Università degli studi di Padova

Corso di laurea triennale in Ingegneria Meccatronica

Studio e utilizzo del microcontrollore Arduino per applicazioni su un robot line follower

“Study and use of an Arduino microcontroller for application on a robot line follower”

Relatore : Prof. Sona Alessandro

Laureando: Dal Santo Marco

26 settembre 2011
Anno Accademico 2010/2011

Indice

1.INTRODUZIONE

L'obiettivo di questo studio è quello di imparare ad usare e programmare la piattaforma hardware “ARDUINO UNO”. Useremo come “cavia” per questo tipo di lavoro un robot del tipo “LINE FOLLOWER” (d'ora in poi LFR) con lo scopo di programmarlo nel eseguire le sue funzioni e nel perseguire i suoi obiettivi (rincorrere il tracciato disegnato) e nel vedere come reagisce e come si comporta al variare di alcuni parametri come ad esempio la frequenza dell’onda quadra del duty cycle o dal cambiamento del numero di bit dedicati alla lettura del segnale.

Tutto questo installando tra di loro: sensori ad infrarossi per il rilevamento del contrasto(linea da seguire-superficie d'appoggio),la piattaforma Arduino UN),un chip per il comando dei motori ed infine un rover per la movimentazione.

Nel nostro caso scegliamo componenti molto economici per riuscire a rientrare in un budget di circa 80€, questo riduce le configurazioni possibili per il robot: la maggior parte dei modelli che si trovano in rete lavorano con una moltitudine di sensori per sapere in ogni istante dove si trova la linea rispetto alle ruote del robot, mentre nel nostro caso il numero di sensori è limitato a 3.

E sempre per lo stesso motivo, data l'impossibilità (sia economica che di carattere tecnico) di poter acquistare un rover già pronto all'utilizzo si è optato per l'acquisto di un robot a basso costo cingolato dotato di un braccio meccanico e di un comando remoto a filo a cui verranno applicate le seguenti modifiche:

- Controllo elettronico: eliminazione del comando remoto con filo e della scheda di potenza che verranno sostituiti da un sistema di controllo e movimentazione autonomo e da una scheda di potenza adeguata.
- Struttura meccanica: eliminazione del braccio meccanico al quali andrà sostituita una struttura su cui installare il kit di comando “Arduino UNO” e la nuova scheda di potenza; inoltre verranno installate delle staffe per supportare dei sensori per la movimentazione.

2.STRUTTURA HARDWARE

In questa sezione è descritta la struttura fisica dei componenti che formano il robot, con considerazioni di carattere elettronico. Si farà riferimento anche ad informazioni presenti nei datasheet dei componenti disponibili online.

2.1 Rover

Rover “Remote Controller kit 2” (fig.1): Kit della serie Tamiya Educational Construction Series. Questi robot sono utilizzati nelle scuole, università, centri di ricerca e istituti.

Fig.1

2.2 Motoriduttore

I motori in corrente continua sono costruiti per funzionare permanentemente a velocità prossime alla propria velocità a vuoto. Tali velocità sono generalmente troppo elevate nella maggior parte dei casi. Per ridurre la velocità viene messa a disposizione degli utilizzatori una gamma completa di motoriduttori, ciascuno dotato di una serie di rapporti.
Questo insieme permette di soddisfare una molteplicità di applicazioni.
Per la scelta del rapporto di riduzione si possono usare vari metodi, uno dei quali è basato

sulla velocità richiesta sull'asse del riduttore e per la sua semplicità è il metodo più spesso utilizzato:

$$R=N1/Nb$$

N1 = velocità richiesta

Nb = velocità di base del motore

Il motoriduttore utilizzato nel nostro studio è il "doppio motoriduttore Tamiya 70097 (fig.2).

Quest'ultimo e' molto compatto (7.5cm di lunghezza), contiene due piccoli motori DC che trasmettono il moto a due assi esagonali da 3mm separati. Si può assemblare in due diversi rapporti di riduzione: veloce 58:1 o lento 203:1. In entrambe i casi i motori producono una elevata potenza.

Caratteristiche tecniche:

Alimentazione 3Vdc (4.5Max)
Tipo Motore FA-130 : Tensione 2.4 – 3V
Velocità 12300 rpm
Coppia 4.6 gcm
Corrente 0.5 A
Rapporti 58:1 - 203:1
Dimensioni 75 x 50 x 23 mm

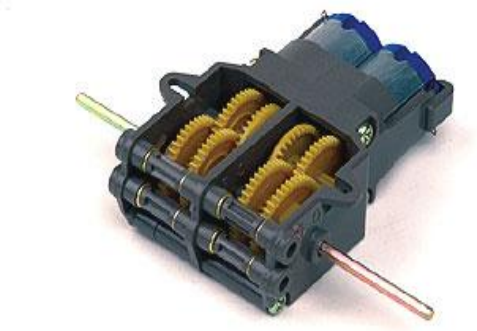


Fig.2

2.3 Motor Driver L293D

Chip della SGS Thompson, l'L293D (fig.3) e' l'ideale per pilotare piccoli motori in CC. Può pilotare indipendentemente una coppia di motori CC da 0.6 A (max) con voltaggio da 4.5 a 36 V. E' dotato di una protezione interna che, in caso di surriscaldamento, limita la corrente in uscita fino al ripristino della temperatura ottimale.

Fig.3

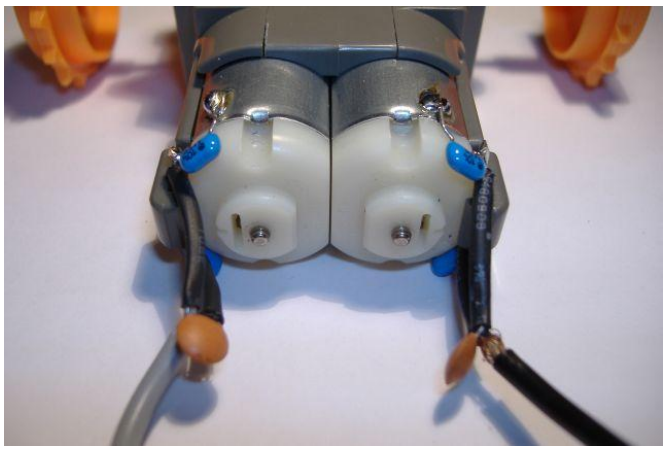
2.4 Motore elettrico

Vengono utilizzati dal motoriduttore Tamiya 70097 due motori DC “FA-130” collocati all'interno di un pezzo plastico sempre della Tamiya (Tamiya 70168 double gearbox) . I due motori sono pilotabili in modo completamente indipendente e i due rapporti di riduzione sono stati impostati in modo opportuno.
Caratteristiche tecniche FA-130:

Voltage	Operating range	1.5 ÷ 3.0	V
	Nominal	3V costant	V
No load	Speed	12300	rpm
	Current	0.15	A
At maximum efficiency	Speed	9710	Rpm
	Current	0.56	A
	Torque	0.74	mN*m
		7.6	g*cm
	Output	0.76	W
Stall	Torque	3.53	mN*m
		36	g*cm
	Current	2.10	A

In figura 4 è visibile lo schema di collegamento dei condensatori utilizzati come filtri anti disturbo. Essi servono a sopprimere i disturbi causati dallo scintillio del motore elettrico, infatti lo sfregamento delle spazzole sul collettore può causare interferenze con la circuiteria elettronica. Per questo occorre saldare tre condensatori ceramici(o meglio

poliestere) da 100nF uno tra positivo e carcassa, uno tra negativo e carcassa e uno tra positivo e negativo.



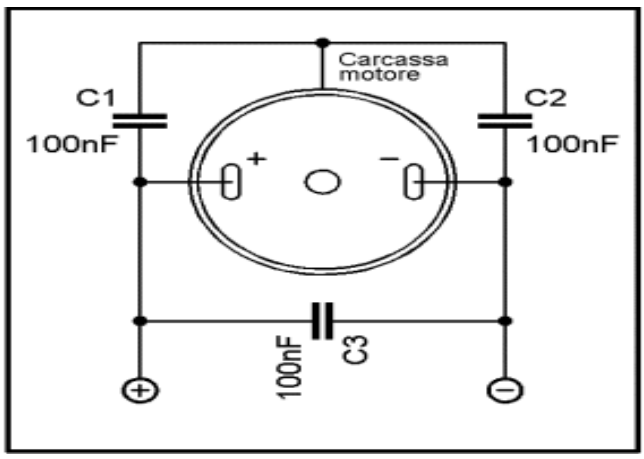


Fig.4

2.5 Sensore riflettivo CNY70

2.5.1 Sensori in generale

I sensori sono componenti elettronici in grado di convertire una grandezza fisica in una grandezza elettrica misurabile, nel nostro caso ad esempio, da Arduino.

Una gran parte dei sensori si basa sul fatto che molti materiali cambiano resistenza elettrica al cambiare di qualche fenomeno fisico. Per esempio, qualunque conduttore aumenta la propria resistenza elettrica quando aumenta la temperatura, diventando perciò un sensore di temperatura. Negli anni è stato svolto un lavoro di ricerca per scoprire in quale materiale questa relazione è la più prevedibile possibile. Questo ha portato alla realizzazione dei termistori NTC (Negative Temperature Coefficient) (resistenza che decresce con l'aumentare della temperatura) dove la resistenza si riduce tra il 2% e il 6% per ogni grado di variazione della temperatura. Questo permette di realizzare termometri piuttosto precisi. Altri sensori molto diffusi, per esempio, sono i fotoresistori che cambiano la loro resistenza in base alla luce che li colpisce. In questo caso il fotoresistore ha una resistenza molto alta al buio e, appena la luce ne colpisce la superficie, la resistenza scende in maniera proporzionale. Questo permette di realizzare, per esempio, gli esposimetri (strumenti utilizzati in fotografia e nel cinema per quantificare la luce presente in una scena) oppure i sensori che accendono la luce in giardino quando diventa buio.
La ricerca ha portato a scoprire innumerevoli materiali che reagiscono ai più diversi fenomeni fisici.

Nella pratica i sensori si dividono in tre grandi famiglie: Digitali, Analogici e "Complessi".
I sensori digitali sono quelli che hanno solo due stati possibili: acceso o spento. Per esempio, un pulsante è un sensore digitale perché può essere solo premuto oppure no. Un sensore analogico, invece, produce una variazione continua in base al valore del fenomeno fisico che sta misurando, per esempio un sensore di temperatura cambia resistenza proporzionalmente alla temperatura. Questa variazione produce una serie di valori continui.
Infine per sensori complessi si indicano quei dispositivi che contengono una certa intelligenza e producono le loro informazioni attraverso dei flussi di dati digitali. Per esempio un ricevitore GPS che è in grado di darci la nostra posizione geografica produce i dati inviandoli sotto forma di caratteri ASCII fornendo oltre che la posizione anche la

direzione di movimento e la velocità. Non sarebbe semplice rappresentare questi valori attraverso delle variazioni di tensione elettrica ma è necessario che l'Arduino implementi il "protocollo di comunicazione" del sensore.

Il CNy70 (fig.5) e' un sensore molto diffuso, si interfaccia facilmente con qualsiasi microcontrollore, produce un'uscita analogica quando individua una superficie. E' composto da un LED ad infrarossi ed un fototransistor. Il fototransistor è un transistor a giunzione bipolare che viene inscatolato in un contenitore trasparente in modo che la luce possa raggiungere la giunzione del collettore di base: si può paragonare praticamente ad una foto resistenza in cui la cui resistenza elettrica è inversamente proporzionale alla quantità di luce che la colpisce. Si comporta come un tradizionale resistore, ma il suo valore in Ohm diminuisce mano a mano che aumenta l'intensità della luce che la colpisce. Ciò comporta che la corrente elettrica che transita attraverso tale componente è proporzionale all'intensità di una sorgente luminosa. In tale maniera si realizza una sorta di potenziometro attuabile tramite la luce anziché tramite forze meccaniche o segnali elettrici. Sfrutteremo questa caratteristica del sensore nel far passare o meno corrente a seconda della superficie che gli si presenta, e con delle prove determineremo la tensione che si misura con una superficie nera e parallelamente con una superficie bianca in modo tale da trovare un range di tensioni dove poter lavorare durante la programmazione.

Fig.5

Tre di questi sensori sono montati in linea a capo del rover ad una distanza dal suolo non superiore ai 3 millimetri. Tali dispositivi hanno dei package di dimensione molto ridotta (inferiore al centimetro di lato) e hanno 4 piedini per il loro utilizzo (fig. 6).

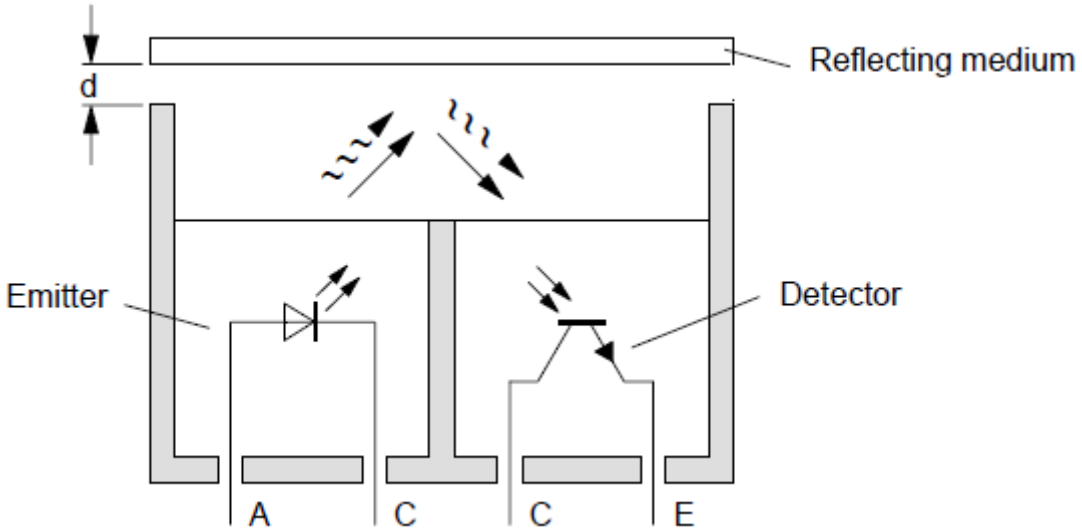


Fig.6

Da sinistra verso destra evidenziamo:

1) **A – Anodo**

Questo va collegato alla batteria esterna in modo tale da fornire al diodo emettitore una corrente sufficiente (il valore effettivo si può regolare con una opportuna resistenza).

2) **C – Catodo**

Collegato a massa.

3) **C – Collettore**

Quando il transistor BJT fotorilevatore viene illuminato con luce infrarossa riflessa da un oggetto ad alto fattore di riflessione, entra in conduzione (interruttore chiuso) e il collettore viene portato alla tensione dell'emettitore. Al collettore collegheremo pertanto una resistenza di pull-up e all'emettitore collegheremo la massa.

4) **E – Emettitore**

Collegato a massa.

Lo schema di montaggio (fig.7) che si è usato per ognuno dei 3 sensori è allora il seguente.

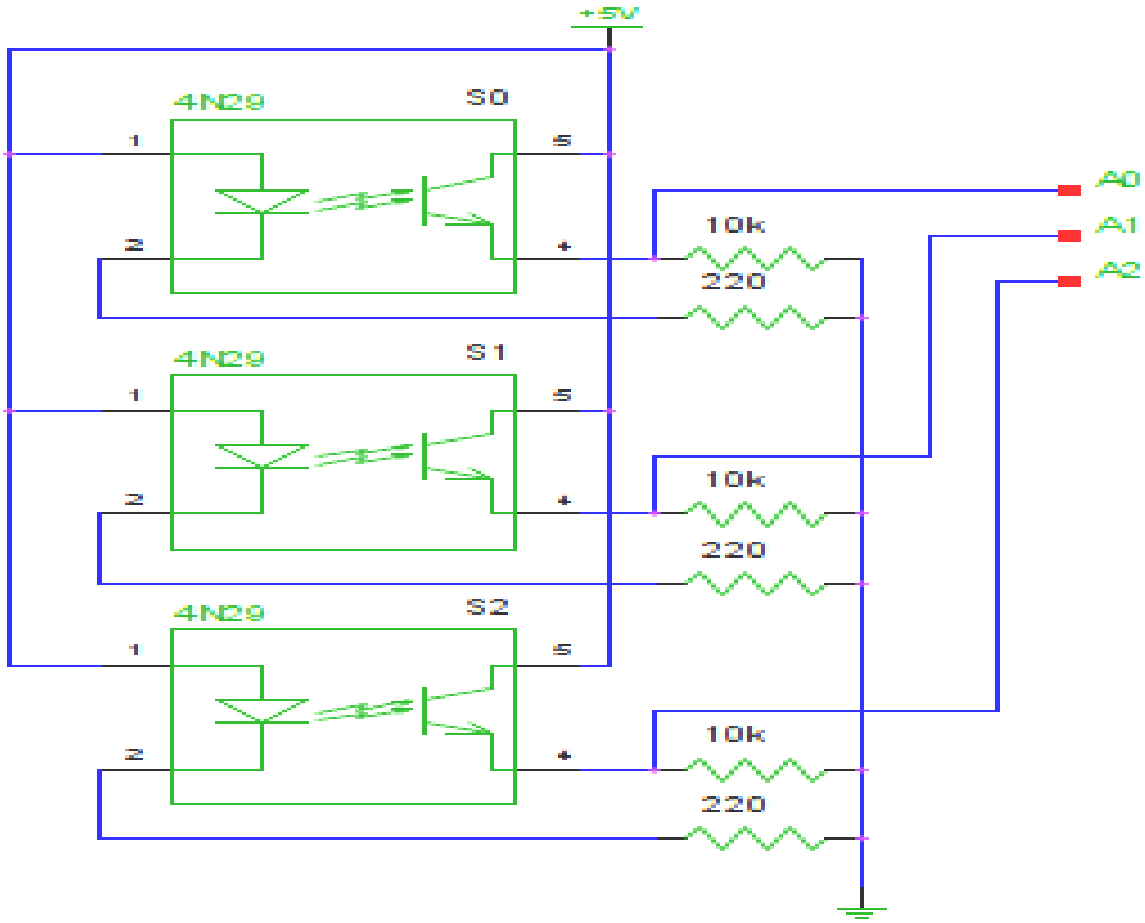


Fig.7

La resistenza da 220 Ω per il pull-up dell'anodo del diodo garantisce una corrente di circa 22mA che forza il diodo ad emettere una luce IR sufficiente ad essere rilevata dal transistor. La resistenza da 10 kΩ per il pull-up sul collettore del fototransistor garantisce invece una corrente molto contenuta all'interno del transistor: questo comporta anche consumi di potenza molto ridotti per la parte relativa al fototransistor.

In generale, la tensione misurata sul piedino del microcontrollore sarà un valore intermedio tra 0 e 4.99 V a seconda della corrente che il transistor di rilevazione lascia scorrere (la quale influenza la caduta di potenziale sulla resistenza da 10 kΩ).

2.6 Arduino UNO

Arduino UNO è una piattaforma open-source basata su una scheda a microcontrollore ed un semplice ambiente di sviluppo che implementa il linguaggio Processing/Wiring. Le sue dimensioni sono quelle di una carta di credito, ed il suo costo è molto contenuto, circa 26.00 euro.

La principale caratteristica di Arduino sta nel fatto che il suo sistema è "aperto" sia nel software e sia nel hardware: gli schemi elettrici e tutte le altre informazioni per costruire un clone di Arduino sono reperibili online gratuitamente. Quindi abbiamo un sistema con software aperto, hardware aperto e documentazione facilmente raggiungibile e utilizzabile in internet, spiegando l'elettronica ed un linguaggio più semplice possibile è stato creato un movimento che copre praticamente tutto il mondo dove Arduino viene implementato per realizzare progetti che vanno dagli strumenti musicali alle installazione interattive per i musei, ai prodotti tecnologici come i lettori mp3 e micropc portatili, alla riproduzione di strumenti da laboratorio ad alto costo (non disponibili a tutti per questo motivo) ed in molte università e scuole per scopi didattici.

Oltre alla scheda elettronica, Arduino include anche un ambiente di sviluppo (IDE in inglese) cioè un programma da installare sul pc per scrivere i programmi che si vogliono caricare sulla scheda Arduino (nello slang di Arduino sono denominati sketch). Una volta pronto, il programma, attraverso un pulsante viene trasformato in un eseguibile adatto al processore di Arduino e caricato sulla scheda.

Arduino funziona manipolando segnali elettrici. L'elettronica digitale si basa a sua volta sulla numerazione binaria, cioè al suo interno il processore conosce solo i numeri 1 e 0 che vengono rappresentati come la presenza o l'assenza di una tensione elettrica. Immaginiamo un sistema dove ogni bit è rappresentato da una lampadina collegata ad una batteria attraverso un interruttore. Quando questa è accesa rappresenta un 1 mentre quanto è spenta rappresenta uno 0. Assemblando un certo numero di lampadine (per esempio 8) è possibile rappresentare tutti i numeri dallo 0 al 255.

Con questi numeri binari è anche possibile fare delle operazioni: si possono sommare, sottrarre, moltiplicare come i numeri decimali. Questo significa che se anche all'interno del nostro Arduino tutto viene rappresentato solo con 1 e 0, all'esterno non ce ne accorgiamo perché il linguaggio con cui si programma Arduino si occupa di tradurre quello che scriviamo noi in "uno" e "zero" e a far funzionare tutto correttamente.

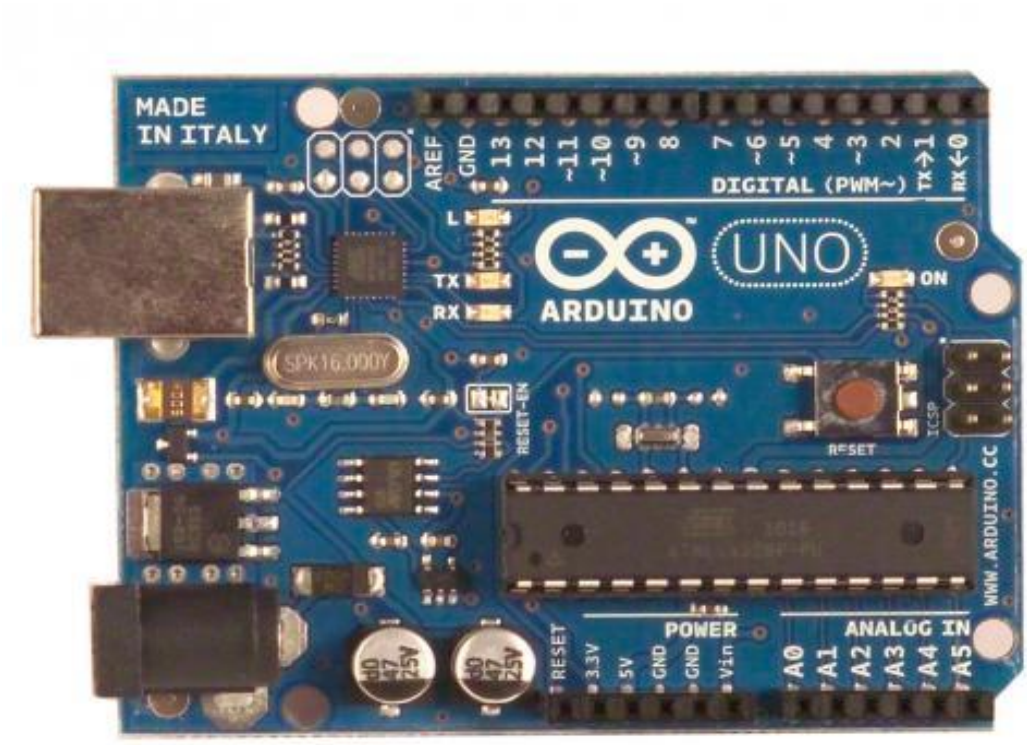


Fig.8

Guardando la scheda in fig.8 da sinistra a destra, dall'alto in basso

- Analog Reference pin (orange)
- Digital Ground (light green)
- Digital Pins 2-13 (green)
- Digital Pins 0-1/Serial In/Out – TX/RX (dark green)
- Reset Button – S1 (dark blue)
- In-circuit Serial Programmer (blue-green)
- Analog In Pins 0-5 (light blue)
- Power and Ground Pins (power: orange, grounds: light orange)
- External Power Supply In (9-12VDC) – X1 (pink)
- Toggles External Power and USB Power (place jumper on two pins closest to desired supply) – SV1 (purple)
- USB (used for uploading sketches to the board and for serial communication between the board and the computer; can be used to power the board) (yellow)

Microcontrollori			
ATmega168 (usato sulla maggior parte delle schede)		**ATmega8** (usato nelle prime schede)	
Digital I/O Pins	14 (of which 6 provide PWM output)	Digital I/O Pins	14 (of which 3 provide PWM outp
Analog Input Pins	6 (DIP) or 8 (SMD)	Analog Input Pins	6
DC Current per I/O Pin	40 mA	DC Current per I/O Pin	40 mA
Flash Memory	16 KB	Flash Memory	8 KB
SRAM	1 KB	SRAM	1 KB
EEPROM	512 bytes	EEPROM	512 bytes

2.6.1 Pin Digitali

Oltre alle funzionalità specifiche elencate più sotto, i Pin Digitali dell'Arduino possono essere utilizzati come input o output attraverso i comandi pinMode(), digitalRead(), e digitalWrite(). Ogni Pin ha una resistenza di pull-up interna che può essere accesa o spenta usando digitalWrite() (con un valore di HIGH o LOW)

quando il Pin é configurato come input. Il carico massimo di corrente per Pin é di 40 mA.

- Serial: 0 (RX) and 1 (TX). usati per ricevere (RX) o inviare (TX) dati seriali (TTL). Sull'Arduino questi Pin sono connessi con i rispettivi pin del chip FTDI che traduce la trasmissione da Seriale a USB.
- External Interrupts: 2 e 3.Questi pin possono essere configurati per attivare un interrupt su un valore basso, un fronte di salita o di discesa, o un cambiamento di valore.
- PWM: 3, 5, 6, 9, 10, e 11. Fornire a 8-bit l'uscita PWM con la funzione analogWrite (). Sulle schede con un ATmega8, l'uscita PWM è disponibile solo sui pin 9, 10 e 11.
- BT Reset: 7. Collegato alla linea di reset del modulo bluetooth.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Questi pin supportano la comunicazione SPI, che, pur messe a disposizione dall'hardware sottostante, non è attualmente incluso nel linguaggio Arduino.
- LED: 13. Sul Diecimila e LilyPad, vi è un built-in LED connesso al pin digitale 13. Quando il pin è un valore HIGH, il LED è acceso, quando il pin è LOW, è spento.

2.6.2 Pin Analogici

Oltre alle funzionalità specifiche elencate più sotto, i Pin Analogici supportano conversioni da analogico a digitale con una risoluzione di 10 bit (ADC) usando la funzione analogRead(). I Pin Analogici possono essere anche usati come Pin Digitali: analog input 0 come digital pin 14 fino a analog input 5 come digital pin 19. I Pin Analogici 6 and 7 (presenti sull'Arduino Mini e BT) non possono essere usati come Pin Digitali.

2.6.3 Pin di Alimentazione

- VIN (alcune volte appare come “9V”) : il voltaggio della scheda quando l’Arduino usa un’alimentazione esterna (diversa quindi dall’alimentazione standard dell’USB di 5V). Si può alimentare la scheda attraverso questo Pin, se la scheda é alimentata attraverso il jack si può ricevere l’alimentazione da questo Pin. Le modalità di alimentazione variano da scheda a scheda.
- 5V: l’alimentazione standard usata per alimentare la scheda e i sensori/attuatori attaccati ad essa. Questo può avvenire anche attraverso VIN con il regolatore di tensione a bordo della scheda, o da un altra alimentazione a 5V.
- 3,3V (Solo 2009) : pin di alimentazione legato al Chip FTDI, ma accessibile a diversi usi.
- GND : masse.

2.6.4 Altri Pin

- AREF: tensione di riferimento per gli ingressi analogici. Utilizzato con analogReference ().
- Reset (Solo 2009):portare tale linea a LOW per resettare il microcontrollore.

3.Cenni sul motore a spazzole

Il classico motore in corrente continua ha una parte che gira detta appunto rotore o anche armatura e una parte che genera un campo magnetico fisso (nell'esempio i due magneti colorati) detta statore.
Un interruttore rotante detto commutatore o collettore a spazzole inverte due volte ad ogni giro la direzione della corrente elettrica che percorre i due avvolgimenti generando un campo magnetico che entra ed esce dalle parti arrotondate dell'armatura. Nascono forze di attrazione e repulsione con i magneti permanenti fissi (indicati con N ed S nella figura 9).

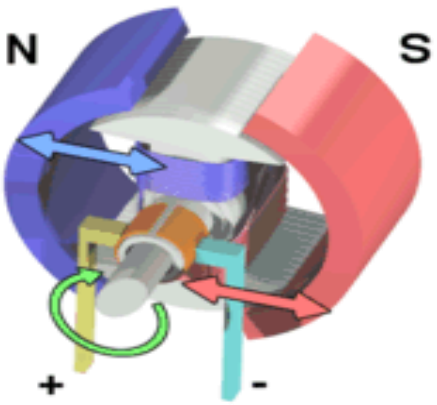


Fig.9

La velocità di rotazione dipende da:

- Tensione applicata.
- Corrente assorbita dal rotore.
- Carico applicato.

La coppia generata è proporzionale alla corrente. Il controllo più semplice agisce sulla tensione di alimentazione. Nei sistemi più complessi si usa un controllo automatico in

retroazione che legge le variabili per generare la tensione da applicare al motore. Il motore CC a magneti permanenti ha un comportamento reversibile: diventa un generatore di corrente continua se si collega un altro motore all'albero. Si può allora prelevare l'energia elettrica prodotta collegandosi alle spazzole.

Il suo limite principale è nella necessità del commutatore a spazzole:

- Le spazzole sono in grafite, mentre nei piccoli servomotori e nei tipi utilizzati nei lettori CD/DVD o registratori a cassette sono in lega metallica bianca. La differenza è nella frequenza della loro sostituzione, infatti nelle macchine utensili come smerigliatrici o trapani, si utilizzano spazzole in grafite, perché è molto semplice e veloce sostituirle, le spazzole in metallo, sono usate su apparecchi dove risulta scomodo o non conveniente cambiarle, come nei motori d'avviamento dei mezzi di trasporto.
- Le spazzole pongono un limite alla massima velocità di rotazione: maggiore è la velocità e più forte è la pressione che bisogna esercitare su di esse per mantenere un buon contatto, comunque i motori usati negli aspirapolvere e negli elettroutensili portatili (trapani, mole, ecc.) possono raggiungere i 35000-45000 giri al minuto.
- Tra spazzole e collettore, nei momenti di commutazione, si hanno transitori di apertura degli avvolgimenti induttivi e quindi scintillio (attenuabile con opportuni sistemi ma non eliminabile).
- Queste scintille comportano disturbi elettrici sia irradiati nell'ambiente circostante che trasmessi al generatore di tensione (che alimenta il motore); questi disturbi, in determinati settori di impiego, possono causare problemi di compatibilità elettromagnetica.

La presenza di avvolgimenti elettrici sul rotore ha anche due aspetti negativi:

- Se il motore è di grossa potenza si hanno dei problemi di smaltimento del calore (gli avvolgimenti si riscaldano per effetto Joule e il campo magnetico alternato nel nucleo del rotore genera altre perdite, causate da isteresi

magnetica e correnti parassite nel nucleo stesso, e quindi altro calore.

- Gli avvolgimenti appesantiscono il rotore (aumenta il momento d'inerzia): se il motore deve rispondere con rapidità e precisione (come avviene nelle automazioni industriali e nella robotica) il controllo diventa più complesso; per piccole potenze (da 1 a 200W) e servocontrolli a volte si usano particolari tipi di motori con rotore con avvolgimenti a forma di bicchiere e privo del nucleo di ferro, detti "ironless": hanno bassa inerzia e rendimento elettrico più elevato dei loro corrispondenti con rotore avvolto su nucleo di ferro.

4.Pilotaggio e controllo dei motori cc

4.1 Modalità ON/OFF

I motori DC possono essere pilotati in vari modi. Il modo più semplice per comandare un motore DC che debba girare in un solo verso è per mezzo di un transistor BJT o per mezzo di un transistor MOS (fig.10). Tale configurazione rappresenta la stessa che viene utilizzata per comandare carichi induttivi.

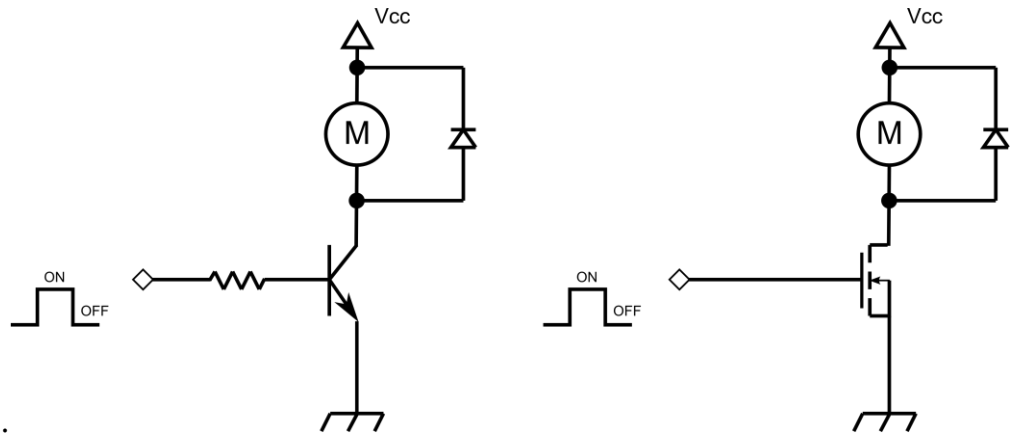


Fig.10

Il diodo di ricircolo viene utilizzato per proteggere la giunzione dei transistor dalle sovratensioni che si vengono a creare nel momento dello spegnimento del transistor stesso. Nel caso di motori DC a spazzole, ovvero gran parte dei motori DC presenti nei giocattoli, sono spesso utilizzati 1-3 condensatori al fine di ridurre gli effetti nocivi delle spazzole (ai fini elettromagnetici). Questi risultano spesso saldati tra i terminali di alimentazione e il contenitore del motore. La configurazione appena descritta, se pur semplice, oltre a permettere di attivare e disattivare il transistor, ovvero il motore, permette, se abbinata ad un segnale PWM (Pulse Width Modulation) di controllare la velocità del motore stesso. Se non viene fatto uso di alcun encoder per rilevare la posizione del rotore, il controllo della velocità è detto ad anello aperto, mentre nel caso in cui sia presente un encoder e quindi la

possibilità di misurare la velocità in uscita (ovvero un feedback) si parla di controllo ad anello chiuso (closed loop).

4.2 Ponte H (H-Bridge)

Per far girare il motore nel verso opposto di marcia è necessario far entrare nei morsetti una corrente inversa rispetto a quella iniziale; per far questo si fa utilizzo di un circuito denominato a "ponte H"(fig.11) costituito da 4 interruttori comandati e 4 diodi di ricircolo:

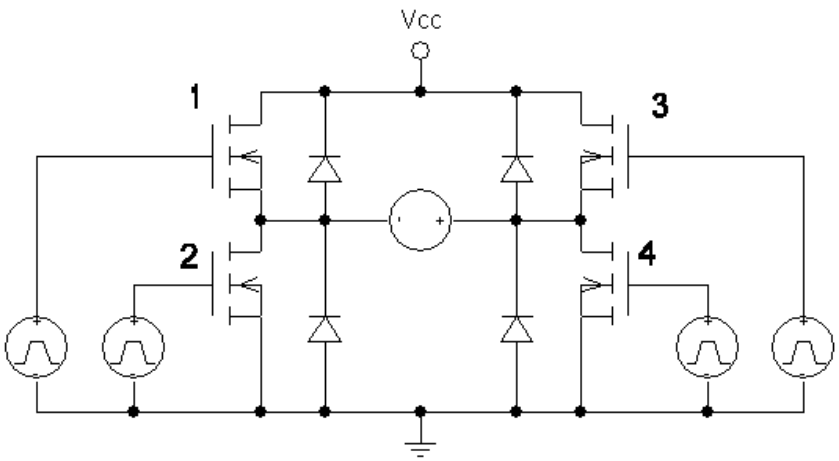


Fig.11: schema elettrico di un ponte h

Lo schema mostra quattro transistor sono connessi. In genere i due transistor inferiori (2 e 4 nello schema) sono detti di sink in quanto assorbono la corrente proveniente da motore oppure low side switch*;* i due transistor connessi direttamente alla Vcc sono detti di source oppure high side switch.

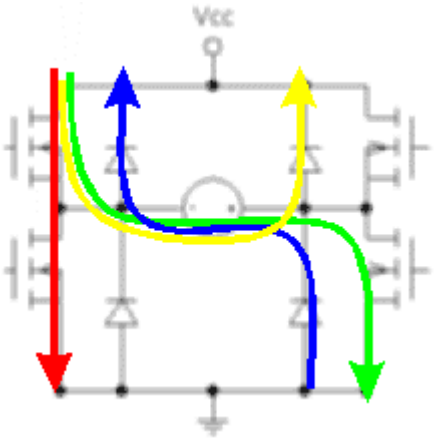


Fig.12

A seconda di quali transistor sono attivi, abbiamo diversi possibili percorsi per la corrente (fig.12):

- (linea verde) Se è attiva un transistor di sink ed uno di source appartenenti a lati opposti del ponte, abbiano passaggio di corrente nel motore. In questo caso il motore è in rotazione in un verso; per ottenere la rotazione opposta è evidentemente necessario attivare la coppia simmetrica.

- (linea rossa) Se è attiva un transistor di sink ed uno di source appartenenti allo stesso lato del ponte abbiamo un corto circuito. Inutile dire che questa situazione deve essere evitata nel modo più assoluto in quanto porterebbe alla distruzione del ponte o dell'alimentazione in tempi brevissimi.

- (linea blu) Se tutti i transistor sono spenti non abbiamo maglie in cui possa passare la corrente fornita dall'alimentatore. Quella indicata è la via che l'eventuale corrente accumulata dall'induttore percorre: si tratta ovviamente di un fenomeno temporaneo ma che deve necessariamente essere previsto. Terminata la scarica dell'induttore non si ha più passaggio di corrente e se il motore era precedentemente in moto si arresta lentamente a causa degli attriti meccanici.

- (linea gialla) Se è attivo almeno uno dei transistor di source e nessuno di quelli di sink non vi sono percorsi in cui passa la corrente fornita dall'alimentatore. La differenza rispetto alla situazione precedente è il sostanziale cortocircuito che si viene a creare ai capi del motore: infatti la tensione ai capi del motore è pari alla tensione diretta del diodo sommata a quella di conduzione del transistor. L'effetto è una vigorosa azione frenante.

Esistono due tipi di ponte H: i “ponti H discreti”, costituiti da componenti discreti sparsi come diodi e transistor, ed i “ponti H integrati” dove tutto il circuito è racchiuso in un package plastico di tipo DIP (dual in-line package). Questi ultimi sono molto versatili e, oltre a garantire una bassa occupazione di area nel circuito hanno buone prestazioni, possono essere montati in parallelo per ottenere alte correnti e riescono a lavorare in un intervallo di tensioni di alimentazione molto ampio (da V a 48V a seconda del modello).

5.Ponte H con l'integrato L293D

Nel nostro caso utilizziamo per costruire il "ponte H" l'integrato ST-L293D della THOMSON (fig.14) il cui schema a blocchi ed il diagramma delle connessioni sono riportati in figura 13:

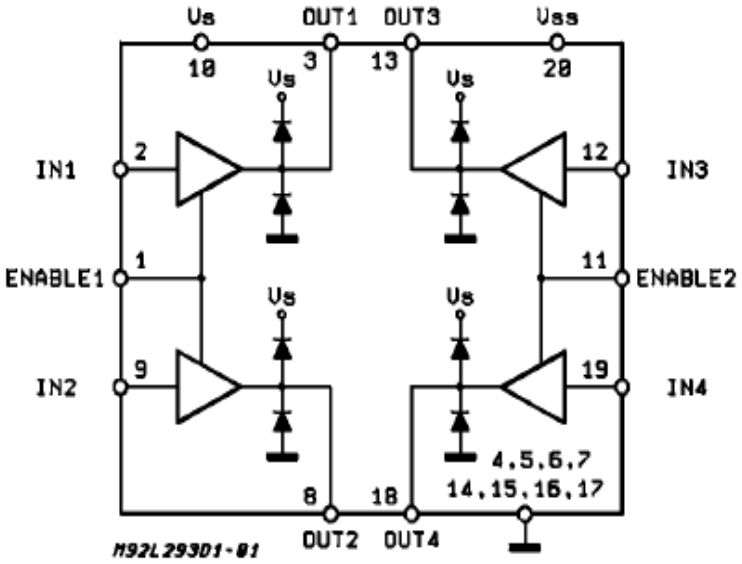


Fig.13

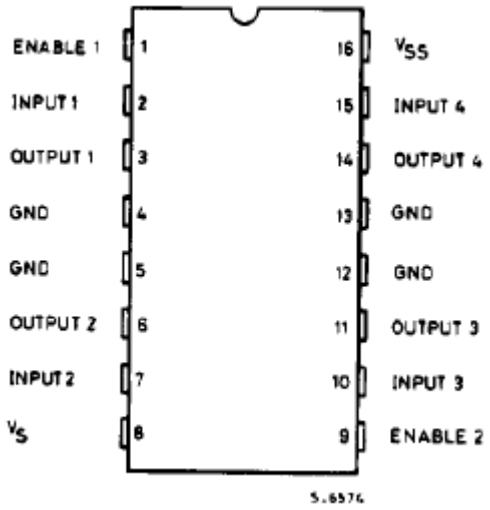
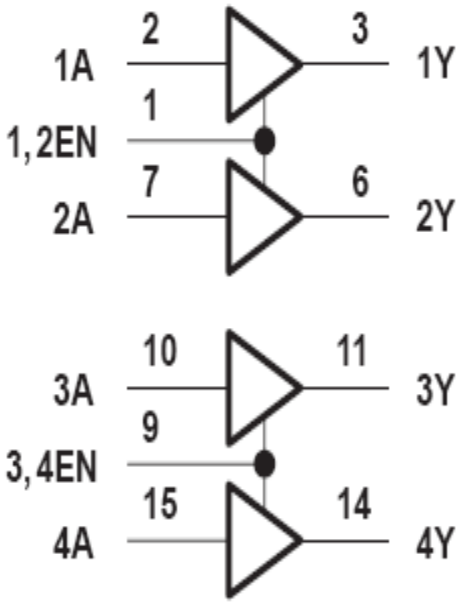


Fig.14

Nella tabella vengono riportate le funzioni dei vari piedini:

Piedini	Funzione	
1	Enable motore #1	
2-7	Ingressi driver motore #1	
3-6	Uscite motore #1	
9	Enable motore #2	
10-15	Ingressi driver motore #2	
11-14	Uscite motore #2	
4-5-12-13	Negativo	
8	Alimentazione positiva motori	
16	Alimentazione positiva logica	

L'integrato L293D presenta quattro piedini (2,7,10 e 15 che verranno collegati rispettivamente ai pin 13,11,10 e 12 di Arduino), che servono per disabilitare il motore corrispondente senza dover togliere la tensione di alimentazione al ponte. Questi piedini vengono usati anche per controllare la velocità del motore, inviando al ponte H un segnale PWM. Nello schema proposto questi due piedini sono forzati al livello logico 1 (HIGH)

tramite le resistenze di pull-up da 10k. In questo modo i motori funzionano regolarmente. Per disabilitare i motori, i piedini 2,7,10 e 15 vanno portati a livello logico 0 (LOW) cioè collegati al negativo.

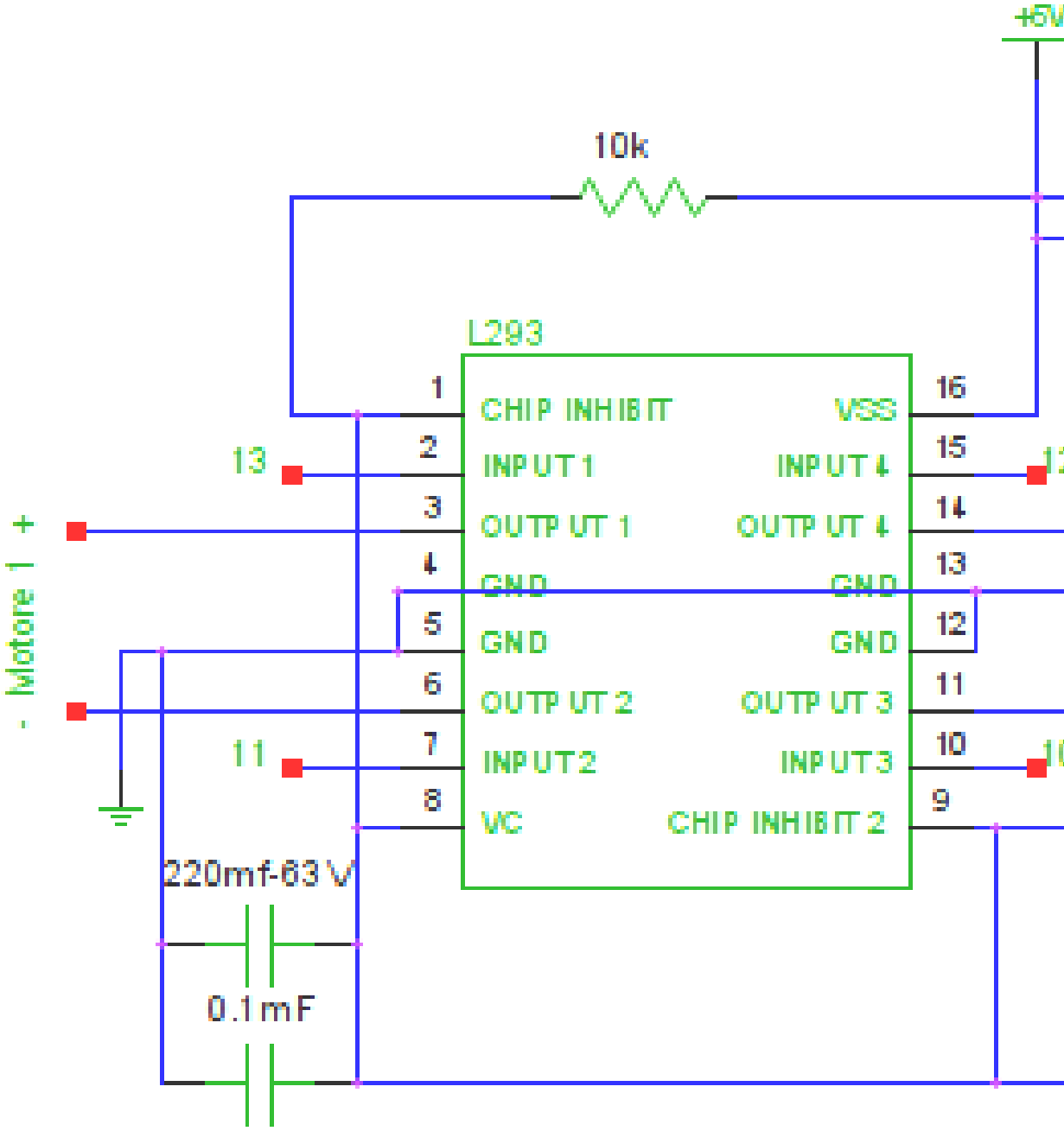


Fig.15: Circuito elettrico ponte H utilizzato

Per risolvere il problema del verso di rotazione basta osservare che impostando a livello “HIGH” i piedini 2 e 15 e contemporaneamente a livello “LOW”i piedini 7 e 10 il motore ruoterà in un senso, mentre invertendo le connessioni (2 e 15 a livello “LOW” e 7 e 10 a livello “HIGH”) cambieremo il senso di rotazione.

L’altro problema è quello di poter regolare la velocità di rotazione dei motori, e per far ciò è necessario introdurre il concetto di PWM e di controllo PWM.

6. Regolazione velocità motori cc : modalità PWM

Un segnale PWM (Pulse Width Modulation ovvero modulazione a variazione della larghezza d'impulso) è un' onda quadra di duty cycle variabile (fig.16) che permette di controllare l'assorbimento (la potenza assorbita) di un carico elettrico(nel nostro caso il motore DC), variando (modulando) il duty cycle.

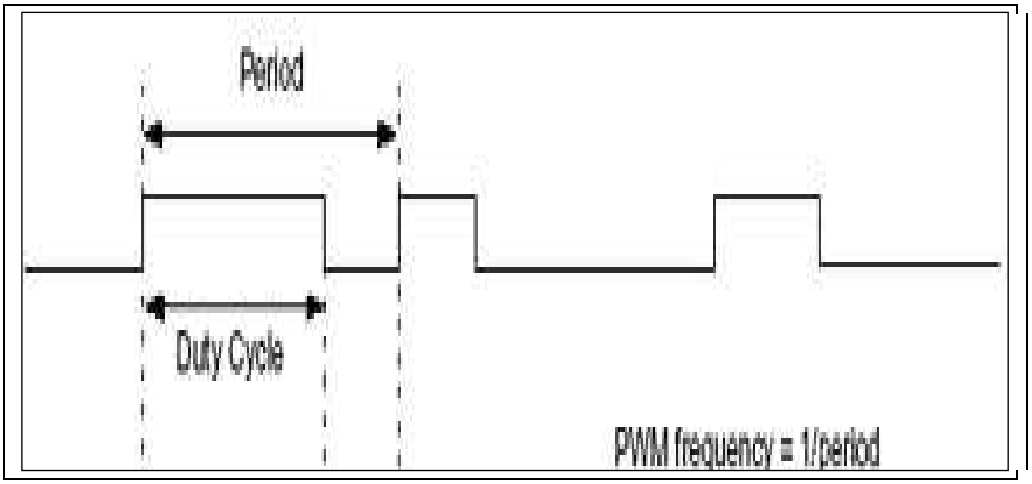


Fig.16

Un segnale PWM è caratterizzato dalla frequenza (fissa) e dal duty cycle (variabile); come si deduce dalla figura, il duty cycle è il rapporto tra il tempo in cui l'onda assume valore alto e il periodo T (l'inverso della frequenza: T=1/f). Ne segue che un duty cycle del 50% corrisponde ad un'onda quadra che assume valore alto per il 50% del tempo, un duty cycle dell'80% corrisponde ad un'onda quadra che assume valore alto per l'80% del tempo e

basso per il restante 20%, un duty cycle del 100% corrisponde ad un segnale sempre alto e un duty cycle dello 0% ad un segnale sempre basso.
Ora è necessario capire come applicare il segnale PWM al ponte H per controllare il motore, esamineremo due modalità: il PWM sign-magnitude e il PWM locked anti-phase.

6.1 Sign-magnitude pwm

Il pilotaggio SM (sign-magnitude) (schema di fig.17) consiste nell'inviare il segnale PWM all'ingresso di enable del ponte e di comandare la direzione di rotazione del motore tramite i due ingressi di controllo del ponte. Tali due ingressi devono essere comandati da segnali invertiti, in questo modo si riduce anche il numero di pin necessari per il controllo.

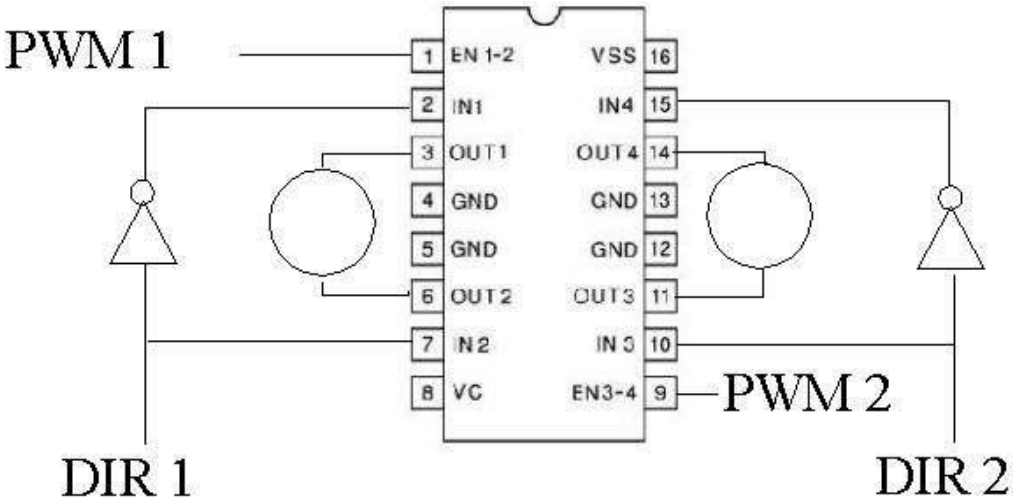


Fig.17

Per il controllo SM sono necessari quindi due segnali: il primo è un'onda quadra di duty cycle variabile tra 0 e 100% che stabilisce la velocità di rotazione, il secondo è

un segnale costante che determina il verso di rotazione (segnale basso rotazione in un verso, segnale alto rotazione nell'altro verso).

6.2 Locked anti-phase pwm

Il controllo LAP (locked anti-phase) si basa sulla stessa configurazione circuitale del controllo SM tuttavia i segnali di comandi sono applicati in modo diverso (fig.18) :

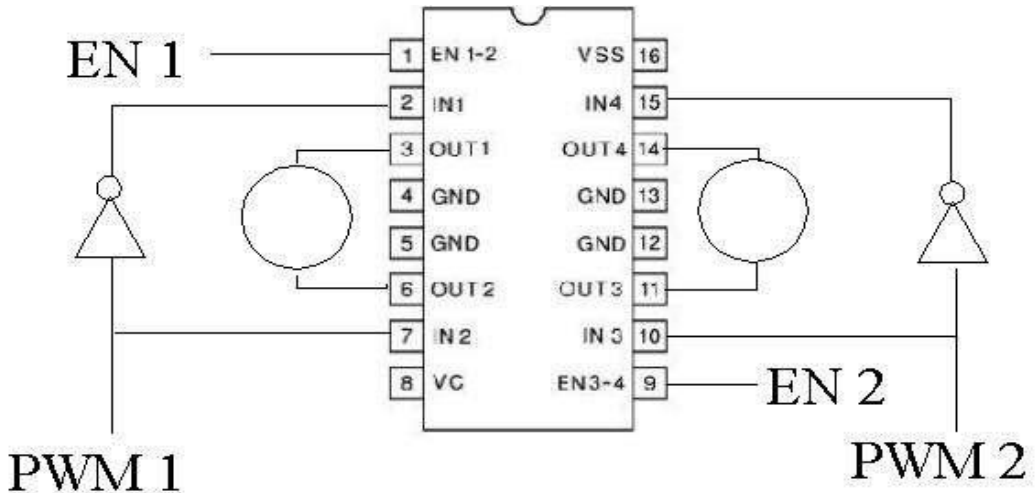


Fig.18

In questo caso il segnale PWM viene messo in ingresso all'invertitore in modo da avere ai due lati opposti del ponte due segnali invertiti tra loro; agendo sull'enable è possibile spegnere il rispettivo ponte.
Per il controllo LAP può bastare anche solo un segnale di comando(l'enable può essere fissato alto se non necessario) infatti l'onda quadra stabilisce sia la velocità che il verso di rotazione nel seguente modo:

• Duty cycle a 0 : rotazione alla massima velocità in un verso
• Duty cycle al 50%: motore fermo

• Duty cycle al 100%: rotazione alla massima velocità nell'altro verso

7.Algoritmo Line follower

Sfruttando i valori restituiti dai 3 sensori ottici riflessivi, il robot deve essere in grado di individuare l'andamento del tracciato e conseguentemente modificare la propria direzione per mantenersi sulla linea nera. Per farlo, è stato implementato un semplice algoritmo che si basa sulle rilevazioni dei sensori riflettevi.

Attraverso il linguaggio che usa Arduino UNO (linguaggio simile al Java), è stato trascritto l'algoritmo sottoforma di codice.

Il LFR è in grado di curvare con due diverse velocità, è cioè in grado di affrontare curve più o meno secche: direzione e intensità della curva sono ricavate dai valori letti dal sensore. Distinguiamo due diversi tipi di curve: curva a dx/sx o curva stretta a dx/sx, due diverse velocità dei motori (a seconda del duty cycle imposto): velocità max e velocità min, e diamo il nome S0 al sensore di sinistra, S1 al sensore centrale e S2 al sensore di destra. Arduino rileva i segnali, provenienti dai sensori attraverso gli ingressi analogici A0, A1 e A2, sottoforma di una tensione che può variare da 0V a 4.99V a seconda del colore della superficie analizzata: se la superficie è bianca la tensione rilevata da Arduino entra in un range compreso tra 1.50V (Vmin_bianco) e 1.74V (Vmax_bianco); se la superficie è nera la tensione rilevata da Arduino entra in un range compreso tra 0.13V (Vmin_nero) e 0.20V (Vmax_nero). Quindi viene spontaneo impostare una tensione di soglia che

distingua i due casi; nel nostro caso la soglia impostata è stata trovata attraverso il seguente calcolo:

Vsoglia = {[(Vmax_bianco+Vmin_bianco)/2] + [(Vmax_nero+Vmin_nero)/2]}/2

Il valore trovato è pari a : 0.89 V

Con queste premesse possiamo procedere ad illustrare l'algoritmo usato.
Il comportamento è il seguente:

- se solo il sensore centrale vede nero, il robot va dritto .
- se due sensori adiacenti vedono nero il robot curva lentamente nella direzione indicata (rispettivamente a sinistra e a destra).
- se solo un sensore laterale vede nero, il robot curva più bruscamente nella direzione del sensore.

In questo modo rispondiamo in maniera più marcata qualora il LFR si rendesse conto di essere prossimo a uscire dal percorso.

Consideriamo ad esempio il caso di una curva verso destra, partendo dalla condizione iniziale in cui il robot è centrato sulla linea nera:

1. nella fase precedente la curva il robot procede dritto, in quanto rileva che solo il sensore centrale è posto sulla linea;

2. entrando nella curva, ad un certo punto anche il sensore laterale destro si trova sopra il nastro, pertanto il robot capisce di trovarsi in una curva e comincia a sterzare lentamente verso destra attivando con velocità max il motore sx e con velocità min il motore dx (in questo modo la curva risulterà affrontata in modo più lineare e meno scattoso);

3. qualora la curva sia particolarmente decisa, la modifica alla traiettoria del robot non sarà sufficiente a mantenerlo nel circuito, il LFR dunque devierà verso l'esterno della curva fintantoché solo il sensore destro rileverà la linea nera; a questo punto l'algoritmo

prevede un'azione ancora più accentuata sui motori, in modo da diminuire il raggio di curvatura e quindi permettere al dispositivo di rientrare in pista (verrà attivato il motore sx con velocità max e verrà invertito il moto del motore dx impostandolo con velocità min).
I passi appena esposti sono rappresentati nelle seguenti immagini. Se la sterzata del robot risultasse sufficiente, dopo il passo 3 seguiranno nuovamente, in ordine, il passo 2 e 1, per tornare così alla condizione iniziale.

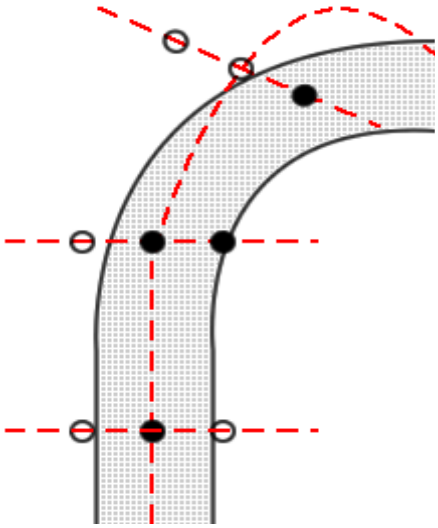


Fig.19

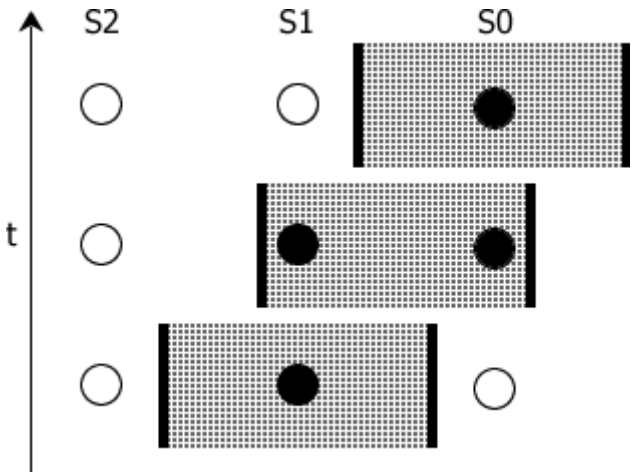


Fig.20

Per quanto riguarda le combinazioni non discusse (S2.S1.S0 == 000 ovvero 111 ovvero 010 dove per S = 0 diciamo che il sensore rileva la linea nera e per S = 1 il sensore rileva una superficie bianca), queste rappresentano situazioni anomale o casi particolari, la cui gestione richiede la conoscenza del comportamento passato del robot. Per questo motivo la reale implementazione dell'algoritmo è più complessa rispetto ad una semplice rete combinatoria. Nello specifico:

- **S2.S1.S0 == 000**

questa configurazione può essere ottenuta nel caso si incontri una intersezione lungo il tracciato, oppure, nel caso di una curva ad 'S' molto stretta, rientrando in modo quasi perpendicolare al nastro. Sia nel primo e nel secondo caso (presupponendo che ci stessimo muovendo di moto rettilineo o che il LFR stese eseguendo la curva in una certa direzione) si mantiene l'impostazione dei motori rilevata per ultima dai sensori (es: si continua a curvare a dx se l'ultima rilevazione imponeva di girare a dx).

- **S2.S1.S0 == 111**

La reazione a questa combinazione varia a seconda del momento in cui è rilevata: al reset del microcontrollore infatti questa ci indica che il robot è stato posto in una zona bianca, pertanto il LFR si muoverà di moto rettilineo fino a trovare la linea del tracciato.
Da questo momento l'interpretazione della combinazione 111 risulterà diversa, ossia significherà l'uscita dal tracciato in seguito ad una curva particolarmente stretta e ad essa dunque si reagirà come nel caso S2.S1.S0 == 000.

- **S2.S1.S0 == 010**

una rilevazione del genere si ha nel caso di biforcazione: il comportamento del robot è difficilmente predicibile in prossimità di un bivio, in quanto altamente influenzato dalla posizione dei sensori rispetto alla linea. A seconda infatti che il robot rilevi prima l'una o l'altra diramazione, la sua direzione sarà diversa; semplicemente allora manteniamo la

curvatura decisa dall'algoritmo nei passi precedenti. Nel caso in cui invece il bivio sopraggiunga mentre il robot si muove in linea retta, si decide arbitrariamente di svoltare a destra.

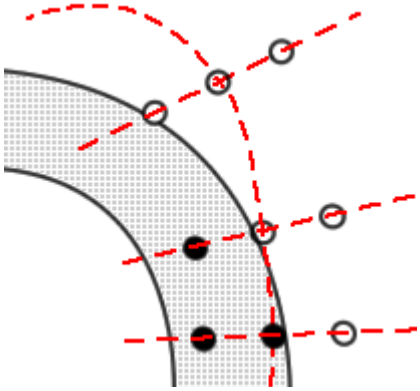


Fig.21

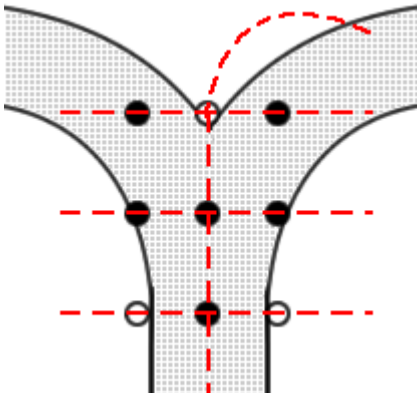


Fig.22

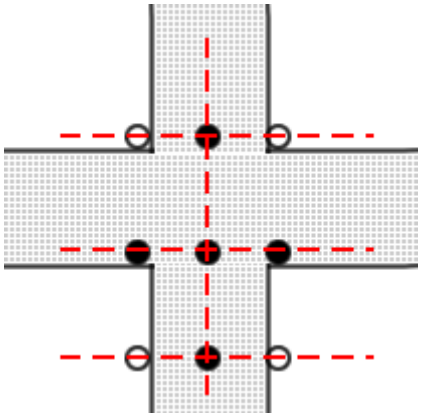


Fig.23

Riassumendo, risulta che gli elementi circuitali gestibili tramite l'algoritmo del Line follower sono complessivamente i seguenti:

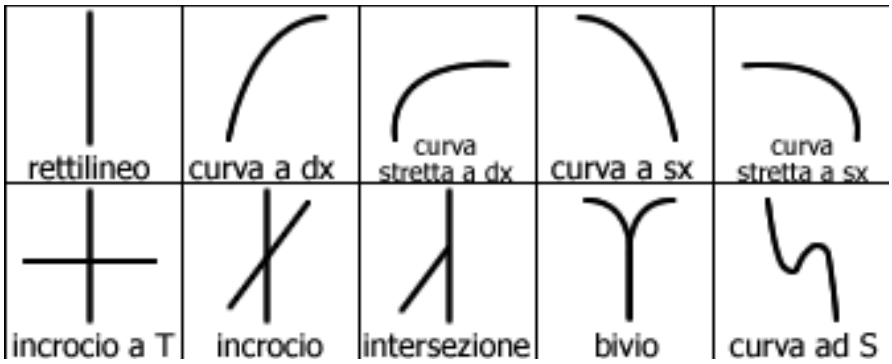


Fig.24

8.Codice

```
void setup ()
{
  Serial.begin(9600);
  pinMode(13, OUTPUT); //imposta il pin 13 come uscita
  pinMode(12, OUTPUT); //imposta il pin 12 come uscita
  pinMode(11, OUTPUT); //imposta il pin 11 come uscita
  pinMode(10, OUTPUT); //imposta il pin 10 come uscita
}
double sensore_0 ; // in " sensore_() " verrà inviato ad ogni ciclo il valore di tensione ristituito dal circuito dei sensori
double sensore_1 ;
double sensore_2 ;
double soglia=0.89; //imposto la soglia di tensione tra la lettura di una riflessione su superficie nera e di una riflessione su superficie bianca
double temp_sens_0;
double temp_sens_1;
double temp_sens_2;


int periodo=20;  // periodo onda quadra
int duty_cycle_max=20; // durata duty cycle per velocità max
int differenza_max=periodo-duty_cycle_max;


int duty_cycle_min=10; // durata duty cycle per velocità min
int differenza_min=periodo-duty_cycle_min;

int permanenza=2; // diviso  per 2 si trova il numero di periodi effettuati durante un operazione di curvatura o di tratto rettilineo
```

```
void loop()
{
  digitalWrite(13, HIGH); /* setto le uscite 13 e 12 ad un valore di tensione "HIGH", cioè pari a 5V (il robot appena acceso parte
                        //seguendo una traiettoria rettilinea, cioè va dritto di default) */
  digitalWrite(12, HIGH);

  digitalWrite(11, LOW);// setto le uscite 11 e 10 ad un valore di tensione "LOW", seguo una logica che è opposta tra uscite 13-12 e le uscite 11-10

  digitalWrite(10, LOW);



  int i;
  int p;
  for (i=0;i<100000;i++)
  {

    sensore_0 = (double)(analogRead(A0))*5/1023; // il comando "analogRead()" permette la lettura della tensione in entrata nei pin analocigi selezionati (A1,A2 e A3 nel nostro caso)
    sensore_1 = (double)(analogRead(A1))*5/1023;
    sensore_2 = (double)(analogRead(A2))*5/1023;



//                              sensore_0 = random(0.00,4.99);
  //                              sensore_1 = random(0.00,4.99);
    //                              sensore_2 = random(0.00,4.99);



        if ((sensore_0 <= soglia && sensore_1 <= soglia && sensore_2 <= soglia) || (sensore_0 > soglia && sensore_1 > soglia && sensore_2 > soglia))// i sensori
                                        // rilevano tutto nero o tutta superficie bianca

        {
          sensore_0=temp_sens_0;
          sensore_1=temp_sens_1;
          sensore_2=temp_sens_2;
        }



       // caso rettilineo
        if (sensore_1 <= soglia && sensore_0 > soglia && sensore_2 > soglia) // attivo entrambi i motori alla velocità max
```

```
{

        for (p=0;p<permanenza;p++)
        {
         if (p % 2 !=0 ) // segnale basso , numero dispari
         {
          digitalWrite(13, LOW);
          digitalWrite(12,LOW);
          digitalWrite(11, LOW);
          digitalWrite(10,LOW);
          delay(differenza_max);
         }
         else if (p % 2 == 0 ) //segnale alto , numero pari
         {
          digitalWrite(13, HIGH);
          digitalWrite(12,HIGH);
          digitalWrite(11, LOW);
          digitalWrite(10,LOW);
          delay(duty_cycle_max);
         }
       }
}

// caso curva a dx
else if (sensore_0 <= soglia && sensore_1 <= soglia && sensore_2 > soglia ) // attiverò il
motore di sx con velocità max, ed il motore di dx con velocità min
{
 for (p=0;p<permanenza;p++)
 {
  if (p % 2 !=0 ) // segnale basso , numero dispari
  {
   {
   digitalWrite(13, LOW);
   delay(differenza_max);
   }
   {
   digitalWrite(12,LOW);
   delay(differenza_min);
   }
   {
    digitalWrite(11, LOW);
    digitalWrite(10,LOW);
   }

  }
  else if (p % 2 == 0 ) //segnale alto , numero pari
  {
   {
   digitalWrite(13, HIGH);
```

```
          delay(duty_cycle_max);
          }

          {
          digitalWrite(12,HIGH);
          delay(duty_cycle_min);
          }
          {
            digitalWrite(11, LOW);
            digitalWrite(10,LOW);
          }

        }
       }

      }
      // caso curva a dx stretta
      else if (sensore_0 <= soglia && sensore_1 > soglia && sensore_2 > soglia) // attivo il motore
di sx con velocità max, e inverto il motore dx e lo imposto a velocità min
      {
       for (p=0;p<permanenza;p++)
       {
        if (p % 2 !=0 ) // segnale basso , numero dispari
        {
          {
          digitalWrite(13, LOW);
          delay(differenza_max);
          }
         {
          digitalWrite(10,LOW);
          delay(differenza_min);
         }
         {
           digitalWrite(12, LOW);
           digitalWrite(11,LOW);
         }

        }
        else if (p % 2 == 0 ) //segnale alto , numero pari
        {
          {
          digitalWrite(13, HIGH);
          delay(duty_cycle_max);
          }
          {
          digitalWrite(10,HIGH);
          delay(duty_cycle_min);
          }
          {
```

```
          digitalWrite(12, LOW);
          digitalWrite(11,LOW);
         }
        }
       }
      }

      // caso curva sx
      else if (sensore_0 > soglia && sensore_1 <= soglia && sensore_2 <= soglia ) // attivo il motore di dx con velocità max, ed il motore di sx con velocità min
      {
       for (p=0;p<permanenza;p++)
       {
        if (p % 2 !=0 ) // segnale basso , numero dispari
        {
         {
         digitalWrite(13, LOW);
         delay(differenza_min);
         }
        {
         digitalWrite(12,LOW);
         delay(differenza_max);
        }
        {
          digitalWrite(11, LOW);
          digitalWrite(10,LOW);
        }

        }
        else if (p % 2 == 0 ) //segnale alto , numero pari
        {
         {
         digitalWrite(13, HIGH);
         delay(duty_cycle_min);
         }
         {
         digitalWrite(12,HIGH);
         delay(duty_cycle_max);
         }
         {
          digitalWrite(11, LOW);
          digitalWrite(10,LOW);
         }
        }
       }
      }
      // caso curva a sx stretta
      else if (sensore_0 > soglia && sensore_1 > soglia && sensore_2 <= soglia ) // attivo il motore di dx con velocità max ed inverto il motore di sxve lo imposto a velocità min
```

```
      {
       for (p=0;p<permanenza;p++)
       {
        if (p % 2 !=0 ) // segnale basso , numero dispari
        {
          {
         digitalWrite(11, LOW);
         delay(differenza_min);
          }
         {
         digitalWrite(12,LOW);
         delay(differenza_max);
         }
         {
           digitalWrite(13, LOW);
           digitalWrite(10,LOW);
         }
        }
        else if (p % 2 == 0 ) //segnale alto , numero pari
        {
          {
         digitalWrite(11, HIGH);
         delay(duty_cycle_min);
          }
          {
         digitalWrite(12,HIGH);
         delay(duty_cycle_max);
          }
          {
           digitalWrite(13, LOW);
           digitalWrite(10,LOW);
          }
         }
       }
      }


      temp_sens_0=sensore_0;
      temp_sens_1=sensore_1;
      temp_sens_2=sensore_2;

                        Serial.println(sensore_0);
                        Serial.println(sensore_1);
                        Serial.println(sensore_2);

  }
}
```

9.Conclusioni

Per il collaudo del LFR (fig.25) si è creato un percorso misto con curve di media difficoltà da percorrere.
Si son usati i seguenti nella programmazione i seguenti parametri:

- Periodo onda quadra = 20ms (frequenza onda quadra = 50 Hz)
- Durata duty cycle per velocità max = 20ms (Duty cycle del 100%)
- Durata duty cycle per velocità min = 10ms (Duty cycle del 50%)

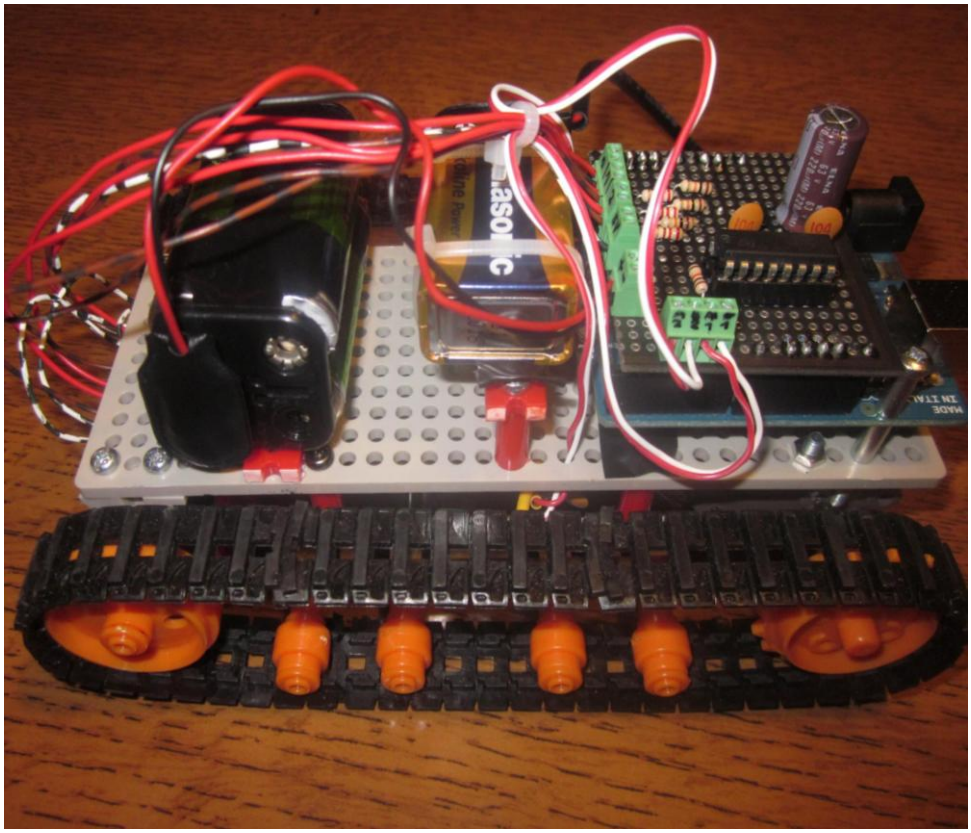


Fig.25

Per ogni lettura del segnale uscente dai sensori viene fatto passare un periodo di onda (nel codice è il parametro chiamato “permanenza”) prima di una nuova lettura, quindi impostando il periodo si determina anche la frequenza di lettura del segnale.

Si osserva che affrontando una curva “stretta” (mantenendo i parametri del duty cycle pari al 100% per velocità max e pari al 50% per velocità min) e diminuendo la frequenza di lettura del segnale (aumento cioè il periodo dell’onda quadra nel nostro caso) il LFR tende a non essere in grado di concludere la curva andando fuori traiettoria (va via “dritto” in curva). Per ovviare a questa incapacità si deve aumentare la frequenza di lettura del segnale o diminuire i parametri del duty cycle, in particolare per quanto riguarda il duty cycle per la velocità max (si porterà il valore dal 100% al 60-70%)con la conseguente però diminuzione di velocità di crociera del rover.

Un fattore importante che va ad influenzare anche il raggio di sterzata massimo accettabile è la distanza tra i sensori : mantenendo i sensori vicini tra di loro (13mm da asse ad asse

dei sensori) ,e tenendo presente che per effettuare una sterzata in curva stretta abbiamo a disposizione la configurazione di un cingolo che funziona a velocità max e del cingolo opposto che gira in senso contrario con velocità min e che la larghezza della linea nera è di 15mm, non riusciamo ad effettuare curve strette perché la staffa dove sono installati i sensori viene portata fuori "linea" e i sensori rilevano solo superficie bianca; nel caso opposto invece, se i sensori vengono installati ad una distanza elevata tra di loro osserviamo che il LFR anche nel affrontare un semplice rettilineo presenta un moto a "zig-zag" non accettabile.

Nel nostro caso abbiamo installato i sensori ad una distanza pari a 15mm (un buon compromesso) lasciando solo meno di 2 mm per eventuali oscillazioni non desiderate.

Infine il raggio di curvatura massimo del robot dipende anche da dov'è installata la staffa che supporta i 3 sensori: tanto più vicina al baricentro del LFR tanto più grande sarà il raggio di curvatura che si potrà effettuare.

Per quanto riguarda la risoluzione della nostra scheda, Arduino dispone del microcontrollore Atmega168 che contiene un integrato a 6 canali analogico-digitale (A / D). Il convertitore ha 10 bit di risoluzione, e quindi ritorna numeri interi da 0 a 1023. Nel nostro caso usiamo una tensione di riferimento pari a 5V che corrisponde quindi ad una risoluzione di 4.9mV/unità (5V/1023). Avendo solo due possibili stati d'ingresso (sup. bianca o nera) il range di valori trovati in corrispondenza di queste due configurazioni risulta ben distinto, e quindi anche lavorando con una risoluzione non delle migliori riusciamo ad ottenere un buon risultato anche con degli errori dovuti a disturbi di rumore e con delle imprecisioni nella linea da seguire.

10.Sviluppi futuri

La prima fase dello sviluppo del LFR è stata sviluppata durante questo studio di progettazione e creazione fisica del robot.

Si è visto come semplicemente cambiando frequenza di campionamento del segnale cambi in meglio o in peggio il comportamento del robot.

In futuro, nell'ambito delle misure, si vogliono studiare le reazioni del LFR in presenza di errori sui campioni d'ingresso, e di come essi possano modificare ed alterare la sua risposta; in particolare: errori in frequenza dovuti allo sfregamento delle spazzole nel statore dei motori, errori dovuti ad imprecisioni della superficie riflessa, errori di quantizzazione (diminuendo il numero di bit dell'ADC), errori di deriva ed errori di

guadagno applicando dei filtri appositi ed analizzando e studiando il segnale "sporco" con degli analizzatori di spettro.
Analizzare le tensioni provenienti dai due motori cc: discutere il fenomeno della commutazione.
Implementare un nuovo algoritmo più efficacie e che consenta al robot di distinguere più superfici di diversi colori e di diversi materiali e che riesca a superare e completare labirinti di elevata difficoltà con successo.
Applicare altri sensori che permettano di riconoscere pareti ed ostacoli di diversa specie per migliorare la sua autonomia fino ad ottenere un robot più completo possibile.

11.Bibliografia

- Studio del ponte h: http://digilander.libero.it/beamweb/ponte_h.htm

- Studio del controllo dei motori DC:

  http://www.dimeg.unipd.it/mecapp/materialepdf/meccatronica/motori.pdf

  http://www.tmasi.com/robotica/pwmtut/pwmtut_2.htm

  http://www.fabbrimarco.com/droboitalia/PWM_tutorial_1_0.pdf

  Dispense "Fondamenti di macchine ed azionamenti elettrici" [Cap.5(vers. 1.2), 6(vers.2.1), 7(vers.1.3)] prof. Mauro Zigliotto

- Studio della scheda ArduinoUNO e relativo linguaggio di programmazione:

  http://www.arduino.cc/

- Analisi componenti meccanici, motori DC, sensori riflettevi e datasheet:

  http://www.robot-italy.com/