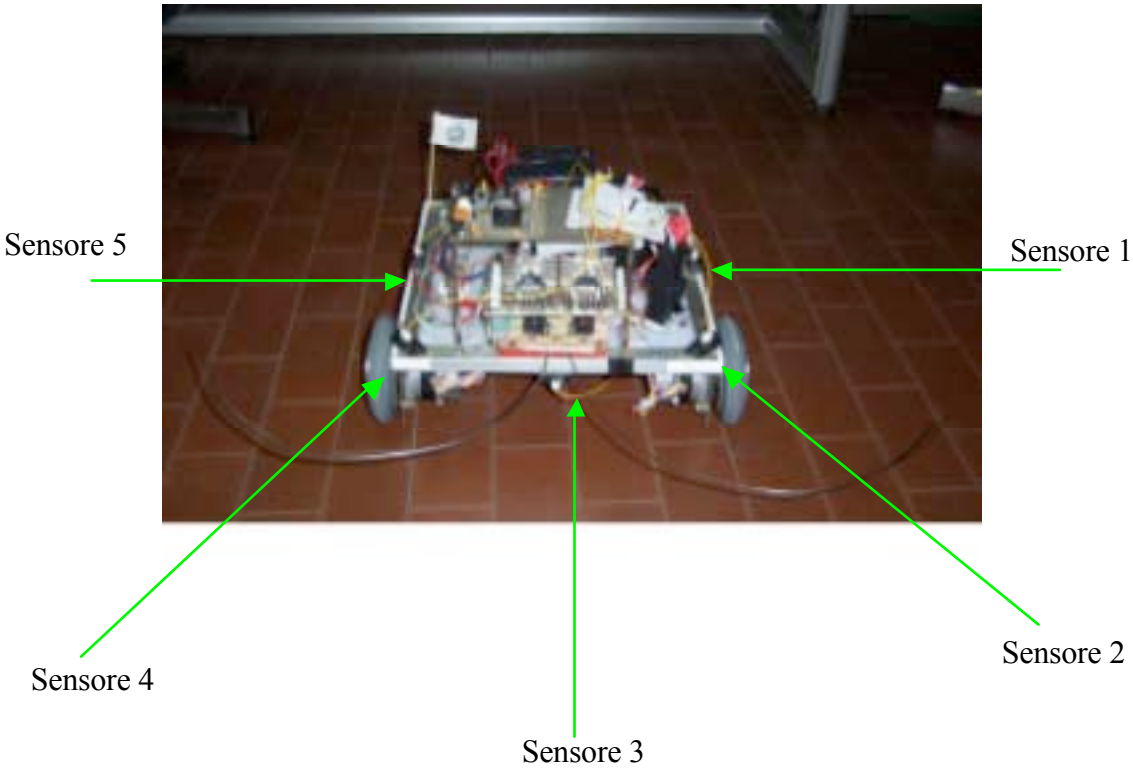
Gongolo prima di essere dotato di sensori



Sensori di distanza GP2D12



Gongolo dopo essere stato dotato di sensori



Sono stati montati cinque sensori su Gongolo con dello scotch biadesivo spesso un millimetro, per non vincolare eventuali modifiche future.

Ogni sensore era dotato di tre file: alimentazione (rosso), massa (nero) e linea di uscita (giallo).

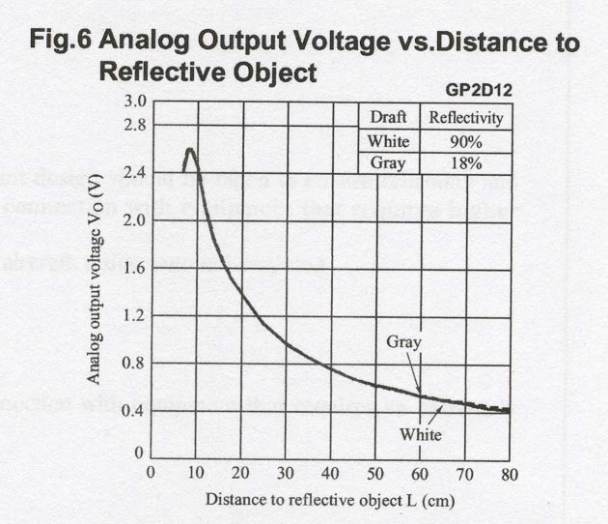
I cinque fili di massa sono stati connessi alla massa (0 Volt) della batteria già presente su Gongolo.

Le alimentazioni dei sensori sono state connesse alla batteria di Gongolo tramite il chip 7805 che riceve in ingresso 12 Volt e restituisce 5 Volt in uscita.

Le cinque uscite sono state collegate ai primi cinque ingressi analogici della porta E.

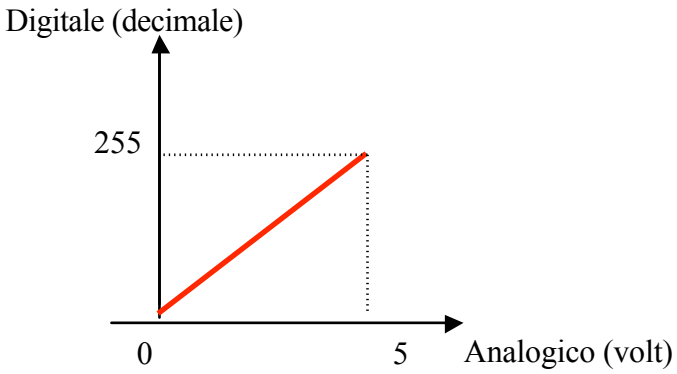
3.3. Acquisizione dei dati dai sensori

Il sensore rileva la distanza dell'oggetto e la converte in un valore analogico, come si può vedere dal seguente **grafico**[1]:



Poi tramite il microcontrollore 68HC11 è possibile effettuare la conversione da analogico a digitale.

La conversione da analogico a digitale segue un andamento lineare, come si può vedere dal seguente grafico:



[1] **Appendice C**: Manuale del sensore GP2D12

Il valore in uscita dai sensori dipende, anche da altri fattori, come per esempio: la riflessione di luce dell'oggetto, la temperatura dell'ambiente in cui avviene la misurazione, etc.

Per una maggiore garanzia abbiamo personalmente testato i valori in uscita dai sensori, costruendo la seguente tabella:

Distanza (cm)	Valore medio (esadecimale)	Valore mimino (esadecimale)	Valore massimo (esadecimale)
10	71	70	73
15	55	52	57
20	42	40	44
25	35	32	39
30	2D	2B	30
35	27	25	2A
40	22	1F	25
45	1E	1A	20
50	1C	19	1F
60	15	10	19
70	11	07	15
80	08	04	12

3.4. Programmazione

Il sorgente in formato .**asc** è stato compilato mediante il programma assemblatore per 68HC11 producendo 2 tipi di file: .**lst** e .**s19**.

Il sorgente compilato, che ha una dimensione di 1,684 Kb, è stato scritto nella EEPROM del microcontrollore mediante il programma di monitor-debugger PCBUG11 della Motorola.

La capienza massima della EEPROM è di 2Kb, rimangono quindi 0,316Kb liberi.

Le routine riportate nel codice allegato sono, di conseguenza, usate dal sistema di controllo del robot. Per averne una dimostrazione si legga preventivamente il capitolo successivo (modalità operative).

Nella routine "Cerca" è di rilevante importanza la variabile Trovato, nel caso in cui assuma valore 0, il robot è in mezzo ad un ambiente senza pareti, se trovato assume valore 1 il robot sta seguendo una parete alla sua sinistra, infine se trovato è uguale a 2 il robot sta seguendo una parete alla sua destra.

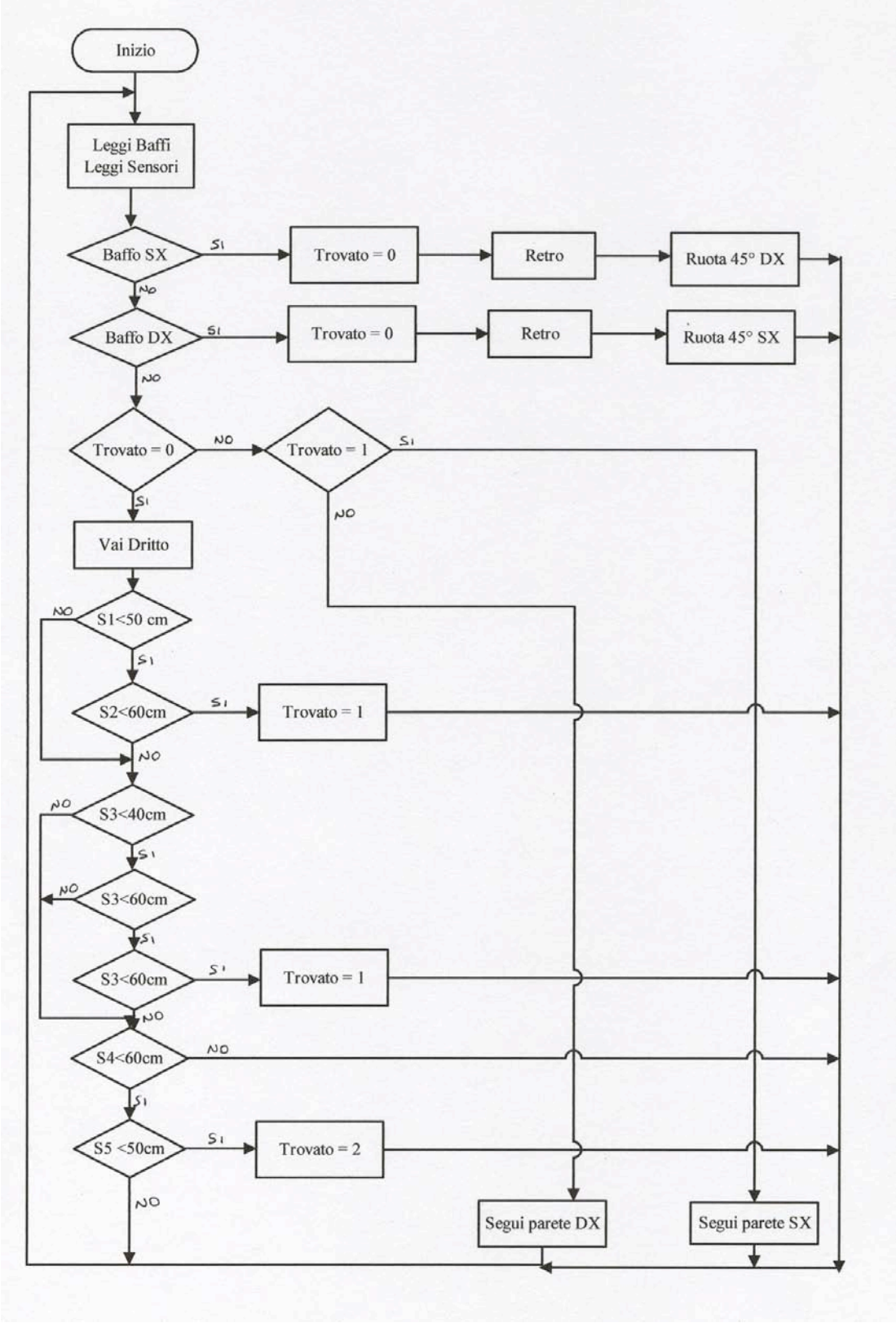
Trovato = 0 → no parete

Trovato = 1 → parete SX

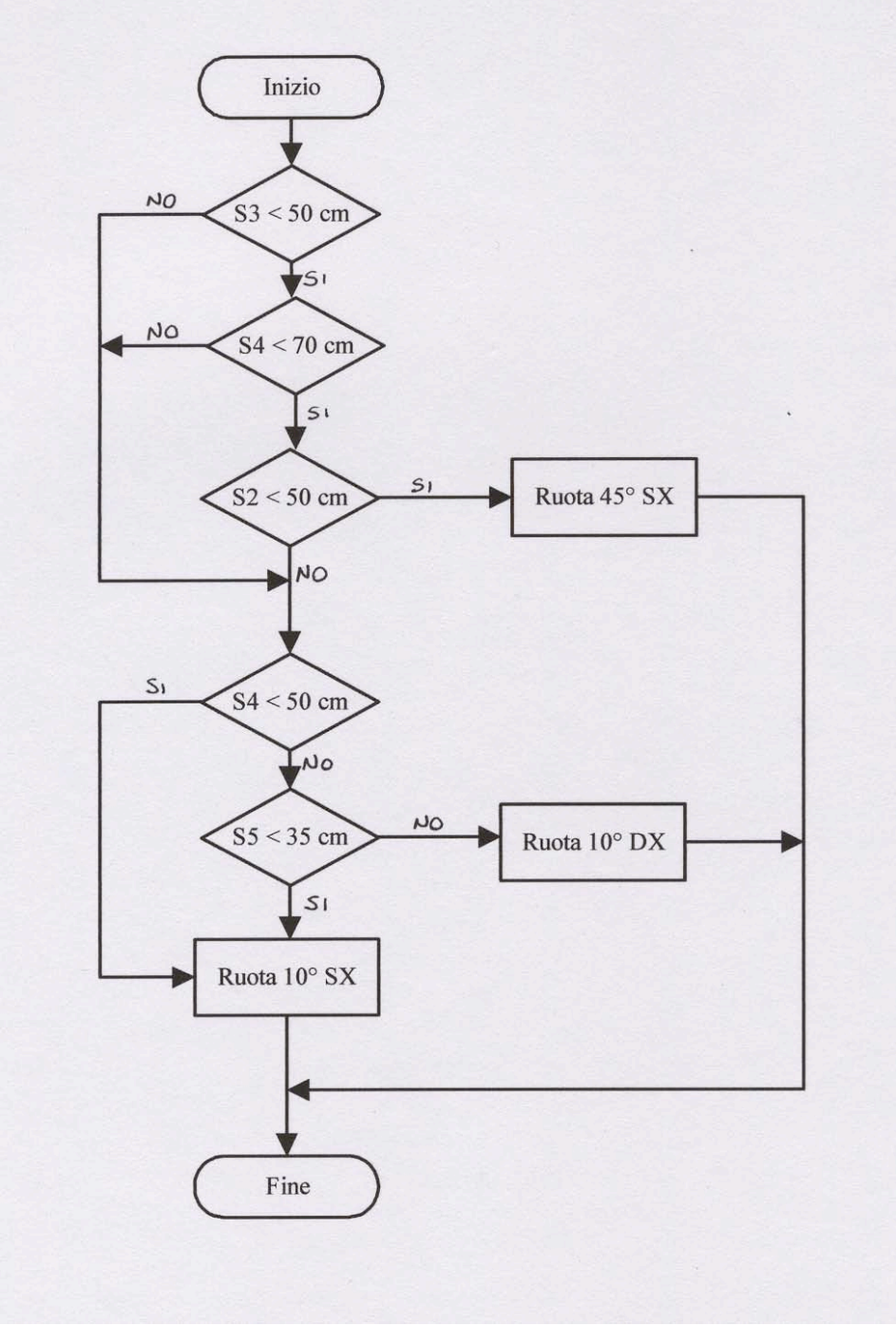
Trovato = 2 → parete DX

Segue l'immagine flow-char che descrive il comportamento del robot.

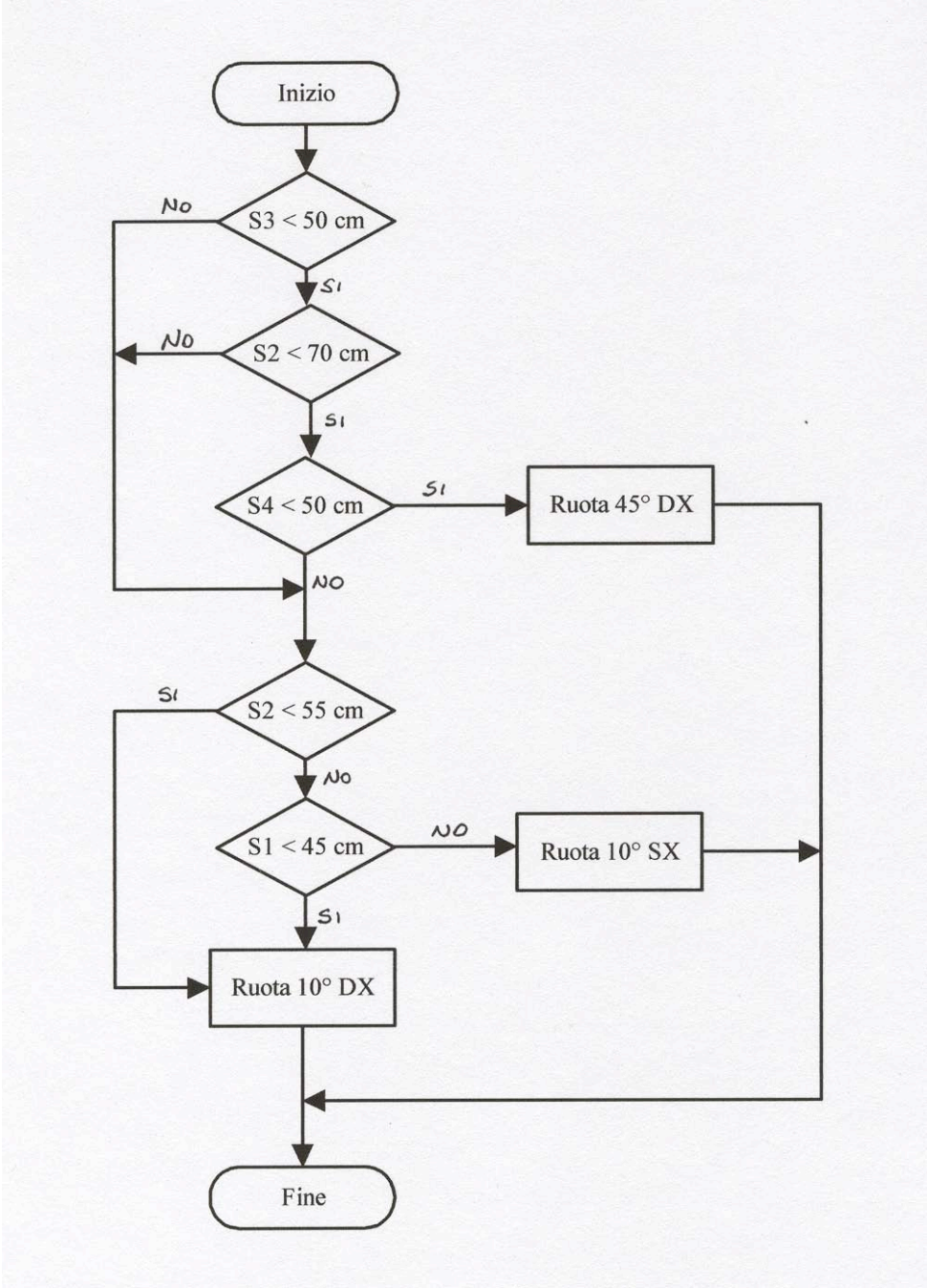
Routine per cercare parete evitando ostacoli



Routine “segui parete DX”



Routine “segui parete SX”



3.5. Sorgente del programma per pilotare Gongolo

```
************************************************************************
************************************************************************
****** programma di pilotaggio di gongolo: segue parete ************
************************************************************************
************************************************************************


* definisce i periodi minimo e massimo per i motori a passo
PERMIN  EQU     3000            * periodo minimo (vel di regime)
PERMAX  EQU     7000            * periodom massimo (vel partenza)
ACCEL   EQU     110             * accelera in  (PERMAX-PERMIN) /
                                * (200 * ACCEL) secondi



* definisce gli scostamenti dal Register Block
PORTA   EQU     $00
PORTC   EQU     $03
PORTCL  EQU     $05
PORTB   EQU     $04
DDRC    EQU     $07
OPTION  EQU     $39
ADCTL   EQU     $30
ADR1    EQU     $31
ADR2    EQU     $32
ADR3    EQU     $33
ADR4    EQU     $34
TOC2    EQU     $18
TOC3    EQU     $1A
TOC5    EQU     $1E
TCTL1   EQU     $20
TCTL2   EQU     $21
TFLG1   EQU     $23
TMSK1   EQU     $22


* definisce gli indirizzi base di RAM, EEPROM, INT TABLE, RESET
stacktop EQU    $ff
RAM     EQU     $00
EEPROM  EQU     $F800

VETOC   EQU     $FFE4
RESET   EQU     $FFFE


* definisce le variabili in memoria RAM
      ORG     RAM

per2    RMB     2               * periodo motore 1
per3    RMB     2               * periodo motore 2
baffo   RMB     1               * per registrare l'informazione del
                                * port C
dir     RMB     1               * direzione dei motori
inc     RMB     2               * si usa per accelerare i motori
```

```
* sensori di distanza
s1      RMB     1           * sensore 1
s2      RMB     1           * sensore 2
s3      RMB     1           * sensore 3
s4      RMB     1           * sensore 4
s5      RMB     1           * sensore 5

trov    RMB     1           * trov = 0 --> no parete
                            * trov = 1 --> parete SX
                            * trov = 2 --> parete DX
cont    RMB     2



* Scrive nella Tabella degli interrupt gli indirizzi delle routine
* da attivare per gli interrupt TOC2, TOC3 e RESET
      ORG     VETOC
      FDB     #isr_oc3
      FDB     #isr_oc2

      ORG     RESET
      FDB     $F800


* inizio programma scritto in EEPROM
      ORG EEPROM
      lds     #stacktop
      ldx     #$1000

      ldaa    #$00              * no ENABLE, avanti per tutti
      staa    dir
      staa    PORTB,x           * inizialliza il port B

      staa    DDRC,x            * setta il port C come ingresso
      staa    PORTC,x           * inizialliza il port C

      ldd    #ACCEL             * inizializa inc
      std    inc



* configura il conversore analogico digitale
      ldaa    #$80
      staa    OPTION,x

      ldd     #PERMAX          * fa partire con il periodo massimo
      std     per2
      std     per3
      std     TOC2,x
      std     TOC3,x

* configura gli Output Compare e le interruzioni associate
      ldaa    #%01010001
      staa    TCTL1,x
      ldaa    #%01100000
      staa    TFLG1,x
      ldaa    #%01100000
      staa    TMSK1,x
```

```
* abilita interruzioni
      cli

      * configura Output Compare 5 che temporizer… la chiamata
      * alla funzione muovi
      ldd #2500
      std TOC5,X

      ldaa    #$00                * setta il port C come ingresso
      staa    DDRC,x

      ldaa    #%00000100          * attiva ENABLE dei motori
      staa    dir
      staa    PORTB,x



* inizializza variabili

      clr  trov
      clr  baffo
      clr  s1
      clr  s2
      clr  s3
      clr  s4
      clr  s5


*********************************************************************
********************** Programma Principale ************************
*********************************************************************
main
      jsr    legge
      jsr     chk_oc5
      bra     main

coc5_end
      rts



*********************************************************************
************ Funzione che temporizza la routine Muovi***************
*********************************************************************
chk_oc5

      brclr TFLG1,x,#$08,coc5_end

      ldd TOC5,x
      addd #2500
      std TOC5,x

      ldaa $08
      staa TFLG1,x

      jsr muovi
```

```
    rts


*********************************************************************
*************** Funzione che fa muovere il robor ******************
*********************************************************************
muovi

       *jsr legge
      ldaa baffo
      bita #%00000001         * testa se un baffo ha toccato
                              * un ostacolo
      beq  muovi1
      jsr sp_led
      clr trov
      jsr retro               * se baffo SX attivo e va indietro
      jsr desbaf              * e ruota di 45ø a DX
      bra fine

muovi1
      bita #%00000010         * testa se il baffo DX ha toccato
                              * un ostacolo
      beq  muovi2
      jsr sp_led
      clr trov
      jsr retro               * se baffo DX attivo e va indietro
      jsr sinbaf              * e ruota di 45ø a SX

      bra  fine
                              * se non ci sono ostacoli cerca e
                              * segue una parete
muovi2
      ldaa trov
      bne  muovi3
      jsr avanti              * se trov = 0 --> va avanti
      jsr  cerca
      bra fine
muovi3
      ldaa  trov
      cmpa #$01
      bne muovi4
      jsr seguesx             * se trov = 1 --> segue parete SX
      bra fine
muovi4
      jsr seguedx             * se trov = 2 --> segue parete DX

fine
      rts


*********************************************************************
****************** Funzione che cerca una parete ********************
*********************************************************************
cerca

      ldaa s2
      cmpa  #30           * confronta s2 con 50 cm
      ble   cerca2
```

```
cerca1
      ldaa s1
      cmpa  #25          * confronta s1 con 60 cm
      ble  cerca2
      ldaa #$01          * Se s1 e s2 rilavano una parete
      staa trov          * allora trov = 1
      jsr  des45
      bra  fine4

cerca2
      ldaa s3
      cmpa  #38          * confronta s3 con 40 cm
      ble  cerca3
      ldaa  s2
      cmpa #25           * confronta s2 con 60 cm
      ble  cerca3
      ldaa  s4
      cmpa #25           * confronta s4 con 60 cm
      ble  cerca3
      jsr   sin45
      ldaa  #$02         * Se s3, s2 e s4 rilavano una parete
      staa  trov         * allora trov = 2
      bra   fine4
cerca3
      ldaa s4
      cmpa  #25          * confronta s4 con 60 cm
      bge  cerca4
      bra  fine4
cerca4
      ldaa s5
      cmpa  #30          * confronta s5 con 50 cm
      ble  fine4

      ldaa #$02          * Se s4 e s5 rilavano una parete
      staa trov          * allora trov = 2
fine4


      rts


*********************************************************************
*************** Funzione che segue la parete DX ********************
*********************************************************************
seguedx

       ldaa s3
       cmpa #31            * confronta s3 con 50 cm
       ble  dx1
       ldaa s4
       cmpa #23            * confronta s4 con 70 cm
       ble dx1
       ldaa s2
       cmpa #30            * confronta s2 con 50 cm
       ble  dx1
       jsr sin45           * Se s3, s4 e s2 rilavano una parete
```

```
                              * il robot gira 45ø a SX
       bra fine3



dx1
       ldaa s4
       cmpa #31               * confronta s4 con 50 cm
       ble dx2
       bra dx3

dx2
       ldaa s5
       cmpa #43               * confronta s5 con 35 cm
       bge dx3
       jsr des10              * Se s4 e s5 sono lontani dalla parete
                              * il robot si avvicina giando 10ø a DX
       bra fine3

dx3
       jsr sin10              * Se s4 o s5 sono vicini alla parete
                              * il robot si allontana giando 10ø a Sx

fine3

      rts


*********************************************************************
************** Funzione che segue la parete SX ******************
*********************************************************************
seguesx

       ldaa s3
       cmpa #31                * confronta s3 con 50 cm
       ble  sx1
       ldaa s2
       cmpa #23                * confronta s2 con 70 cm
       ble sx1
       ldaa s4
       cmpa #30                * confronta s4 con 50 cm
       ble sx1
       jsr des45               * Se s3, s2 e s4 rilavano una parete
                               * il robot gira 45ø a DX
       bra fine2


sx1

       ldaa s2
       cmpa #27               * confronta s2 con 55 cm
       ble sx2
       bra sx3

sx2
       ldaa s1
       cmpa #34               * confronta s1 con 45 cm
       bge sx3

       jsr sin10              * Se s2 e s1 sono lontani dalla parete
```

```
                              * il robot si avvicina giando 10ø a SX

       bra fine2

sx3
       jsr des10              * Se s1 o s2 sono vicini alla parete
                              * il robot si allontana giando 10ø a DX

fine2
      rts


*********************************************************************
************ Funzione che fa andara avanti il robot ****************
*********************************************************************
avanti

      ldaa    #%00000100
      staa    PORTB,x
      ldd     #PERMIN          * controlla se per2 e' maggiore di
      cpd     per2             * PERMIN (periodo minimo = velocita'
                               * massima)
      ble     MIN2             * se PERMIN<per2 salta a  proc che
                               * accelera

      ldd    #ACCEL
      std    inc

      ldd     #PERMIN          * setta il motore 1 al massimo, nel
      std     per2             * caso in cui l'accelerazione ha
                               * portato ad un periodo troppo
                               * piccolo


      ldd     #PERMIN          * setta il motore 2 al massimo, nel
      std     per3             * caso in cui l'accelerazione ha
                               * portato ad un periodo troppo
                               * piccolo


MIN2
      ldd    inc
      subd    #$0001
      std    inc

      ldd     per2             * accelera il motore 1 diminuendo il
                               * periodo
      subd    inc
      std     per2


      ldd     per3             * accelera il motore 2 diminuendo il
                               * periodo
      subd    inc
      std     per3
```

```
      rts



*********************************************************************
********** Funzione che gira a destra di 10 gradi  ***************
*********************************************************************
des10

      ldaa    #%00000100
      staa    PORTB,x
      ldd     #5000
      cpd     per3
      bge     d_alza
      std     per3
d_alza
      ldd    per3
      addd   #$0040
      std    per3

      ldd     #3000
      cpd     per2
      ble     d_abassa
      std     per2
d_abassa
      ldd     per2
      subd    #$0010
      std     per2
      rts



*********************************************************************
********** Funzione che gira a sinistra di 10 gradi  ***************
*********************************************************************
sin10

      ldaa    #%00000100
      staa    PORTB,x
      ldd     #5000
      cpd     per2
      bge     s_alza
      std     per2
s_alza
      ldd    per2
      addd   #$0040
      std    per2

      ldd     #3000
      cpd     per3
      ble     s_abassa
      std     per3
s_abassa
      ldd     per3
      subd    #$0010
      std     per3
```

```
      rts




*********************************************************************
************ Funzione che genera un ritardo software  **************
*********************************************************************
ritardo1
      psha
      pshb
      ldd #$ffff
salto1
      subd #$0001
      bne  salto1
      pulb
      pula

      rts

ritardo2
      psha
      pshb
      ldd #$09ff
salto2
      subd #$0001
      bne  salto1
      pulb
      pula


      rts

*********************************************************************
************ Funzione che legge i valori dai sensori  **************
*********************************************************************
legge
      ldaa   PORTC,x
      staa   baffo

      * conversione analogico digitale delle linee PE0..PE3
      ldaa   #%10010000
      staa   ADCTL,x
      jsr    aspetta

      * scrittura dei dati in memoria
      ldaa   ADR1,x
      staa   s1
      ldaa   ADR2,x
      staa   s2
      ldaa   ADR3,x
      staa   s3
      ldaa   ADR4,x
      staa   s4

      * conversione analogico digitale delle linee PE4..PE7
      ldaa   #%10010100
```

```
      staa   ADCTL,x
      jsr    aspetta

      * scrittura dei dati in memoria
      ldaa   ADR1,x
      staa   s5
      rts



*********************************************************************
********** Funzione per aspettare che la conversione ***************
*************** analogico digitale termini ***********************
*********************************************************************
aspetta
asp     ldaa   ADCTL,x
      anda   #$80
      beq    asp
      rts



ac_led
      psha
      ldaa   PORTB,x
      oraa   #%00100000
      staa   PORTB,x
      pula
      rts

sp_led
      psha
      ldaa   PORTB,x
      anda   #%11011111
      staa   PORTB,x
      pula
rts


*********************************************************************
*********** Funzione che fa andare in retro il robot  **************
*********************************************************************
retro
      psha
      pshb
      clr  baffo
      ldd  #3000
      std  per2
      std  per3
      ldab #%00000111
      stab  PORTB,x
      jsr ritardo1
      jsr ritardo1
      jsr ritardo1
      jsr ritardo1
      pulb
      pula
      rts
```

```
*********************************************************************
******** Funzione che fa girare a destra di 45° il robot  **********
*********************************************************************
des45
      ldab #%00000110
      stab  PORTB,x
      jsr ritardo1
      jsr ritardo1
      jsr ritardo1
      jsr ritardo1
      rts


*********************************************************************
****** Funzione che fa girare a sinistra di 45° il robot  **********
*********************************************************************

sin45
      ldab #%00000101
      stab  PORTB,x
      jsr ritardo1
      jsr ritardo1
      jsr ritardo1
      jsr ritardo1
       rts

*********************************************************************
************ Funzione che fa girare a destra il robot  *************
*************** dopo una collisione con il baffo SX ****************
*********************************************************************

desbaf
      ldab #%00000110
      stab  PORTB,x

      jsr ritardo1
      jsr ritardo1
      jsr ritardo1

      rts



*********************************************************************
********** Funzione che fa girare a sinistra il robot  *************
*************** dopo una collisione con il baffo DX ****************
*********************************************************************

sinbaf
      ldab #%00000101
      stab  PORTB,x
      jsr ritardo1
      jsr ritardo1
      jsr ritardo1

      rts
```

```
*********************************************************************
********* Funzione di interrupt che genera un'onda quadra ***********
***************** per il clock del motore 1 ************************
*********************************************************************
isr_oc2
      ldd     per2
      addd    TOC2,x
      std     TOC2,x

      ldaa    #$40
      staa    TFLG1,x
      rti


*********************************************************************
********* Funzione di interrupt che genera un'onda quadra ***********
***************** per il clock del motore 2 ************************
*********************************************************************
isr_oc3

      ldd     per3
      addd    TOC3,x
      std     TOC3,x

      ldaa    #$20
      staa    TFLG1,x
       rti


    END
```

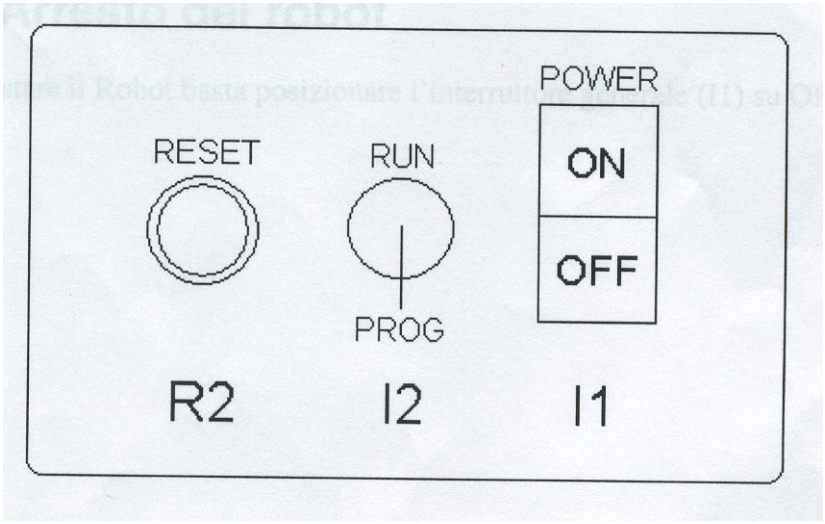
4. Modalità operative – Manuale utente

4.1. Scatola comandi

L'immagine sotto mostra la scatola comandi del sistema di controllo di Gongolo.

Sono presenti due interruttori e un pulsante:

- "**Power**" - Interruttore Il - (posizioni possibili ON - OFF): chiude il circuito di alimentazione del robot (motori, sistema di interfaccia dei motori, scheda di interfaccia dei sensori, sistema di controllo).
- "**Switch della modalità operativa**" - Interruttore I2 - (posizioni possibili ON - OFF): Il microcontrollore MC68HC11-E2 prevede due modalità di funzionamento, "Single chip" e "Special bootstrap". Il micro deve essere avviato in modo "Special bootstrap" (posizione di I2 su PROG) per essere programmato (scrittura del programma in EEPROM). Mentre per l'esecuzione del programma l'interruttore deve essere posto in modalità "Single chip" (posizione di I2 su RUN).
- "**Reset**" - Pulsante R2 - : Il microcontrollore deve essere resettato dopo ogni cambiamento di modalità operativa.



4.2. Avvio del Robot

Grazie alla scatola comandi realizzata nel corso del “Progetto Neogongolo”, per avviare il robot basta seguire le istruzioni seguenti:

1) Assicurarsi che l'interruttore argentato (I2), posto sulla scatola comandi, sia posizionato su RUN (modalità avvio).
2) Accendere il Robot mediante l'interruttore generale (I1) posto sulla scatola comandi.

A questo punto il Robot parte e, dopo una breve fase di accelerazione (dovuta alle rampe di accelerazione generate per i motori passo) prosegue con movimento rettilineo uniforme fintanto che i baffi o i sensori non rilevano un ostacolo o una parete.

4.3. Uso del Robot

Una volta avviato il Robot non necessita, in generale, di intervento umano. Il sistema sensoriale costituito dai baffi anteriori e dai 5 sensori, gli permette di rilevare un ostacolo o una parete. Il sistema di controllo, da noi implementato, prende le dovute iniziative per fermare il robot e tentare un cambiamento di direzione adeguato per seguire la parete.

Può succedere che la presenza di ostacoli non percettibili dai baffi o dai sensori forzino un intervento dall'esterno per ripristinare la posizione del robot.

Il tasto di reset (R2) posto sulla scatola comandi permette di riavviare il robot.

4.4. Arresto del robot

Per arrestare il Robot basta posizionare l'interruttore generale (I1) su OFF.

5. Conclusioni e sviluppi futuri

L'aggiunta dei cinque sensori di distanza permette l'acquisizione di maggiori informazioni sull'ambiente circostante, consentendo eventuali sviluppi futuri sul comportamento di Gongolo.

Oltre a migliorie di tipo software, si potrebbero pensare ulteriori sviluppi hardware.

Si potrebbe dotare Gongolo di sensori molto più precisi, come per esempio: bussola digitale, sonar e persino, sensori basati su GPS.

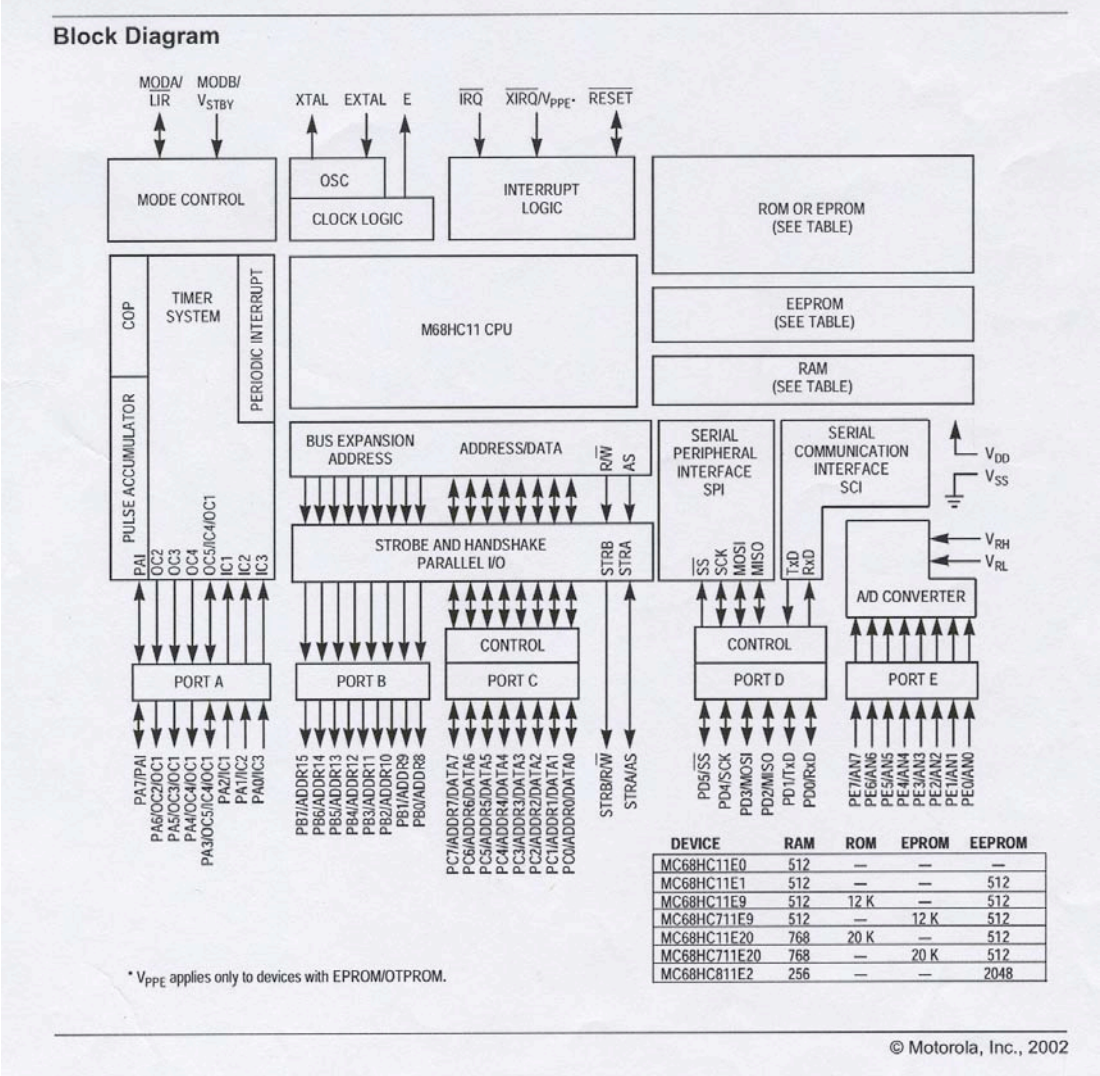
Il microcontrollore della Motorola 68HC11 consente una buona connettività, anche se la memoria EEPROM a disposizione del programmatore può rilevarsi insufficiente per routine troppo complesse.

A tal proposito la nostra esperienza in laboratorio testimonia che un comportamento un po' più articolato, come quello implementato attualmente su Gongolo, può richiedere l'impiego di numerose risorse in termini computazionali, di programmazione e di memoria usata per la scrittura del programma.

Suggeriamo, quindi, per eventuali lavori futuri su Gongolo, di fare un preliminare studio di fattibilità, per testarne gli eventuali limiti hardware e software.

Non è esclusa la necessità di una versione del microcontrollore in cui si possa utilizzare la modalità expanded multiplexing, per usufruire di una memoria esterna più capiente.

6 . Appendice A: Microcontrollore Motorola MC68HC11 (serie E)



DEVICE	RAM	ROM	EPROM	EEPROM
MC68HC11E0	512	—	—	—
MC68HC11E1	512	—	—	512
MC68HC11E9	512	12 K	—	512
MC68HC711E9	512	—	12 K	512
MC68HC11E20	768	20 K	—	512
MC68HC711E20	768	—	20 K	512
MC68HC811E2	256	—	—	2048

La sigla del processore che è stato montato sulla scheda di Gongolo è **MC68HC811E2**.

7. Appendice B: Istruzioni personalizzate della scheda 6811LABE

Per la descrizione tecnica della scheda si rimanda alla dispensa descitta al punto [1] della bibliografia.

Segue la tabella contenente una breve descrizione della configurazione dei jumper presenti sulla scheda 6811LABE adottata per il controllo di GONGOLO,

Jumper	Descrizione	Stato
J1	Se chiuso consente di attivare automaticamente il programma presente EEPROM in modalità "bootstrap"	Chiuso
J2	Cavo seriale (RS232): dritto (1-2 chiuso, 3-4 chiuso), null-modem (1-3 chiuso, 2-4 chiuso)	1-3 chiuso 2-4 chiuso
J3	Consensi automatici sul canale RS232	Tutti chiusi
J4	Tensioni di riferimento per converitore A/D	1-2 chiuso 3-4 chiuso
J5	Modalità di utilizzo: "special bootstrap" (chiuso) per la modalità RUN, "single chip" (aperto) per la modalità PROG	Variabile (Interruttore su scatola comandi)
J6	Se chiuso connette il pin XIRQ al morsetto 3 (+10V) (per programmazione del MC68HC711E9)	Aperto

8. Appendice C: Sensori piezoelettrici montati sui baffi

Parallelamente al lavoro svolto nel progetto per l'aggiunta di nuovi sensori a Gongolo, abbiamo analizzato la sensibilità dei baffi di Gongolo.

Innanzitutto va detto che Gongolo attualmente monta due baffi costituiti da fili di ferro ricurvi in grado di trasmettere vibrazioni (ad alta frequenza) ai sensori piezoelettrici accoppiati ad essi.

Un primo piccolo problema deriva dal fatto che uno dei due fili di ferro ha una sezione più piccola dell'altro. Questo fatto rende uno dei due baffi meno sensibili agli urti.

Tutto ciò è causato dalla minor rigidezza del filo a sezione minore, infatti l'urto genera vibrazioni più marcate sul filo più rigido.

Inoltre sono presenti dei giochi eccessivi nel punto di bloccaggio dei baffi alla struttura portante del robot. Questi giochi attenuano le vibrazioni (in questo caso su entrambi i baffi).

Per finire le guaine di gomma presenti nel punto di bloccaggio dei baffi sono sensibilmente deformate.

Conseguenza di tutto questo è che la sensorialità di Gongolo è leggermente diminuita nel tempo specialmente da parte del baffo destro che essendo più sottile è soggetto ad un gioco maggiore.

In futuro è possibile sostituire i sensori piezoelettrici con una nuova coppia e fissare in maniera differente i baffi alla struttura del robot.

9. Appendice D: SHARP GP2D12

GP2D12/GP2D15 General Purpose Type Distance Measuring Sensors

■ Features

1. Less influence on the color of reflective objects, reflectivity
2. Line-up of distance output/distance judgement type
   Distance output type (analog voltage) : **GP2D12**
   Detecting distance : 10 to 80cm
   Distance judgement type : **GP2D15**
   Judgement distance : 24cm
   (Adjustable within the range of 10 to 80cm)
3. External control circuit is unnecessary
4. Low cost

■ Applications

1. TVs
2. Personal computers
3. Cars
4. Copiers

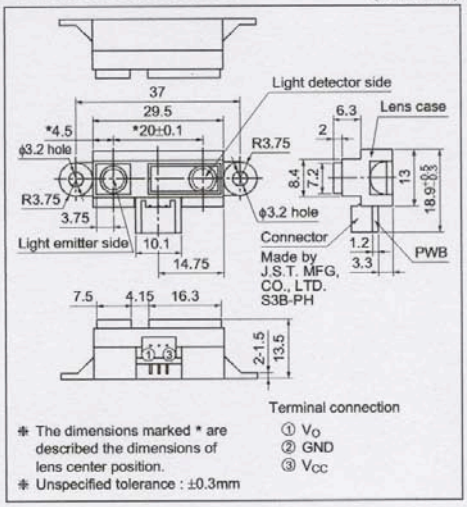
■ Absolute Maximum Ratings

(Ta=25°C, Vcc=5V)

Parameter	Symbol	Rating	Unit
Supply voltage	V_{CC}	−0.3 to +7	V
Output terminal voltage	V_O	−0.3 to Vcc +0.3	V
Operating temperature	T_{opr}	−10 to +60	°C
Storage temperature	T_{stg}	−40 to +70	°C

■ Outline Dimensions

(Unit : mm)



■ Recommended Operating Conditions

Parameter	Symbol	Rating	Unit
Operating supply voltage	V_{CC}	4.5 to +5.5	V

■ Electro-optical Characteristics

(Ta=25°C, V_{CC}=5V)

Parameter		Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Distance measuring range		ΔL	*1 *3	10	–	80	cm
Output terminal voltage	GP2D12	V_O	L=80cm *1	0.25	0.4	0.55	V
	GP2D15	V_{OH}	Output voltage at High *1	Vcc –0.3	–	–	V
		V_{OL}	Output voltage at Low *1	–	–	0.6	V
Difference of output voltage	GP2D12	ΔVo	Output change at L=80cm to 10cm *1	1.75	2.0	2.25	V
Distance characteristics of output	GP2D15	V_O	*1 *2 *4	21	24	27	cm
Average Dissipation current		I_{CC}	L=80cm *1	–	33	50	mA

Note) L : Distance to reflective object.

*1 Using reflective object : White paper (Made by Kodak Co. Ltd. gray cards R-27 · white face, reflective ratio ; 90%).
*2 We ship the device after the following adjustment : Output switching distance L=24cm±3cm must be measured by the sensor.
*3 Distance measuring range of the optical sensor system.
*4 Output switching has a hysteresis width. The distance specified by Vo should be the one with which the output L switches to the output H.

Fig.1 Internal Block Diagram

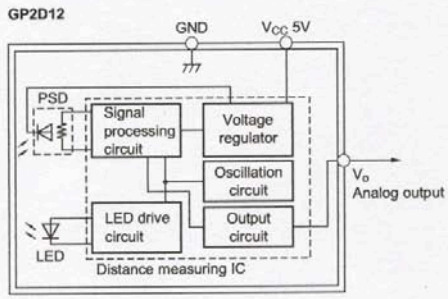


Fig.2 Internal Block Diagram

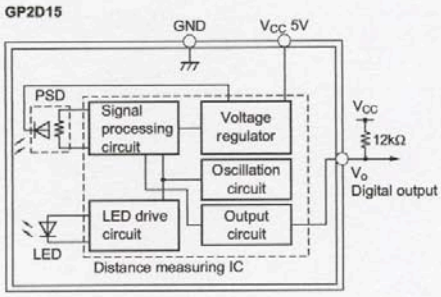


Fig.3 Timing Chart

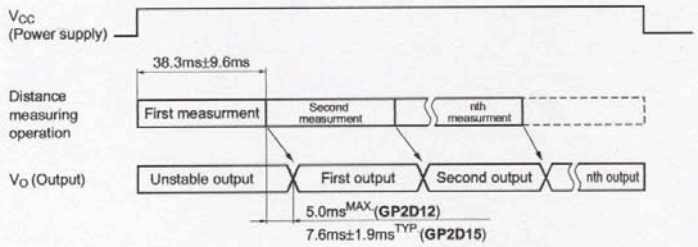


Fig.4 Distance Characteristics

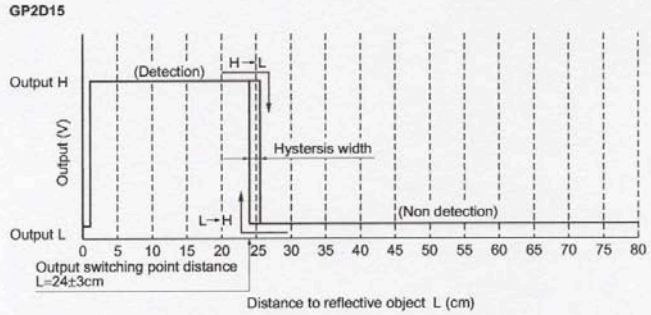


Fig.5 Analog Output Voltage vs. Surface Illuminance of Reflective Object

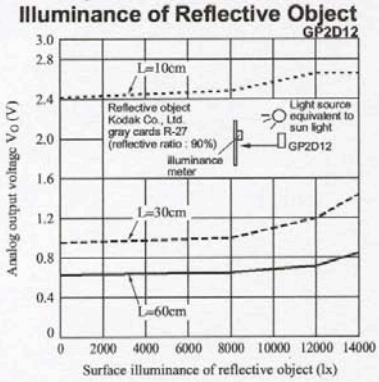


Fig.6 Analog Output Voltage vs.Distance to Reflective Object

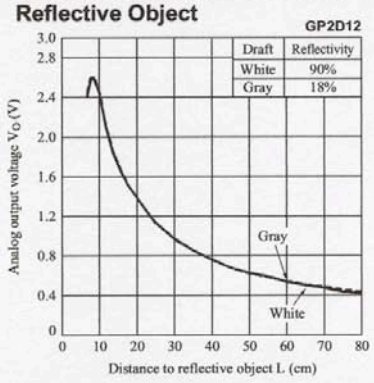


Fig.7 Analog Output Voltage vs.Ambient Temperature

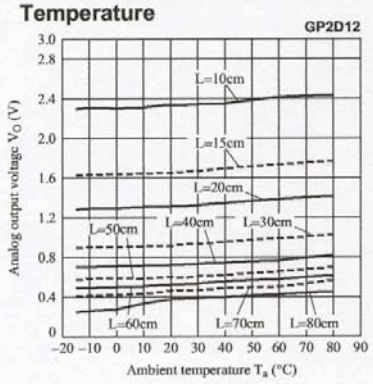
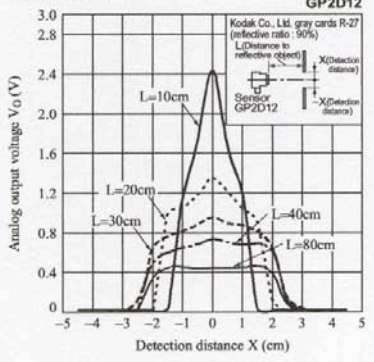


Fig.8 Analog Output Voltage vs.Detection Distance



Application Circuits

NOTICE

- The circuit application examples in this publication are provided to explain representative applications of SHARP devices and are not intended to guarantee any circuit design or license any intellectual property rights. SHARP takes no responsibility for any problems related to any intellectual property right of a third party resulting from the use of SHARP's devices.
- Contact SHARP in order to obtain the latest device specification sheets before using any SHARP device. SHARP reserves the right to make changes in the specifications, characteristics, data, materials, structure, and other contents described herein at any time without notice in order to improve design or reliability. Manufacturing locations are also subject to change without notice.
- Observe the following points when using any devices in this publication. SHARP takes no responsibility for damage caused by improper use of the devices which does not meet the conditions and absolute maximum ratings to be used specified in the relevant specification sheet nor meet the following conditions:
  (i) The devices in this publication are designed for use in general electronic equipment designs such as:
  – Personal computers
  – Office automation equipment
  – Telecommunication equipment [terminal]
  – Test and measurement equipment
  – Industrial control
  – Audio visual equipment
  – Consumer electronics
  (ii) Measures such as fail-safe function and redundant design should be taken to ensure reliability and safety when SHARP devices are used for or in connection with equipment that requires higher reliability such as:
  – Transportation control and safety equipment (i.e., aircraft, trains, automobiles, etc.)
  – Traffic signals
  – Gas leakage sensor breakers
  – Alarm equipment
  – Various safety devices, etc.
  (iii) SHARP devices shall not be used for or in connection with equipment that requires an extremely high level of reliability and safety such as:
  – Space applications
  – Telecommunication equipment [trunk lines]
  – Nuclear power control equipment
  – Medical and other life support equipment (e.g., scuba).
- Contact a SHARP representative in advance when intending to use SHARP devices for any "specific" applications other than those recommended by SHARP or when it is unclear which category mentioned above controls the intended use.
- If the SHARP devices listed in this publication fall within the scope of strategic products described in the Foreign Exchange and Foreign Trade Control Law of Japan, it is necessary to obtain approval to export such SHARP devices.
-
- Contact and consult with a SHARP representative if there are any questions about the contents of this publication.

SHARP

Bibliografia

[1] Alessandra Flammini: "**Progettazione a microprocessore**", Laboratorio di Elettronica Digitale A.A. 2001-02

[2] Programming and Reference Guide **MC68HC11** (serie E), Motorola, 2002

[3] M68HC11 **PCbug11** User's Manual, Motorola 1992

[4] Data Sheet del Sensore di distanza: **GP2D12**

[5] Marco Merigo, Luca Alberici, nicola Gatta, "**Progetto di Neogongolo**", Elaborato di Esame del corso di Robotica, 18 settembre 2002

[6] Riccardo Cassinis, dispense del Corso di Robotica, A.A. 2002-03

[7] Documentazione disponibile in formato HTML su WEB

Indice