UNIVERSITÀ DI BRESCIA
FACOLTÀ DI INGEGNERIA
Dipartimento di Ingegneria dell'Informazione

Laboratorio di Robotica Avanzata
Advanced Robotics Laboratory

Corso di Robotica
(Prof. Riccardo Cassinis)

Realizzazione di un robot olonomo

Elaborato di esame di:

Francesco Bocchi, Simone Brognoli

Consegnato il: **30 maggio 2014**

Sommario

Questo elaborato ha come obiettivo la realizzazione di un robot olonomo, utilizzando come unità centrale di controllo il Lego Mindstorm NXT. La parte di gestione e controllo del robot verrà implementata tramite un'architettura client/server. Il lato client si occuperà della gestione dell'input dell'utente, mentre il lato server si occuperà dell'interpretazione dei comandi e della loro esecuzione da parte del robot.

1. Introduzione

Questo progetto ha avuto come principale obiettivo la realizzazione e la programmazione di un robot olonomo. In laboratorio era disponibile un Lego Mindstorm NXT dal quale si è partiti.

1.1. Lego Mindstorm NXT

Fig. 1 - Lego NXT brick

Il Lego Mindstorm NXT è un kit robotico programmabile sia da computer che manualmente rilasciato dalla Lego alla fine di luglio 2006. Il componente principale del kit è il computer a forma di mattone chiamato "NXT brick". Può ricevere l'input da un massimo di quattro sensori e controlla fino a tre motori elettrici, attraverso cavi RJ12. Il NXT brick ha un display LCD monocromatico di 100x64 pixel e quattro pulsanti che possono essere utilizzati per navigare l'interfaccia utente a menu gerarchici. Esso ha anche un altoparlante che può riprodurre file sonori campionati a 8 kHz. La corrente è fornita da 6 batterie AA (1.5 V ognuna) oppure da una batteria ricaricabile Li-Ion.

1.1.1. Specifiche tecniche

- Microprocessore ARM7 32-bit (256 KB flash memory, 64 KB RAM).
- Microcontroller a 8-bit ATmega48 4 MHz (4 KB flash memory, 512 bytes RAM).
- CSR BlueCore 4 Bluetooth controller 26 MHz (Memoria flash esterna da 8 MBit, 47 KB RAM).
- Schermo LCD con matrice da 100×60 pixel.
- Altoparlante da 8 KHz di qualità del suono.
- Quattro porte di input a 6 pin per connettere sensori.
- Tre porte di output a 6 pin per connettere servo motori o lampadine.
- Connettività wireless Bluetooth (Classe II), per trasferire programmi all'NXT senza fili o per poter controllare il robot remotamente.
- Una porta USB 2.0.

- Il NXT brick è programmabile tramite il software Lego Mindstorm Education NXT software derivato da LabVIEW di Nation Instruments. Il programma è disponibile per i sistemi operativi Windows (da Xp service pack 2 in poi) e Mac OS X (10.3.9 o 10.4).

1.2. Ruote omnidirezionali

Per la realizzazione di un robot olonomo sono necessarie delle ruote omnidirezionali. Una ruota omnidirezionale possiede dei rulli attorno alla propria circonferenza, il cui senso di rotazione è perpendicolare alla direzione di rotolamento della ruota stessa. In questo modo la ruota può non solo rotolare come di consueto, ma anche scorrere lateralmente.

Per il progetto sono state utilizzate 3 ruote di 48mm di diametro e dotate di 8 rulli.

Fig. 2 - Ruota omnidirezionale

1.3. Libreria Java Lejos

Successivamente alla realizzazione della struttura del robot è stato affrontato il tema della sua programmazione. Esistono diversi software e librerie che permettono la programmazione e comunicazione con il Lego NXT. Dopo una attenta ricerca è stato deciso di utilizzare una libreria Java proveniente da Lejos NXJ. Lejos NXJ è un ambiente di sviluppo Java open source che permette di programmare il Lego Mindstorm. Consiste principalmente in:

- Firmware sostitutivo per il NXT che include una Java Virtual Machine.
- Una libreria di classi Java che implementano le API di Lejos NXJ.
- API per calcolatore che permettono di scrivere programmi in modo da consentire la comunicazione e controllo via USB, Bluetooth oppure utilizzando il protocollo di comunicazione Lego (LCP).

Per quanto riguarda gli scopi del progetto sono state utilizzate solamente le API per la comunicazione e controllo via calcolatore. Tutto il software realizzato è eseguito sul calcolatore che comanda direttamente il brick NXT. Così facendo non è stata necessaria né la sostituzione del firmware, né la scrittura di programmi appositi da eseguire sul brick NXT.

1.4. Scelta dell'ambiente di programmazione

Dato che le librerie di comunicazione con il robot sono scritte nel linguaggio di programmazione Java, è stato deciso di utilizzare questo linguaggio per l'intero sviluppo software, cioè sia per la parte client sia per la parte server. La versione di Java utilizzata è la 1.7.

1.5. Hardware utilizzato per lo sviluppo software

Il progetto è stato sviluppato su due calcolatori, in particolare:

- Ultrabook Hp envy 4 con processore Intel Core I5, da 1.7 GHz, 4 GB di RAM. Sistema operativo Windows 7.
- Ultrabook Asus s400 con processore Intel Core I5, da 1.7 GHz e 4GB di RAM. Sistema operativo Windows 8.

1.6. Client/Server

Per quanto riguarda l'aspetto della programmazione l'applicazione finale è stata struttura secondo un'architettura client/server.

Il client presenta un'interfaccia visuale atta a ricevere i comandi dell'utente, che verranno inviati al server.

Il server svolge le seguenti operazioni:

- Instaura una connessione tramite bluetooth con il robot;
- Riceve i comandi dal client tramite il protocollo TCP utilizzando la porta 7777.
- In base ai comandi ottenuti aziona i motori del robot.

2. Il problema affrontato

In questo paragrafo saranno descritti i diversi problemi affrontati per la realizzazione del robot omnidirezionale.

2.1. La struttura del robot

Il primo problema che è stato affrontato è stato la creazione della struttura di supporto al NXT brick che permettesse l'olonomia del robot. La struttura prevedeva 3 ruote omnidirezionali, ognuna delle quali collegata ad un motore dedicato.

2.2. Il movimento delle ruote

Per permettere al robot di muoversi in ogni direzione è stato necessario calcolare in modo adeguato le velocità delle singole ruote. Infatti ogni volta che il robot riceve un comando di movimento da eseguire, deve essere in grado di calcolare la corretta velocità delle tre ruote affinché il movimento venga svolto. I calcoli non vengono svolti dal NXT brick, ma dalla applicazione server che, una volta risolti, comanda direttamente i motori del robot.

2.3. Comunicazione dei comandi al robot

Il client deve collegarsi al server per poter interagire con il robot. Il server riceve i comandi dal client, li traduce in comandi per l'unità centrale di controllo del robot, e glieli invia.

La comunicazione tra server e client viene fatta tramite TCP/IP, mentre la comunicazione tra server e il robot avviene tramite Bluetooth.

3. La soluzione adottata

Di seguito vengono esposte le soluzione adottate per risolvere i problemi sopra esposti.

3.1. La struttura del robot

La prima struttura è stata realizzata utilizzando solamente materiale Lego. L'architettura avrebbe imposto che le ruote venissero a trovarsi ai vertici di un triangolo equilatero, ma l'esigenza di montare i motori ha imposto la creazione di una struttura esagonale.

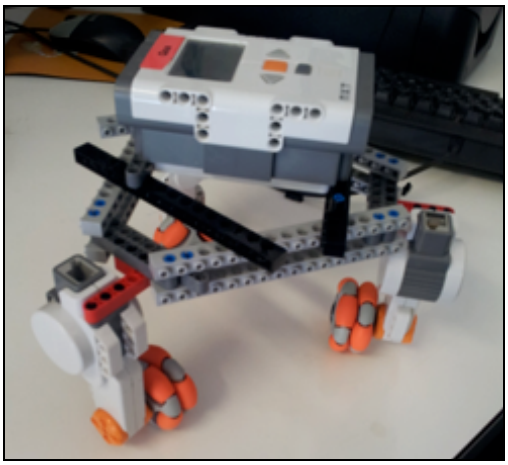

Fig. 3 - Prima versione della struttura del robot

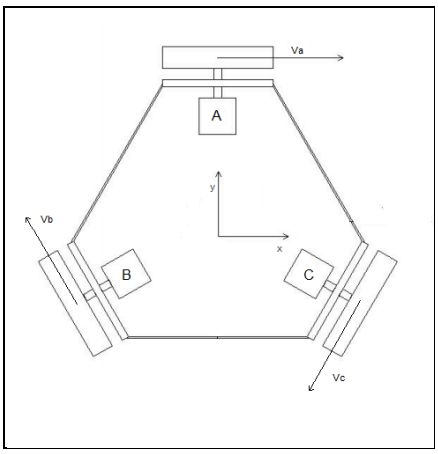

Fig. 4 - Modello del robot

La struttura realizzata presentava tuttavia dei problemi, infatti era troppo poco rigida e troppo alta. Questi due fattori influenzavano il robot durante i suoi movimenti causando imprecisioni troppo elevate. Si è quindi passati a una struttura triangolare costituita da materiale plastico. La base era più bassa rispetto alla precedente e i motori sono stati fissati alla struttura tramite l'ausilio di viti e bulloni. Grazie a questi accorgimenti la struttura è risultata essere molto più rigida. Infine le ruote sono state disposte al centro di ogni lato.

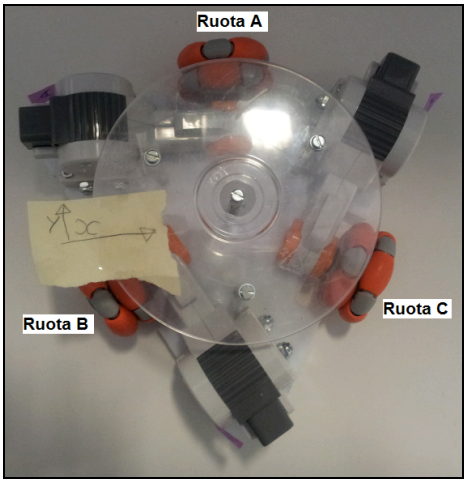

Fig. 5 - Versione definitiva della struttura del robot

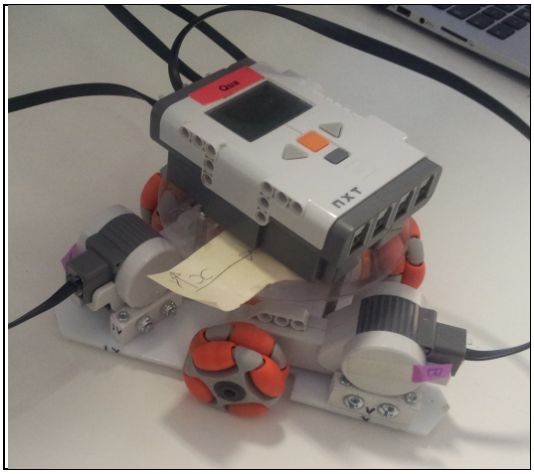

Fig. 6 - Versione definitiva della struttura del robot

3.2. Il movimento delle ruote

L'obiettivo era il calcolo delle velocità delle singole ruote data la velocità che il robot doveva assumere, ciò ha trovato soluzione nella determinazione della trasformazione cinematica inversa.

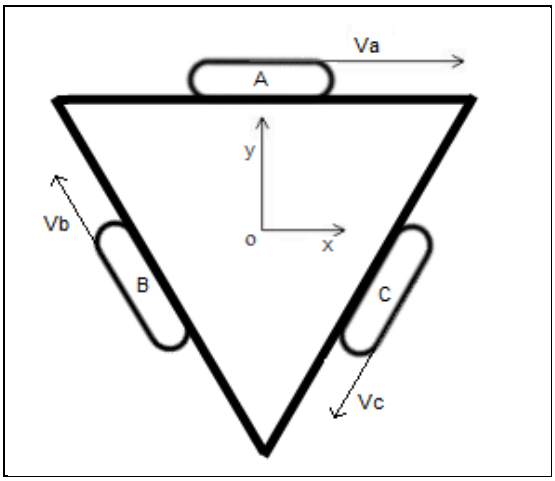

Fig. 7 - Modello del robot. L'origine degli assi si trova nel circocentro del triangolo

Dato lo schema in Fig. 7 - le equazioni che risolvono il problema cinematico sono le seguenti:

$$\begin{cases} Va = \omega a * r = Vx + \omega R \\ Vb = \omega b * r = -\frac{1}{2}Vx + \frac{\sqrt{3}}{2}Vy + \omega R \\ Vc = \omega c * r = -\frac{1}{2}Vx - \frac{\sqrt{3}}{2}Vy + \omega R \end{cases}$$

Legenda

Vi = velocità tangenziale della ruota i.

Vx, Vy = Velocità rispetto al sistema di riferimento

ωi = velocità angolare della ruota i.

ω = velocità angolare di rotazione del robot rispetto al proprio centro

r = raggio della ruota. Valore 24 mm.

R = distanza tra il centro del robot e le ruote. Valore 67.25 mm.

3.3. Connessione con il robot

Per connessione con il robot si intende la comunicazione client/server e l'invio dei comandi dall'applicazione lato server al robot.

3.3.1. Comunicazione client/server

Essendo il programma scritto in Java, per permettere la comunicazione tra client e server si è deciso di utilizzare un socket su protocollo IP che utilizzasse come protocollo di trasporto il TCP. Per la comunicazione di rete Java offre il modello a stream. Esistono due tipi di stream per un socket: uno per l'input e uno per l'output. Per inviare i dati attraverso la rete un processo scrive sullo stream di output associato al socket, mentre dall'altra parte un altro processo leggerà lo stream di input, associato allo stesso socket.

Per iniziare la comunicazione il server viene eseguito per primo e rimane in attesa di una connessione sulla porta 7777. In caso tale porta risulti già occupata il server si preoccupa di cambiare porta aumentando di un'unità il numero di porta da provare, fino a un massimo di 20 tentativi.

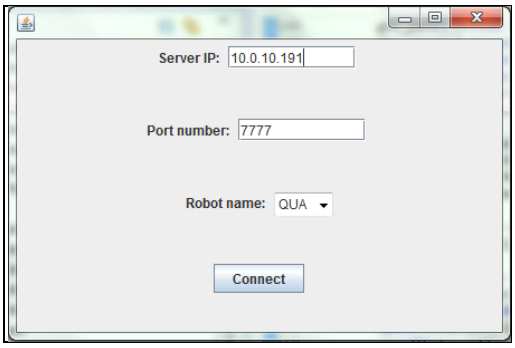

Fig. 8 - Schermata di connessione al server

Il client una volta che il server ha inizializzato la connessione con successo è pronto per essere eseguito.
I parametri da configurare sono: l'indirizzo Ip del calcolatore sul quale è attivo il programma server, il numero di porta sulla quale è aperta la connessione e il nome del robot con cui il server si connetterà via Bluetooth.
Una volta instaurata la connessione client/server il server si preoccupa immediatamente di avviare il collegamento tramite Bluetooth con il robot il cui nome è stato fornito in precedenza.

Dopo l'avvio della connessione al client si presenta la schermata mostrata in Fig. 9 -

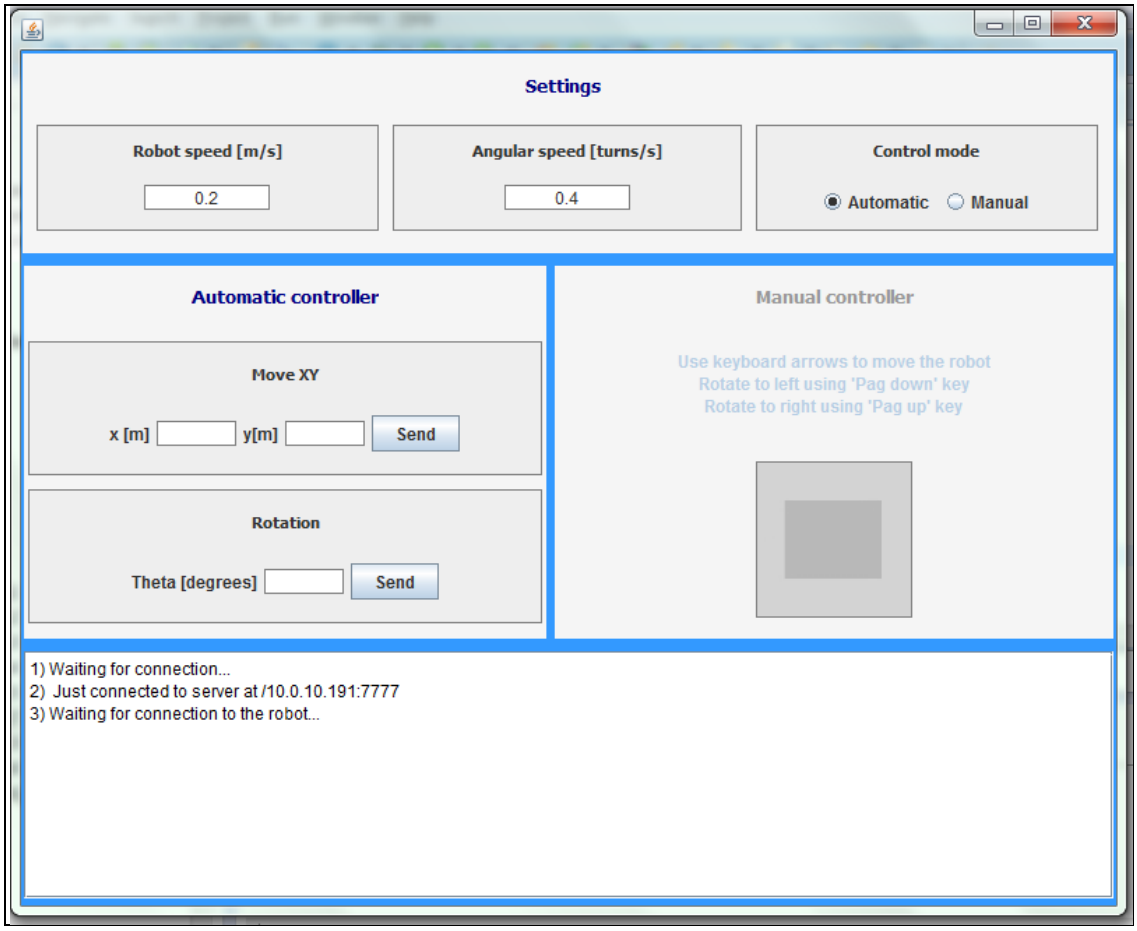

Fig. 9 - Finestra di controllo del robot

Nella parte alta sono presenti tre parametri per la configurazione del movimento:

- Robot speed [m/s]: rappresenta il modulo della velocità del robot;
- Angular speed [giri/sec]: rappresenta il modulo della velocità angolare del robot;
- Control mode: indica la possibilità di controllare il robot tramite comando manuale o automatico.
    - Controllo manuale: utilizzando i tasti freccia della tastiera e tasti pag (per le rotazioni) è possibile muovere direttamente il robot.
    - Controllo automatico: permette al robot di muoversi di una quantità di spazio (o di effettuare una rotazione) fissata.

Nella parte centrale è possibile, per quanto riguarda il controllo automatico, definire la x e y (in metri) di spostamento del robot. Inoltre è possibile inviare al robot anche una determinata rotazione, espressa in gradi, da eseguire. Per la modalità manuale è invece mostrato un feedback dei tasti premuti delle azioni compiute dal robot.

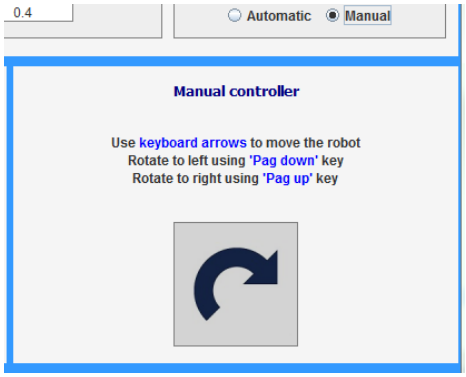

Fig. 10 - Controllo Manuale

Infine nella parte bassa è presente una finestra che mostra i risultati delle azioni compiute.

Tramite l'interazione dell'utente con gli elementi dell'interfaccia, il client invia al server dei comandi che verranno poi tradotti in azionamenti dei motori del robot. I comandi inviati presentano la seguente sintassi: nome Comando@param1@parm2....@parmaN.
I possibili comandi sono:

- robotName@nomeRobot: indica al server il nome del robot al quale collegarsi.
- sendXY@x@y@speed: movimento automatico del robot di x e y metri rispetto al suo sistema di riferimento a una velocità speed.
- sendTheta@theta@angularSpeed: rotazione automatica del robot di un angolo theta a una velocità angularSpeed.
- manualCommand@direction@speed: movimento manuale del robot nella direzione indicata in direction. Sono comprese anche le rotazioni.
- manualStop: arresto dei motori.
- closeConnection: chiusura della connessione con il server.

3.3.2. Comunicazione server/robot

Il server comunica con il robot tramite Bluetooth. Le librerie utilizzate sono la libreria BlueCove versione 2.1.1 per permettere la connessione Bluetooth, e la libreria Lejos per comandare il robot via calcolatore.

Una volta ricevuti i comandi dal client il server si preoccupa di calcolare le velocità delle singole ruote. Una volta calcolate le velocità di rotazione delle ruote, il server le comunica al robot. Inoltre, in caso di controllo automatico, il server imposta anche il tempo di durata delle rotazioni dei motori.

- **I metodi Lejos utilizzati per agire sui motori sono stati forward() e backward(), che permettono la rotazione in avanti e indietro del motore; tali metodi instaurano un moto perpetuo finché non viene richiamato il metodo stop(). Sarebbe stato più intuitivo utilizzare il metodo rotateTo(gradi) che effettua la rotazione del motore di un angolo specificato come parametro, ma sono stati notati degli errori nel suo funzionamento. Infatti applicando consecutivamente due rotazioni di verso opposto, la seconda rotazione non esegue la quantità di gradi specificata.**

4. Modalità operative

È possibile agire in due modi distinti per utilizzare quanto esposto in questo elaborato.

- Modalità esecuzione: permette di utilizzare i programmi client e server immediatamente.
- Modalità progetto: permette l'importazione dei codici sorgenti di client e server tramite l'ambiente di sviluppo Eclipse.

4.1. Componenti necessari

4.1.1. Modalità esecuzione

- Java versione 7 o superiore scaricabile al link: https://www.java.com/it/download/
- LegoOmniClient.jar, applicazione client che si trova nella cartella *jars*
- LegoOmniServer.jar, applicazione server che si trova nella cartella *jars*
- Antenna Bluetooth per il calcolatore che esegue l'applicazione server.

4.1.2. Modalità progetto

- Java versione 7 o superiore scaricabile al link: https://www.java.com/it/download/
- Antenna Bluetooth per il calcolatore che esegue l'applicazione server.
- Eclipse scaricabile al link: http://www.eclipse.org/downloads/
- La cartella LegoOmniServer contenente il progetto sviluppato in Eclipse. Si trova nella cartella *eclipseProject.*
- La cartella LegoOmniClient contenente il progetto sviluppato in Eclipse. Si trova nella cartella *eclipseProject.*

I progetti di Eclipse utilizzano al loro interno le librerie Bluecove e Lejos. Non è necessario scaricarle in quanto sono già contenute nei progetti. Tuttavia sono reperibili ai seguenti link.

- Bluecove versione 2.1.1: http://bluecove.org/
- Lejos versione 2.1.0 http://www.lejos.org/rcx-downloads.php

4.2. Modalità di installazione

È necessario anzitutto associare al calcolatore che eseguirà il programma server, il NXT brick via Bluetooth.

1. Accendere la connessione Bluetooth sul NXT brick. Assicurarsi che il Bluetooth sia visibile.
2. Attivare il Bluetooth sul calcolatore e avviare la ricerca di dispositivi. Per Windows selezionare Pannello di controllo → Dispositivi e Stampanti → Aggiungi dispositivo Bluetooth.
3. Seguire la procedura guidata. Durante la procedura verrà richiesto un codice di autenticazione. Il codice verrà visualizzato sullo schermo del NXT brick.

4.2.1. Modalità esecuzione

1. Aprire la console dei comandi.
2. Posizionarsi nella cartella *jars.*
3. Eseguire il seguente comando per lanciare il programma server: *java -jar LegoOmniServer.jar*
4. Attendere l'avvio del programma.
5. Eseguire il seguente comando per lanciare il programma client: *java -jar LegoOmniClient.jar*

➢ **Attenzione: se i programmi client e server sono eseguiti sullo stesso calcolatore è necessario aprire due finestre di comando diverse.**

4.2.2. Modalità progetto

1. Aprire Eclipse.
2. Importazione del progetto server
    a. Selezionare la voce *File* → *Import.*
    b. Selezionare la voce *Existing Projects into Workspace.*

   c. Selezionare il progetto LegoOmniServer (che si trova nella eclipseProject) tramite il pulsante *Browse…*
   d. Spuntare la casella *Copy projects into workspace.*
   e. Premere il tasto *Finish*
3. Importazione del progetto client
   a. Selezionare la voce *File → Import.*
   b. Selezionare la voce *Existing Projects into Workspace.*
   c. Selezionare il progetto LegoOmniClient (che si trova nella eclipseProject) tramite il pulsante *Browse…*
   d. Spuntare la casella *Copy projects into workspace.*
   e. Premere il tasto *Finish*

4.3. Avvertenze

Il NXT brick è influenzato dal livello di batteria, infatti con valori bassi di carica residua i motori non riescono a lavorare a piena potenza e quindi non sono in grado di supportare determinate velocità.

Con batteria pienamente carica ogni motore riesce a sostenere una velocità angolare di circa 720 gradi al secondo. Più la batteria si scarica e più questo valore diminuisce.

5. Conclusioni e sviluppi futuri

In questo elaborato è stato realizzato un robot olonomo tramite l'utilizzo di Lego Mindstorm. Dopo una prima fase di realizzazione della struttura, l'attenzione è stata posta principalmente sulla sua programmazione. In particolare è stata realizzata un'architettura client/server per il controllo remoto del robot: il lato client presenta un'interfaccia grafica per l'invio di comandi al server tramite TCP/IP; il lato server riceve i comandi dal client e, interpretandoli, li invia al robot tramite Bluetooth per essere eseguiti. Esistono due modalità di controllo: manuale, il robot segue una direzione (o rotazione) fino al comando di stop; automatica, il robot compie una distanza (o rotazione) definita. L'intero software è stato scritto nel linguaggio di programmazione Java.

Uno sviluppo futuro potrebbe essere l'aggiunta di sensori al robot, in quanto in questo momento ne è totalmente sprovvisto. Un esempio di sensore da associare al robot potrebbe essere una telecamera per permetterne la visione, oppure un sensore di prossimità per evitare che il robot vada a urtare altri oggetti senza rendersene conto. Per quanto riguarda la telecamera potrebbe essere interessante implementare una visione dall'alto per fornire feedback, in modo da migliorare la precisione del movimento del robot.

Bibliografia

[1] Borestein, J., Everett, H.R., Feng, L.: "Where am I?", University of Michigan, 1996.

[2] Baede, T.A.: "Motion control of an omnidirectional mobile robot", National University of Singapore and Eindhoven University of Technology, Eindhoven , Settembre 2006. http://alexandria.tue.nl/repository/books/633499.pdf

[3] Foglio, S., Meneghello, M.: "M.A.R.M.O.T Robot omnidirezionale", Università degli Studi di Brescia, Brescia, Maggio 2001.

[4] Documentazione API Lejos per calcolatore: http://www.lejos.org/nxt/pc/api/index.html

Indice

SOMMARIO **1**

1. INTRODUZIONE **1**
1.1. Lego Mindstorm NXT **1**
1.1.1. Specifiche tecniche 1
1.2. Ruote omnidirezionali **2**
1.3. Libreria Java Lejos **2**
1.4. Scelta dell'ambiente di programmazione **2**
1.5. Hardware utilizzato per lo sviluppo software **2**
1.6. Client/Server **3**

2. IL PROBLEMA AFFRONTATO **3**
2.1. La struttura del robot **3**
2.2. Il movimento delle ruote **3**
2.3. Comunicazione dei comandi al robot **3**

3. LA SOLUZIONE ADOTTATA **3**
3.1. La struttura del robot **3**
3.2. Il movimento delle ruote **4**
3.3. Connessione con il robot **5**
3.3.1. Comunicazione client/server 5
3.3.2. Comunicazione server/robot 8

4. MODALITÀ OPERATIVE **8**
4.1. Componenti necessari **9**
4.1.1. Modalità esecuzione 9
4.1.2. Modalità progetto 9
4.2. Modalità di installazione **9**
4.2.1. Modalità esecuzione 9
4.2.2. Modalità progetto 9
4.3. Avvertenze **10**

5. CONCLUSIONI E SVILUPPI FUTURI **10**

BIBLIOGRAFIA **10**

INDICE **11**