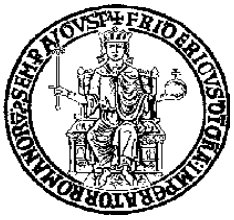
Università degli Studi di Napoli "Federico II"

Dipartimento di Scienze Relazionali "G. Iacono"

Scuola di Dottorato di ricerca in Scienze Psicologiche e Pedagogiche

Indirizzo: Analisi dei Processi Psicologici Normali e Patologici

XXIII CICLO
(A.A. 2007/2008- 2009-2010)



Robotica Educativa:
Sviluppo di un Ambiente di Apprendimento Basato su Robot Lego

Tesi di dottorato di **Maria Assunta Calabretta**

TUTOR

Prof. Orazio Miglino

COORDINATORE

Prof. Orazio Miglino

Anno Accademico

2009/2010

Indice

Introduzione

L'utilizzo della robotica in ambito educativo costituisce oggi un'area di ricerca consolidata anche se relativamente recente. In questa tesi, esploreremo la possibilità di utilizzare i robot come modelli artificiali di creature viventi che possano consentire di acquisire una visione sistemica di aree concettuali (quali ad es. sistema sensoriale, sistema nervoso, sistema motorio, interazione agente/ambiente, apprendimento) difficile da veicolare con metodi alternativi. Tale obiettivo è stato perseguito sviluppando un ambiente di apprendimento chiamato EduBot basato su prototipi hardware e software precedentemente sviluppati presso l'Istituto di Scienze e Tecnologie della Cognizione del CNR di Roma e l'Università Federico II di Napoli.

La tesi è strutturata nel modo seguente. Nel primo capitolo introduciamo la Robotica Educativa. Nel secondo capitolo, gli ambienti di apprendimento più consolidati costituiti da robot o kit robotici, software, e materiale curricolare. Nel terzo capitolo descriviamo EduBot l'ambiente di apprendimento sviluppato nell'ambito di questa tesi. Nel quarto capitolo descriviamo il materiale curricolare sviluppato per tale ambiente. Infine, nella sezione conclusiva, riassumiamo il lavoro fatto e le prospettive per il futuro.

Il lavoro svolto non sarebbe stato possibile senza il contributo del Dott. Onofrio Gigliotta, che ha contribuito in modo significativo allo sviluppo dell'ambiente software, del Dott. Valerio Sperati, che ha contribuito allo sviluppo del prototipo hardware, del Prof. Orazio Miglino che ha supervisionato e indirizzato il lavoro.

1. La Robotica Educativa

Per robotica educativa (Miglino, Lund, Cardaci, 1999; Druin & Hendler, 2000; Martin, 2001; Miller, Nourbakhsh, Siegwart, 2008) si intende lo sviluppo e l'utilizzo di ambienti di apprendimento basati su tecnologie robotiche. Tali ambienti sono costituiti di norma da robot (macchine più o meno complesse dotate di sensori, attuatori, e computer di bordo), software (utilizzato in particolare per programmare il computer di bordo del robot), e materiale curricolare. In questo capitolo ripercorreremo l'origine e lo sviluppo di questa area di ricerca piuttosto recente, i punti di forza e i problemi aperti.

1.1 Learning by doing

L'origine dell'interesse per l'uso di tecnologie robotiche in ambito educativo può essere ricondotto innanzi tutto al costruzionismo elaborato da Saymour Papert (1980) a partire dalle idee di Jean Piaget con cui Papert collaborò agli inizi della sua carriera scientifica. In linea con le idee sviluppate da Jean Piaget, il costruzionismo assume che la conoscenza non deve essere semplicemente trasmessa dall'insegnate allo studente, ma costruita attraverso un processo attivo in cui lo studente costruisce le proprie rappresentazioni mentali interagendo con la realtà esterna. Papert estende questa idea mettendo in evidenza il fatto che lo studente apprende con particolare efficacia mentre è coinvolto in un processo creativo che riguarda la costruzione di qualcosa che è motivante ed interessante dal suo punto di vista. In questo tipo di situazioni gli studenti imparano ad analizzare problemi che non hanno un risposta pre-determinata e che permettono loro di sviluppare delle soluzioni nuove in modo creativo. Ciò avviene attraverso un processo di sperimentazione e modellizzazione in cui gli studenti manipolano la realtà esterna, analizzano quello che vedono, e poi assimilano le osservazioni fatte nei loro modelli interni, oppure cambiano i propri modelli mentali al fine di renderli compatibili con le nuove osservazioni.

Tale processo è influenzato dalla disponibilità di artefatti cognitivi, ovvero oggetti e dispositivi concreti che facilitano lo sviluppo di specifici apprendimenti. Per costruire le proprie conoscenze, il discente ha bisogno di materiali da costruzione appropriati che possano essere esaminati, manipolati, mostrati, discussi, sondati e ammirati. La lentezza

dello sviluppo di un particolare concetto da parte del bambino può dunque non essere dovuta alla maggiore complessità o formalità, ma alla povertà della cultura di quei materiali che renderebbero il concetto semplice e concreto.

Un modo per migliorare il processo di apprendimento consiste dunque nel cercare di creare strumenti e ambienti che motivino gli studenti ad intraprendere attività di costruzione, invenzione e sperimentazione. Per questa ragione Papert, al fine di tradurre questi presupposti teorici in pratica, si occupò direttamente dello sviluppo di ambienti di apprendimento quali il LOGO e i mattoncini programmabili che costituiscono ancora oggi gli ambienti di apprendimento più utilizzati in ambito educativo.

Negli anni 70 Papert sviluppa il LOGO, un linguaggio di programmazione molto semplice che ha conosciuto una larga diffusione nel mondo dell'educazione. Sebbene il LOGO non preveda di norma l'utilizzo di tecnologie robotiche esso presenta alcune analogie importanti con la robotica e con gli ambienti di robotica educativa sviluppati successivamente. Operare in ambiente Logo, infatti, significa programmare una piccola tartaruga che si muove sullo schermo del computer in risposta a dei nostri comandi. Analogamente ad un robot, tale tartaruga è dotata di un orientamento che influenza gli effetti delle azioni che esegue, è situata in un ambiente esterno, ed è in grado di variare la propria posizione relativa rispetto all'ambiente o l'ambiente stesso (in particolare disegnando delle linee durante i movimenti effettuati). Non a caso, successivamente, Papert realizzo anche una versione robotica della tartaruga che era collegata al computer attraverso un cavo (Figura 1.1).

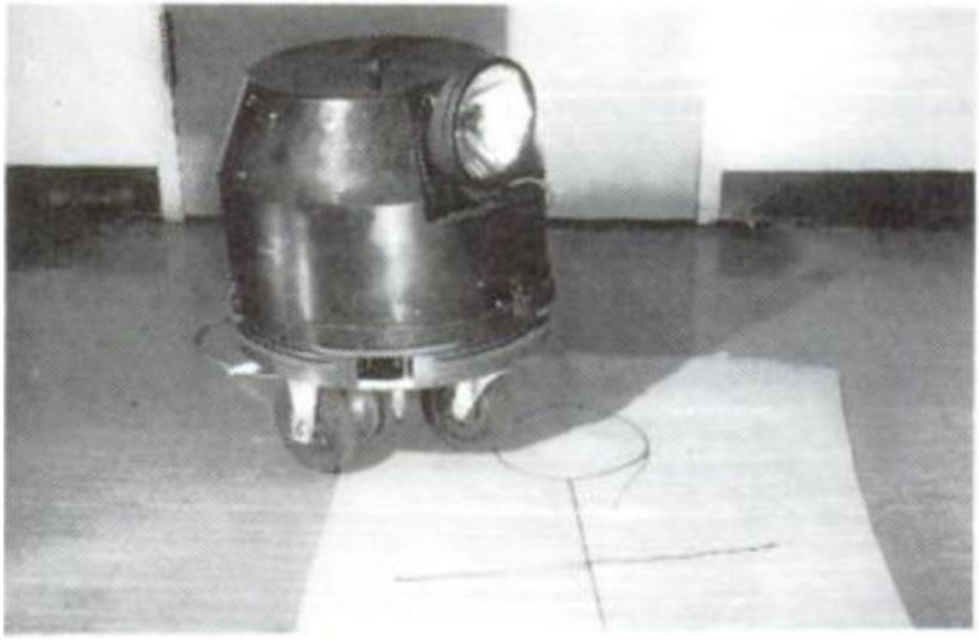


Figura 1.1. Il robot "tartaruga" costruito da Papert. Quella che segue è la descrizione dell'autore: "[The] Figure shows one of our turtles—so names in honor of a famous species of cybernetic animla made by Grey Walter, an English neurophysiologist. Grey Walter turtle had life-like behavior patterns built into its wiring diagram. Ours have no behavior except the ability to obey a few simple commands from a computer to which they are attached by a wire that plugs into a control-box that connect to a telephone line that speaks to the computer, which thinks it is talking to a teletype so that no special system programming is necessary to make the computer talk to the turtle. (If you'd like to make a fancier turtle, you might use a radio link. But we'd like turtles to be cheap enough to every kid play with one.)."

Papert ebbe un ruolo cruciale anche nello sviluppo dei LEGO Mindstorm, il primo ambiente di robotica educativa che ha conosciuto una larga diffusione. In effetti tale ambiente è il risultato dell'attività congiunta del laboratorio di ricerca "Epistemology and Learning Group" dell'MIT (che includeva tra i suoi membri Symour Papert, Martin Resnick e altri collaboratori) e l'industria di giocattoli danese LEGO (Mindell et al., 2000). L'obiettivo che si proposero inizialmente Papert e Resnick fu quello di sviluppare un nuovo tipo di giocattolo. L'dea era quella di sviluppare un set di costruzione che potesse consentire ai bambini di costruire oggetti e macchine analoghe a quelle che potevano essere sviluppate con i kit LEGO dell'epoca ma che al tempo stesso fornissero ai bambini la possibilità di animare le proprie costruzioni. Per questa ragione decisero di cercare di estendere i kit LEGO. Così cominciarono a sviluppare in collaborazione con la LEGO un mattoncino programmabile con in linguaggio LOGO (Figura 1.2, sinistra). Per quanto

riguarda l'ambiente di programmazione furono sperimentate diverse opzioni, inclusi ambienti di programmazione testuali tradizionali che tuttavia, pur essendo molto potenti, richiedevano ai ragazzi un sforzo iniziale relativamente alto. Per questa ragione, Resnick e collaboratori implementarono LogoBlocks, una versione grafica del linguaggio in cui le istruzioni sono costituite da icone/comandi che possono essere posizionati sullo schermo (Figura 1.2, destra).

Sulla base dell'accordo siglato tra l'MIT e la LEGO, i ricercatori dell' Epistemology and Learning Group ebbero la possibilità di pubblicare liberamente il design e la descrizione dei prodotti sviluppati. La LEGO invece, quale sponsor del Laboratorio, si riservava di sfruttare commercialmente i prodotti sviluppati, cosa che si concretizzò pochi anni dopo con la commercializzazione dei LEGO Mindstorms (che descriveremo nel capitolo successivo) disegnati sostanzialmente sulla base dell'ambiente hardware e software sviluppato all'MIT.

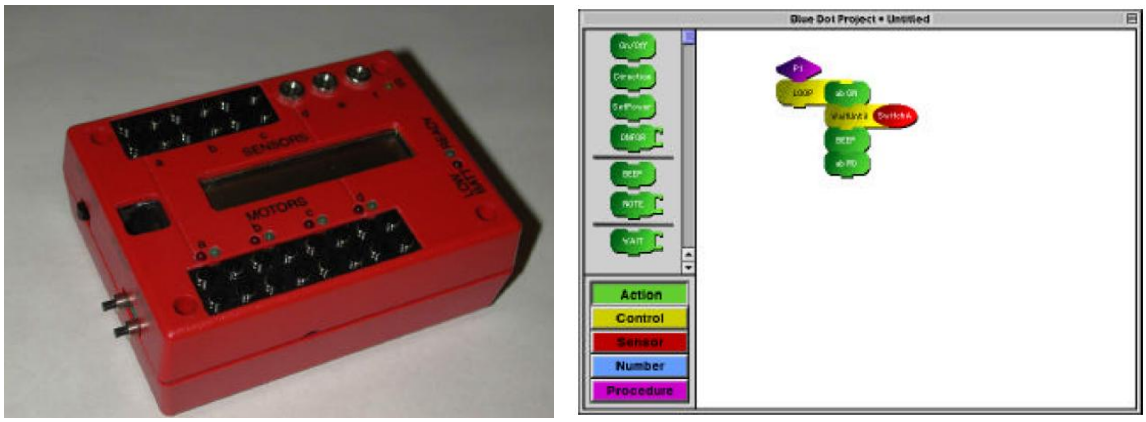


Figura 1.2. Sinistra: Una versione del mattoncino programmabile. Destra: Schermata dell'ambiente di programmazione LegoBLOCKS. Sia il mattoncino che l'ambiente software sono stati sviluppati presso lo Epistemology and Learning Group del MIT Media Laboratory.

L'utilizzo di ambienti di apprendimento di questo tipo permette di creare una situazione in cui gli studenti si comportano come se fossero dei veri scienziati, inventori o ingegneri. Di conseguenza essi entrano in contatto diretto con gli aspetti realmente cruciali delle scienze, dell'ingegneria e della matematica. Inoltre essi non si limitano ad apprendere nozioni, equazioni e metodi ma apprendono ad esercitare forme di pensiero critico e capacità di risoluzione dei problemi.

L’utilizzo di tali strumenti in classe funziona come comunità di pratiche scientifiche in cui gli studenti comunicano e condividono le loro idee, giuste o sbagliate che siano. Si discute ed ognuno apprende dall'altro. Alcune delle idee proposte possono risultare valide, altre un po' meno, ma comunque tutti gli allievi partono da uno stesso piano: ogni idea ha la stessa dignità.

Inoltre, la possibilità di utilizzare questi ambienti al fine di costruire un oggetto che svolge una funzione o che verrà utilizzato per partecipare ad una competizione, crea le condizioni per rendere l’acquisizione di tutte le conoscenze utili allo scopo altamente motivanti dal punto di vista degli studenti. Per es., costruire un macchina in grado di partecipare ad una gara, permetterà di rendere il concetto di frizione altamente rilevante dal punto di vista degli interessi dei discenti.

A Papert si deve anche l’elaborazione dell’idea dell’insegnante come facilitatore del processo di apprendimento che, al posto di o in aggiunta a illustrare concetti e idee, propone agli studenti le ricette su come costruire oggetti e che apprende egli stesso insieme agli studenti partecipando al processo di costruzione. Infine, sebbene il contributo fondamentale di Papert è stato quello di dimostrare la possibile utilità dell’uso delle nuove tecnologie nell’educazione, egli fu anche il primo a realizzare che l’introduzione di questi nuovi strumenti in classe non è efficace senza una loro integrazione con l’ambiente educativo complessivo.

1.2 Project-Based Learning

Un secondo aspetto che rende l’utilizzo di ambienti di apprendimento robotici efficaci dal punto di vista educativo è costituito dal fatto che essi si prestano per realizzare processi di apprendimento cosiddetti “project-based”, vale a dire attività di apprendimento che danno l’opportunità agli studenti di lavorare in modo relativamente autonomo e cooperativo durante un periodo di tempo prolungato che termina con la realizzazione di un prodotto (Bransford & Stein, 1993). Queste forme di apprendimento, che sono orientate alla creazione di un prodotto (nel caso della robotica educativa un robot in grado di svolgere una certa funzione) spingono gli studenti a scegliere ed organizzare le proprie attività, condurre attività di ricerca, sintetizzare informazioni, e risolvere problemi.

L'approccio project-based è in linea con l'approccio costruzionista e con le moderne teorie dell'apprendimento che assumono che conoscere, pensare, fare e i contesti nei quali tali attività vengono svolte sono strettamente interrelati. L'apprendimento viene visto come una attività intrinsecamente sociale che avviene in un contesto culturale e sociale specifico.

Gli elementi più importanti dell'apprendimento project-based (Han & Bhattacharya, 2001) sono i seguenti:

Ambiente di apprendimento centrato sul discente: l'ambiente di apprendimento deve essere realizzato in modo tale da stimolare attività di decisione e di iniziativa da parte degli studenti durante il corso del progetto. Inoltre, deve essere strutturato in modo tale da fornire agli studenti delle informazioni e feedback che possano aiutarli a prendere decisioni opportune e a correggere gli errori fatti. Infine gli studenti debbono essere spinti a documentare le cose fatte e le decisioni prese e a valutare e riflettere sui risultati ottenuti.

Collaborazione: l'attività intrapresa deve favorire lo sviluppo di capacità di comunicazione e cooperazione, di decisione di gruppo.

Concretezza: l'attività intrapresa deve essere collegata il più possibile a problemi reali che hanno un valore per gli studenti o per la comunità e realizzata in un contesto realistico.

Rappresentazioni multiple: gli studenti devono essere supportati da strumenti e tecnologie che permettono di accedere a rappresentazioni diverse del processo in atto.

Management: gli studenti devono essere spinti a controllare e gestire efficacemente le risorse disponibili.

Monitoraggio e Valutazione: Gli studenti devono essere incoraggiati a documentare, monitorare e valutare le attività in corso.

1.3 Affrontare le sfide e i cambiamenti sociali

Un terzo fattore che ha favorito l'interesse verso l'utilizzo della robotica in ambito educativo è costituito dal ruolo che questa può svolgere rispetto ai cambiamenti sociali e alle sfide che questi cambiamenti impongono al mondo dell'educazione. L'educazione ha sempre dovuto affrontare il problema di formare gli studenti di oggi alla società di domani, senza sapere con esattezza come la società evolverà nel tempo. Tuttavia l'accelerazione esponenziale dei cambiamenti sociali in corso causata dall'industrializzazione, dalle innovazioni tecnologiche, e dalla globalizzazione, rendono oggi questo problema

particolarmente difficile da affrontare (Brophy et al, 2008). La disponibilità di nuove tecnologie abilitanti costringe le imprese ad essere molto flessibili e innovative per poter essere competitive. Oltre a questo la società moderna impone di affrontare problemi sempre più complessi. Tutto questo richiede la necessità di formare un numero crescente di cittadini in grado di padroneggiare tecnologie in continuo cambiamento, di sviluppare soluzioni innovative a problemi complessi, di disporre di capacità gestionali, organizzative e comunicative.

La capacità di padroneggiare le nuove tecnologie impone al mondo dell'educazione di formare un numero maggiore di cittadini con competenze analitiche nelle aree cosiddette STEM (scienze, tecnologia, ingegneria e matematica). Paradossalmente, tuttavia, l'interesse degli studenti verso quest'area si è progressivamente ridotto negli ultimi anni. Negli Stati Uniti per es. il numero di studenti che si sono laureati in Ingegneria è sceso da 85.000 nel 1985 a 61.000 negli anni 90 e la situazione non sembra in via di miglioramento. Lo sviluppo di azioni che possano favorire una inversione di tendenza rispetto a questo andamento costituisce dunque una esigenza fondamentale universalmente riconosciuta nelle nazioni sviluppate (Brophy et al, 2008).

Analogamente, la necessità di sviluppare nuovi metodi e strumenti che possano favorire l'acquisizione, oltre alle competenze analitiche disciplinari, di una serie di meta-capacità quali la capacità di risolvere problemi, la capacita di integrare conoscenze di tipo diverso superando le barriere disciplinari, la capacità di risolvere problemi, la capacità di gestire un progetto, la capacità di lavorare in gruppo, la capacità di continuare ad apprendere durante tutta la vita, è oggi universalmente riconosciuto.

La robotica educativa può dare un contributo significativo in questa direzione. L'insegnamento della robotica infatti da la possibilità di integrare l'insegnamento di matematica, scienze, ingegneria e tecnologia in contesti concreti che gli studenti trovano motivanti, divertenti, e facili da seguire. Sviluppare un sistema robotico in grado di svolgere una funzione data implica la necessità di integrare conoscenze di tipo diverso, risolvere problemi, allocare appropriatamente le risorse, comunicare e lavorare in gruppo. Tutte competenze fondamentali per la società moderna che vengono spesso vengono trascurate nell'ambito dell'educazione tradizionale.

Particolarmente rilevante, dal punto di vista del tema trattato in questa tesi, è l'aspetto dell'integrazione di conoscenze di tipo diverso. Una assunzione implicita che viene spesso

fatta nel tentativo di affrontare problemi complessi è costituita dall'ipotesi che ogni componente del sistema oggetto di studio possa essere trattato in modo relativamente isolato. Quando questa assunzione è valida, il comportamento del sistema complessivo è il risultato della somma del comportamento dei suoi componenti. Sfortunatamente, questa assunzione non è valida nella maggior parte dei problemi reali (Beer, Chiel & Drushel, 1999). Il comportamento complessivo del sistema dipende in modo significativo dalle interazioni tra i vari componenti del sistema e non è dunque riconducibile al comportamento dei singoli elementi. Questo tipo di organizzazione complessa è caratteristica dei sistemi robotici nei quali il comportamento complessivo emerge dall'interazione tra le diverse componenti (sistema di controllo, corpo, e ambiente) e non può essere ricondotto a nessun componente isolato dagli altri. La robotica educativa, dunque, come cercheremo di dimostrare, può consentire di acquisire una capacità di riconoscere ed affrontare problemi complessi di questo tipo.

1.4 Problemi aperti

Negli ultimi anni l'interesse per l'utilizzo della robotica in ambito educativo è cresciuto significativamente. Ambienti di apprendimento robotici sono utilizzati con successo in tutto il mondo, a partire dalle scuola materna fino all'università. Contemporaneamente, i progressi tecnologici e l'aumento delle imprese che si rivolgono a questo settore hanno consentito di migliorare la qualità e l'offerta di piattaforme robotiche adatte al mondo dell'educazione e a ridurre i costi. Al di la delle piattaforme hardware, anche lo sviluppo di ambienti di apprendimento integrati che includono risorse software e materiale curricolare ha registrato dei progressi. Tuttavia, sebbene la robotica sembra avere potenzialità enormi in campo educativo, l'utilizzo della robotica in ambito educativo come area di ricerca e applicazione è chiaramente ancora in una fase iniziale (Mataric, 2004).

Uno dei problemi aperti riguarda lo sviluppo di materiale curricolare e materiale per la formazione degli insegnati. Lo sviluppo di materiale curricolare di qualità in questo dominio è un processo lento e costoso, per diverse ragioni: la necessità di partire praticamente da zero, la necessità di rivedere e aggiornare il materiale in base all'esito della sperimentazione con gli studenti, la necessità di aggiornare/riorganizzare il materiale rispetto ad ambienti software hardware in continua evoluzione. Come vedremo nel capitolo

successivo, del materiale curricolare di buona qualità è oggi disponibile. Ma tale materiale copre un'area molto limitata rispetto agli ambiti educativa in cui la robotica potrebbe essere applicata.

Un secondo problema aperto riguarda lo sviluppo di metodi efficaci per la formazione degli insegnanti. Anche in questo caso esistono delle iniziative sistematiche relativamente collaudate, come per es. The Student Teacher Outreach Mentorship Program at Tuft University (Postsmore et al. 2003). Tuttavia molto resta ancora da fare.

Infine un terzo problema aperto riguarda lo sviluppo di ambienti di apprendimento in grado di ridurre quanto più possibile il costo di accesso (cioè la quantità di informazioni da acquisire per poter usare l'ambiente) senza compromettere la capacità dell'ambiente di supportare esperienze di apprendimento progressivamente più complesse.

2. Ambienti di apprendimento

In questo capitolo descriveremo gli ambienti di apprendimento più utilizzati o più promettenti dal punto di vista educativo. Alcuni di questi sono basati su piattaforme hardware/software sviluppate specificativamente per questo scopo. Altri invece sono basate su piattaforme sviluppate originariamente per scopi di ricerca oppure per applicazioni commerciali.

2.1 Bee-Bot

Bee-Bot (Figura 2.1) è un piccolo robot programmabile commercializzato dalla TTS Group (www.tts-group.co.uk). Si tratta di un robot molto semplice, sprovvisto di sensori, in grado di eseguire cinque semplici azioni: andare avanti o indietro di 15cm (poco più della propria dimensione), girare a destra o a sinistra di 90 gradi, e rimanere fermo per 1s. Il robot è in grado di memorizzare una sequenza di azioni, ciascuna costituita da una delle cinque azioni elementari (avanti, indietro, sinistra, destra, pausa) e di riprodurre la sequenza di azioni programmata. Per programmare il robot l'utilizzatore deve semplicemente premere i quattro tasti corrispondenti (freccia in avanti, indietro, sinistra e destra) in sequenza. Il robot è dotato infine di altri 2 tasti: il tasto "GO" che attiva l'esecuzione della sequenza di comandi memorizzata precedentemente, e il tasto "CLEAR" che cancella la sequenza di azioni programmata.

Il robot emette un suono ogni volta che una azione viene eseguita e gli occhi lampeggiano quando la sequenza di azioni viene completata. Durante la programmazione della sequenza il robot emette un suono e lampeggia una sola volta con gli occhi in risposta alla pressione di uno dei tasti.

Figura 2.1. Il robot programmabile Bee-Bot visto dall'alto e dal lato frontale (figura di sinistra e destra rispettivamente).

La sua semplicità lo rende facilmente utilizzabile già dai bambini della scuola primaria e dei primi anni delle elementari che possono essere incoraggiati ad intraprendere giochi che richiedono di implicitamente di sviluppare delle capacità di problem solving, come per es., programmare BeeBot in modo che possa raggiungere un obiettivo superando un ostacolo, buttare giù una serie di legnetti posizionati in modo verticale, o raggiugere dei punti diversi di un area divisa in celle.

BeeBot include anche un ambiente software che consente di programmare e osservare il comportamento del robot in simulazione (Figura 2.2). Tale software include delle funzionalità aggiuntive utili. In particolare la possibilità di osservare la sequenza di azioni programmate, la possibilità di editare la sequenza di comandi senza doverla rigenerare da capo, la possibilità di osservare l'azione delle sequenza correntemente in corso di esecuzione, la possibilità di osservare il mondo esterno dal punto di vista del robot.

Figure 2.2 Il software Bee-Bot. In basso a sinistra è possibile vedere la lista dei comandi correnti che può essere editata dall'utente.

BeeBot può essere anche integrato efficacemente con la didattica tradizionale. Come riportato in Battegazzone (2009), per es., può essere utilizzato per compiere le prime astrazioni di eventi ordinati, verificare la correttezza del proprio pensiero, facilitare la narrazione di storie, rappresentare lo spazio esplorato, esercitarsi con le prime operazioni matematiche, creare ritmi, etc.

2.2 Lego Mindstorm

Lego Mindstorm è l'ambiente più utilizzato in ambito educativo. Si tratta di una linea di prodotti LEGO costituita da un mattoncino programmabile (un piccolo computer), motori elettrici, sensori, e una serie di pezzi LEGO Technic (ingranaggi, assi, parti pneumatiche, parti fisse, ect.) che consentono di costruire una grande varietà di robot autonomi e di altri sistemi automatici interattivi.

Hardware

LEGO Mindstorms (Figura 2.3) è stato distribuito commercialmente a partire dal 1998

come "RIS" (*Robotic Invention System* - "Sistema di Invenzione Robotico") o come sistema educativo (*LEGO Mindstorms for Schools*).

Figura 2.3. Mattone programmabile Lego Mindstorms RCX, motori, e sensori (di contatto e di luce ambientale).

Nel 2006 la LEGO ha immesso nel mercato una nuova generazione del prodotto chiamata NXT basata su un nuovo tipo di mattoncino programmabile, l'NXT appunto (Figura 2.4). il Mindstorms NXT, comprende tre servomotori (molto più grandi di quelli della edizione precedente), un sensore tattile, un sensore luminoso in grado di rilevare colori e l'intensità della luce, un sensore sonoro, un sensore di prossimità a ultrasuoni, 519 pezzi LEGO Technic.

Figura 2.4. Mattone programmabile Lego Mindstorms NCX (al centro), servomotori (in alto), e sensori di contatto, di suoni, di luce, di prossimità (in basso, da sinistra a destra, rispettivamente).

Il mattoncino NXT ha un processore a 32 bit Atmel AT91SAM7S256 (classe ARM7) a 48 MHz, con 256k flash memory e 64k RAM, un coprocessore 8 bit Atmel ATmega48 (classe AVR: è un RISC a 8 bit) a 8 MHz, con 4k flash e 512 byte RAM uno schermo LCD con una risoluzione di 60x100 pixel, una porta USB 2.0 e connettività senza fili Bluetooth. Il Mindstorms NXT possiede quattro porte di ingresso e tre di uscita, ma avendo delle connessioni digitali, è possibile aumentarne il numero con dei moduli esterni. I connettori non sono gli stessi dell'RCX e utilizzano porte simili ad un connettore RJ-11. Integrato nel mattoncino c'è un altoparlante da 8 kHZ. Il mattoncino richiede 6 batterie di tipo AA oppure una batteria al litio.

La versione 2.0 dei lego Mindstorm NXT è stata commercializzara a partire dal 2009. La confezione un nuovo tipo di sensore di luce sensibile al colore, due touch sensors, e il sensore ad ultrasuoni. Il nuovo computer di bordo supporta anche le operazioni in floating point.

Il Lego Mindstorms NXT viene commercializzato in due versioni: *Retail* e *Education*. La versione *Retail* è fornita col software di programmazione NXT-G (che illustreremo piu' avanti). La versione educativa che è venduta con batteria al litio e

caricabatterie, invece, non contiene software. Quest'ultimo è venduto separatamente, con tre licenze distinte (*Personal*, *Classroom*, *Site*).

L'utilizzo di elementi assemblabili, consente di utilizzare lo stesso materiale per costruire robot con morfologie diverse come esemplificato nella Figura 2.5

Figura 2.5: Due esempi di robot costruiti con componenti Lego NXT: Tribot (sinistra) e Alpha Rex (destra).

I lego Mindstorm possono essere programmati con diversi ambienti software sviluppati specificatamente per scopi educativi: NXT-G, RobotC, LabView.

Il computer di bordo (NXT), i sensori e i motori lego possono essere usati anche in combinazione con i componenti strutturali di alluminio TETRIX (http://www.tetrixrobotics.com/Building System/) commercializzati dalla Pitsco, una società che sviluppa e commercializza materiale didattico innovativo. Questi componenti (Figura 2.6) risultano più solidi e versatili degli elementi strutturali LEGO TECHNIC, e per questo rappresentano una valida alternativa, soprattutto per sviluppare robot complessi. Per un esempio di robot sviluppato con componenti LEGO NXT e TETRIX si veda la Figura 2.11.

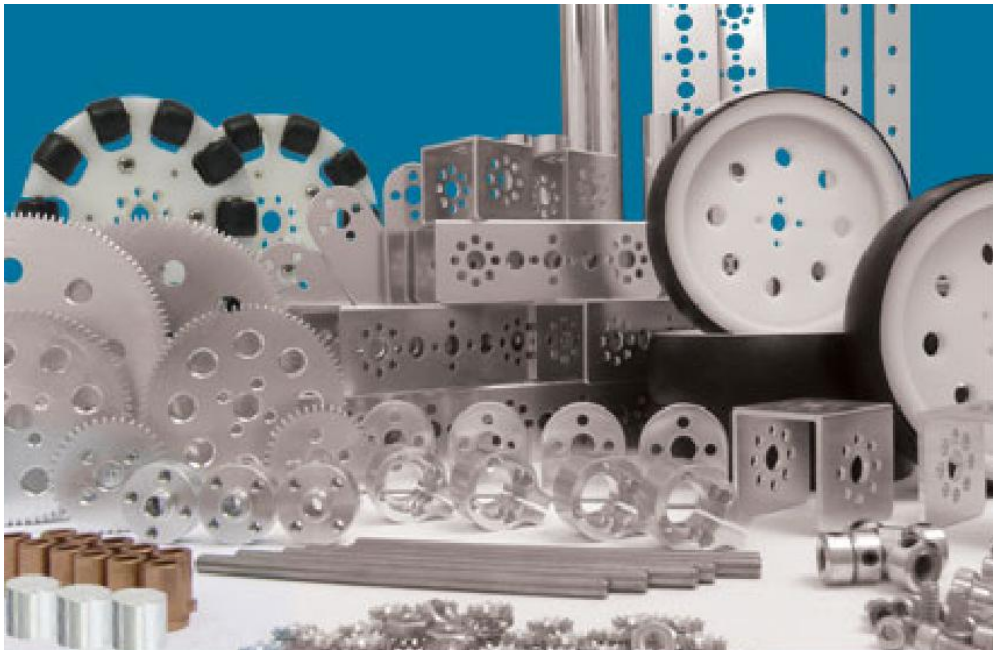


Figura 2.6. Componenti strutturali TETRIX.

NXT-G

NXT-G è il linguaggio di programmazione visuale sviluppato dalla National Instruments (uno dei partner ufficiali della Lego) per i Lego Mindstorm. Si tratta di un adattamento del linguaggio LabView che illustreremo più avanti.

I programmi NXT-G sono costituiti da una o più icone, dette blocchi, disposti attraverso il mouse su un foglio e connessi tra di loro attraverso dei connettori (Figura 2.4). Il linguaggio dispone di diversi tipi di blocchi: inizio, movimento, display, sonori, matematici, condizionali, di attesa e iterativi. Per creare un nuovo blocco è sufficiente sceglierne il tipo dall'elenco dei blocchi predefiniti. I blocchi possono essere trascinati sul foglio quadrettato, trascinati e inseriti tra due blocchi pre-esistenti, o trascinati all'interno di speciali blocchi che possono contenerli (come per es. i blocchi LOOP). Il funzionamento di ciascun blocco viene determinato da una serie di parametri che vengono visualizzati, dopo che il blocco è stato selezionato con il mouse, nella parte inferiore dello schermo e in parte, all'interno dell'icona grafica del blocco. Una volta visualizzati, i parametri possono essere modificati sempre nella parte inferiore dello schermo.

Concatenando una sequenza di blocchi di movimento in sequenza è possibile

creare un sistema di controllo che permette al robot di generare una sequenza di comportamenti. Per es., un semplice sistema di controllo (Figura 2.7) che consente al robot di andare dritto per 3s a velocità massima, girare di 90 gradi a destra, e poi tornare indietro a velocità media per 2s, infine girare a sinistra di 60 gradi, può essere programmato creando e trascinando sulla destra del blocco di inizio, già presente su un foglio vuoto, 4 blocchi di movimento e configurando i blocchi opportunamente. Ciascun blocco motore può controllare più motori contemporaneamente. I parametri che vanno indicati sono: le porte (A,B,C del mattonicino programmabile NXT) a cui i motori sono connessi, il verso di rotazione (orario o antiorario), la direzione relativa di movimento (nel caso il blocco controlli due motori), la velocità desiderata, e l'entità del movimento che deve essere effettuato prima che il controllo passi al blocco successivo (che può essere specificata in termini di rotazioni desiderate, avanzamento in gradi, o tempo).

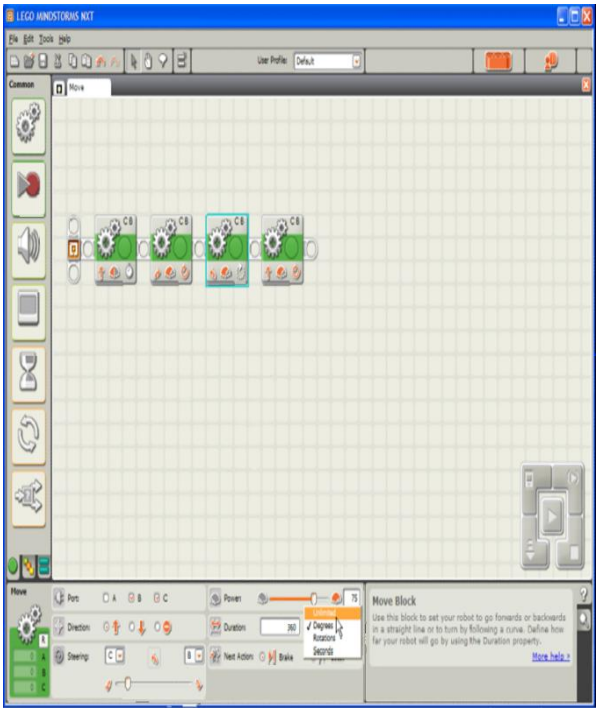


Figura 2.7. Esempio di programma NXT-G. Le icone posizionate all'interno del foglio quadrettato rappresentano 4 blocchi motori collegati in linea che consentono di eseguire una sequenza di azioni semplici. Ciascun blocco può essere configurato in modo che il robot si sposti in avanti o indietro, o si muova lungo una traiettoria circolare in avanti o indietro. Le icone da utilizzare possono essere scelte tra quelle disponibili sul lato sinistro dell'editor e trascinate nel foglio quadrettato. La parte

inferiore dell'editor consente di variare i parametri dell'icona selezionata.

I blocchi di attesa servono a creare dei controlli condizionali che eseguono il comportamento specificato nel blocco successivo solo quando si verificano le condizioni appropriate. Esistono diversi tipi di blocchi condizionali a seconda del tipo di verifica da effettuare. I blocchi condizionali temporali permettono l'esecuzione del blocco successivo dopo che è passato un certo intervallo temporale. I blocchi condizionali sensoriali (di contatto, di luce, o sonar) eseguono il blocco successivo solo quando il sensore corrispondente è attivato o è attivato oltre una certa soglia. Anche in questo caso, i parametri configurabili determinano aspetti importanti come la porta alla quale è connesso il sensore e la modalità di funzionamento del sensore stesso.

Per eseguire le istruzioni contenute in uno o più blocchi più volte si può utilizzare il blocco iterativo LOOP il quale esegue i blocchi contenuti al suo interno un numero infinito di volte, o alternativamente, fino a quando non si verifica una certa condizione specificata. Per inserire uno o più blocchi all'interno del blocco LOOP, basterà trascinare le relative icone all'interno di esso. L'aspetto grafico del blocco si modificherà automaticamente in modo da contenere tutte le icone presenti.

Per eseguire azioni diverse in condizioni diverse si possono utilizzare i blocchi SWITCH. Questi blocchi eseguono le istruzioni contenute in alto oppure in basso a seconda se la condizione specificata è soddisfatta o meno. Il tipo di condizione da verificare viene impostata attraverso i parametri del blocco e può riguardare lo stato di un sensore oppure lo stato di una variabile.

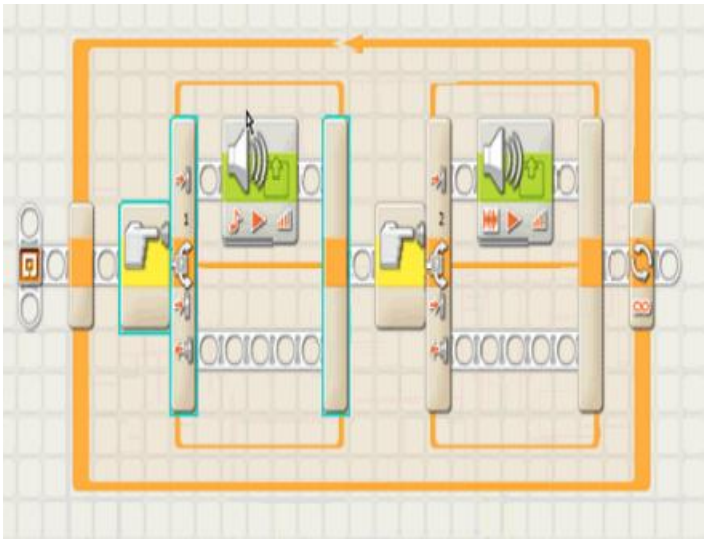


Figura 2.8. Esempio di programma NXT-G che produce una nota quando viene premuto il sensore di tatto 1 e pronuncia una parola quando viene premuto il sensore di tatto 2. Il rettangolo esterno rappresenta il blocco LOOP che esegue le istruzioni contenute all'interno un numero infinito di volte. I due blocchi contenuti all'interno sono dei blocchi SWITCH che eseguono le istruzioni in alto quando la condizione è verificata e le istruzioni in basso quando la condizione non è verificata. In questo caso i due blocchi SWITCH verificano se il sensore di tatto 1 e 2 sono premuti o meno. Si noti come le icone visualizzino anche i parametri del comando (il numero della porta a cui è connesso il sensore, e il tipo di output sonoro, il volume).

L'utente ha anche la possibilità di creare dei nuovi tipi di blocchi, detti MY BLOCKS, costituiti da gruppi di blocchi pre-esistenti. La definizione di un nuovo blocco è dunque analoga alla definizione di una nuova funzione in un linguaggio di programmazione testuale.

I blocchi prevedono anche la possibilità di ricevere degli input (binari, numerici, o testuali) che ne influenzano il comportamento e produrre degli output, che possono influenzare il comportamento di altri blocchi. Per utilizzare questa possibilità l'utente deve visualizzare il DATA HUB del blocco (Figura 2.9), cioè la visualizzazione grafica delle porte in ingresso ed uscita disponibili, e connettere le porte in uscita e le porte in entrata di blocchi diversi attraverso dei cavi (purché il tipo del dato in entrata e in uscita sia congruente).

Figura 2.9. Esempio di programma NXT-G che produce visualizza sul display del computer di bordo del robot il numero del ciclo corrente. Il rettangolo esterno rappresenta un blocco LOOP che viene eseguito un numero infinito di volte. Le due icone all'interno rappresentano rispettivamente un convertitore da formato numero a formato testo e una icona di display. Le colonne sotto le icone rappresentano i rispettivi DATA HUB cioè le porte di ingresso e di uscita. La linea che unisce l'icona LOOP alla seconda icona prende in ingresso il numero corrente del ciclo ripetuto dall'icona LOOP e lo invia all'icona che trasforma una variabile numerica in una variabile testuale. La linea che unisce le due icone contenute all'interno del LOOP, rappresenta un cavo che riceve il numero in formato testo dalla prima icona e lo invia all'icona DISPLAY. Tale icona visualizzerà dunque il testo corrispondente al numero del LOOP.

Per maggiori informazioni sul linguaggio si può consultare il tutorial realizzato da Dale Yocum, insegnate di Robotica presso la scuola Catlin Gabel School in Portland, Oregon, USA consultabile all'indirizzo seguente: http://www.ortop.org/NXT_Tutorial/index.html.

Da questa breve rassegna dovrebbe risultare chiaro come il linguaggio NXT-G consenta di implementare in modo piuttosto semplice ed efficace programmi in grado di far eseguire al robot delle sequenze di azioni eventualmente variando la sequenza quando lo stato di un sensore supera o meno una certa soglia. Il linguaggio consente anche potenzialmente di implementare programmi diversi, che per es. effettuano delle operazioni su delle variabili interne o regolano lo stato dei motori sulla base dello stato dei sensori. In questo secondo caso tuttavia, il linguaggio risulta assai meno agevole da utilizzare.

RobotC

RobotC (http://www.robotc.net/) è un linguaggio di programmazione sviluppato dalla Carnegie Mellon Robotic Academy (http://www.education.rec.ri.cmu.edu/) basato sul linguaggio standard C che può essere utilizzato con i Lego NXT e con le piattaforme VEX Robotics (che descriveremo più avanti). Per poter essere utilizzato sui LEGO-NXT, è necessario prima rimpiazzare il firmware (cioè il sistema operativo standard del NCX) con quello di RobotC.

I robot viene poi programmato scrivendo un programma C (si veda la figura 2.10 per un esempio) attraverso un editor testuale standard che viene poi compilato in linguaggio macchina e caricato sul robot dove può essere eseguito.

```
void forward() {
   motor[motorA] = 100;
   motor[motorB] = 100;
}

void spin() {
   motor[motorA] = 100;
   motor[motorB] = -100;
}

task main() {
  SensorType[S4] = sensorSONAR;
  forward();
  while(true) {
     if (SensorValue[S4] < 25) spin();
     else forward();
  }
}
```

Figure 2.10. Esempio di programma in linguaggio RobotC.

L'esempio riportato nella Figura 2.10 consente ad un robot Lego NXT provvisto di sensore di prossimità sonar montato sul lato frontale di muoversi in avanti evitando gli ostacoli. Il programma è costituito da una funzione principale (main()) e da due funzioni (forward() e spin()) che producono rispettivamente un comportamento di locomozione in avanti e un comportamento di girare sul posto. L'istruzione

while(true) {} produce un ciclo infinito (cioè esegue ripetutamente le istruzioni contenuto all'interno della parentesi graffe che seguono l'istruzione. L'istruzione condizionale if (condizione) istruzione1; else istruzione2; contenuta all'interno di tale ciclo consente di eseguire la prima istruzione quando la condizione è vera e la seconda quando la condizione è falsa. Poiché la condizione è SensorValue[S4] < 25), l'istruzione1 è spin(); e l'istruzione2 è forward(); il ciclo esegue la funzione spin() quando il sensore sonar ha una attivazione minore di 25, cioè quando il robot rileva un ostacolo in prossimità sul proprio lato frontale, e la funzione forward() quando il sensore sonar ha una attivazione maggiore o uguale di 25, cioè quando il lato frontale prossimale del robot è privo di ostacoli. Conseguentemente, il robot tenderà a muoversi in avanti seguendo una traiettoria rettilinea, a ruotare sul posto fino a quando il lato frontale è libero da ostacoli, e a riprendere poi il movimento rettilineo fin quando non viene rilevato un nuovo ostacolo.

Trattandosi di un normale linguaggio di programmazione testuale, l'approccio a RobotC risulta naturalmente assai più faticoso rispetto a NXT-G, per chi non ha già familiarità con la programmazione. D'altra parte, come abbiamo accennato nella sezione precedente, l'utilizzo di NXT-G tende a diventare piuttosto macchinoso quando si affrontano problemi relativamente complessi o problemi che richiedono la rielaborazione delle informazioni. In questi casi RobotC, oltre ad essere più potente, può rivelarsi più semplice da utilizzare.

Un altro aspetto da considerare è che RobotC rappresenta un ottimo strumento didattico quando si utilizza l'ambiente robotico al fine di insegnare l'informatica e la programmazione.

LabVIEW

LabVIEW è un linguaggio di programmazione grafico che usa cioè icone invece che linee di testo per creare programmi, sviluppato della National Instruments a partire dal 1963. Attualmente la National Instruments distribuisce anche una versione Education del software (http://www.ni.com/academic/education_edition/). Come abbiamo detto precedentemente NXT-G è di fatto è basato su LabView, ma contiene solo un subset dei comandi disponibili in LabView. Analogamente l'editor e

l'interaccia grafica di NXT-G risultano semplificate rispetto a quelle di LabView.

La definizione di strutture dati e algoritmi avviene con icone e altri oggetti grafici uniti da linee di collegamento in modo da formare una sorta di diagramma di flusso. Tale linguaggio viene definito dataflow (flusso di dati) in quanto la sequenza di esecuzione è definita e rappresentata dal flusso dei dati stessi attraverso i fili monodirezionali che collegano i blocchi funzionali. La semplicità di programmazione basata sul concetto abbastanza intuitivo di diagramma di flusso e la semplicità di utilizzo hanno reso LabVIEW molto impiegato nell'ambito dell'acquisizione dei dati, nel controllo dei processi industriali, e nella ricerca scientifica.

La versione educativa (che può essere usata con i robot Lego, i robot TETRIX, e con diversi tipi di sensori sviluppati da altre società) è rivolta soprattutto agli studenti delle scuole medie superiori e agli studenti universitari per applicazioni di tipo robotico, per il controllo, e per l'acquisizione di dati.

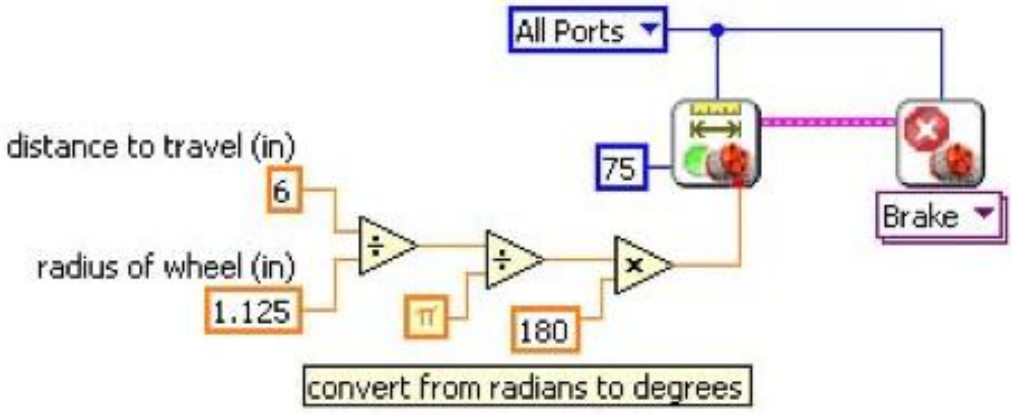


Figura 2.11. Esempio di programma LabView per Lego NXT che fa muovere il robot in avanti di una certa distanza. Il programma riceve in ingresso due valori, che vengono inseriti dall'utente nei due box di input sul lato sinistro, e che codificano la distanza da percorrere e il raggio delle ruote. Tali valori vengono poi elaborati al fine di calcolare il numero di rotazioni in gradi che le ruote devono effettuare e utilizzati come input per il comando/icona motore .

Materiale Curricolare

Esiste un discreta quantità di materiale curricolare per i Lego Mindstorm basato sui diversi dipi di ambienti software descritti sopra.

Il materiale più testato e di migliore qualità, purtroppo non disponibile in Italiano,

è sicuramente quello sviluppato e distribuito dai partners ufficiali LEGO: The Robotics Academy della Università Carnegie Mellon, USA e il Center for Engineering Education and Outreach at Tufts University, USA.

Robotics Academy della Università Carnegie Mellon, USA, (http://www.education.rec.ri.cmu.edu/) sviluppa e distribuisce materiale curricolare a pagamento per studenti delle scuole medie inferiori e superiori. Si tratta di una serie di corsi multimediali molto curati, distribuiti attraverso dei Compact Disk, che coprono diverse aree connesse alla robotica, l'informatica, la matematica, l'introduzione alla tecnologia.

Ciascun corso è articolato attraverso una serie di obiettivi che prevedono lo sviluppo e la programmazione di un robot LEGO NXT, o nel caso dei corsi più avanzati, LEGO NXT e TETRIX o VEX., in grado di svolgere un compito. Nel tentativo di raggiungere l'obiettivo stabilito tuttavia, gli studenti si trovano affrontare una serie di temi connessi alla matematica (come per es. diametro, circonferenza, angolo, unità e conversioni, media, logica booleana) alla tecnologia (come per es. funzione di un artefatto tecnologico, prestazioni, elementi meccanici, controllo), alla scienza (ipotesi e evidenze empiriche, predizione e osservazione, misura, analisi degli errori, velocità/distanza/forza, onde luminose, ultrasuoni) e ad altre meta-capacità (comunicazione, brainstorming, ragionamento, documentazione). Alcuni corsi sono focalizzati direttamente sugli strumenti utilizzati, come per es. l'ambiente NXT-G, o l'ambiente RobotC per LEGO NXT, per LEGO NXT e TETRIX, o per VEX (le piattaforme hardware TETRIX e VEX verranno descritte più avanti) e sono rivolti soprattutto agli educatori.

Center for Engineering Education and Outreach, **Tuft University, USA** (http://www.ceeo.tufts.edu/) ha sviluppato del materiale curricolare rivolto agli studenti delle scuole media superiori. Tale materiale organizzato in forma di un documento scaricabile gratuitamente da internet all'indirizzo http://zone.ni.com/devzone/cda/tut/p/id/10243 (previa registrazione) contiene la descrizione di 5 attività di complessità crescente in cui gli studenti sono invitati a

costruire e programmare dei robot in grado di portare cibo e acqua a delle persone disperse in montagna non raggiungibili a causa di una tempesta in corso. Tutte le attività proposte prevedono l'uso di LabView Educational. Alcune delle attività sono basate su robot LEGO NXT. Altre prevedono anche l'uso di componenti strutturali TETRIX (Pitsco Hardware).

L'obiettivo della prima unità didattica (durata complessiva prevista 90m) è quello di costruire un veicolo NXT (non più grande di certe dimensioni specificate) in grado di muoversi per una distanza predefinita portando del materiale di soccorso (contenuto all'interno di una sfera di plastica) e programmare il sistema di controllo di tale robot utilizzando LabView. Il materiale curricolare fornisce naturalmente indicazioni per l'insegnante su come guidare il processo di sperimentazione/apprendimento dei ragazzi in modo per es. da consentire l'acquisizione di nozioni relative alla meccanica, alla matematica (per es. la relazione tra tempo, velocità e spazio percorso). Inoltre, il materiale curricolare fornisce indicazioni sul tipo di materiale da far utilizzare, e sui diversi modi in cui l'obiettivo può essere raggiunto.

L'obiettivo della seconda unità didattica (durata complessiva prevista 90m) è quello di estendere il robot aggiungendo due sensori di luce rivolti verso il pavimento (che permettono al robot di rilevare il percorso da seguire indicato con una linea tracciata sul pavimento) e estendere il sistema di controllo del robot in modo tale da permettergli di seguire il percorso.

L'obiettivo della terza unità didattica (durata complessiva prevista 180m) è quello di costruire e programmare un robot in grado di arrampicarsi su una superficie inclinata di almeno 20° gradi utilizzando sia componenti Lego, che componenti TETRIX. Un esempio di robot realizzabile in questa unità è visibile nella Figura 2.12.

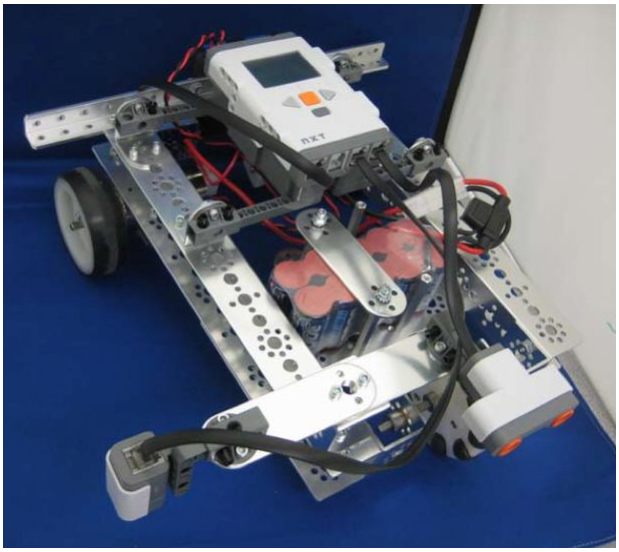


Figura 2.12. Esempio di robot realizzato con componenti LEGO NXT e TETRIX in grado di muoversi su una superficie inclinata seguendo il bordo della strada.

L'obiettivo della quarta unità didattica (durata complessiva prevista 180m) è quello di costruire e programmare un braccio meccanico in grado di depositare il materiale di soccorso in una zona non raggiungibile dal robot orientando un braccio robotico con un grado di libertà a 45 gradi. Nel raggiungere tale obiettivo gli studenti sperimenteranno concetti quali i gradi di libertà, e la forza di torsione.

Infine l'obiettivo della quinta e ultima unità didattica (durata complessiva prevista 180m) è quello di costruire e programmare un robot in grado di raggiungere la "cima della montagna" e poi distribuire il carico attraverso il braccio meccanico ad altezze diverse in modo da raggiungere le diverse posizioni dove sono localizzati i "dispersi". Per far questo gli studenti dovranno utilizzare il teorema di Pitagora e la trigonometria.

2.3 VEX Robotics Design System

VEX Robotics Design System (http://www.vexrobotics.com/) è un kit commercializzato a partire dal 2006 da Innovation First International Inc. (un piccola società Statunitense) per l'educazione e per gli amatori. Il kit (Figura 2.13) è costituito anche in questo caso da un piccolo computer di bordo "VEX Brain", da una serie di sensori, motori, e parti, e un telecomando. Il computer di bordo "Cortex" è dotato di

una scheda di comunicazione wireless, 8 porte per i servo-motori, 2 porte per motori standard, 8 porte analogiche per i sensori, una porta I2C (che può essere utilizzata per connettere diversi sensori in serie). Tale computer può essere programmato con RobotC and EasyC, un'altro ambiente di programmazione testuale basato sul linguaggio C. Il telecomando può essere utilizzato per comandare manualmente il robot o per realizzare un controllo misto in cui il robot si muove autonomamente in base al proprio sistema di controllo e al tempo stesso reagisce ai comandi manuali dell'utilizzatore inviati attraverso il telecomando. I motori includono sia motori elettrici standard che servomotori. I sensori includono sensori di contatti, sensori ad ultrasuoni, sensori di luce etc.

Dal punto di vista del materiale curricolare, esiste un corso introduttivo basato su RobotC sviluppato e commercializzato dalla Robotics Academy della Carnegie Mellon University.

Figura 2.13 Il Kit robotico VEX.

2.4 iRobot Create & Robotic Primer WorkBook

IRobot Create (Figura 2.14, sinistra) è un robot commercializzato per scopi educativi derivato da iRobot Roomba (Figura 2.14, destra), un robot aspirapolvere autonomo di cui sono stati venduti ad oggi circa 2 milioni di esemplari. Entrambi i robot vengono commercializzati dalla iRobotics, una società fondata dal Prof. Rodney Brooks, uno dei padri della robotica moderna. iRobot Create è praticamente identico al Roomba ma, al posto dell'aspirapolvere include una zona di carico vuota che viene utilizzata per contenere il computer di bordo e delle porte che possono esser utilizzare per collegare

sensori e motori addizionali. Il computer di bordo fornito con iCreate (iRobot Command Module) ha capacità piuttosto limitate. In alternativa tuttavia è possibile utilizzare dei sistemi più potenti, come per es. i computer miniaturizzati della linea Gumstix (http://www.gumstix.com/). Inoltre il robot è fornito di una serie di accessori. In particolare un componente in grado di creare un muro virtuale percepibile dal robot attraverso il sensore ad infrarosso, una stazione di ricarica, e un telecomando. Il robot inoltre è in grado di alloggiare a bordo sensori e attuatori addizionali come per es. un braccio meccanico (Figura 2.15).

Il modulo base del robot contiene un sensore omnidirezionale ad infrarossi, 4 sensori di prossimità, 2 sensori di contatto utilizzati di norma per allinearsi con la stazione di ricarica, 2 due ruote cingolate retrattili. Il robot viene fornito con una serie di esempi pre-programmati in grado di produrre comportamenti, quali: (1) seguire le pareti di una stanza utilizzando i sensori di prossimità e di contatto, (2) muoversi seguendo delle spirali in avanti e indietro in modo da esplorare l'area intorno al punto di inizio, (3) raggiungere la stazione di ricarica e collegarsi ad essa utilizzando il sensore ad infrarosso omnidirezionale, (4) esplorare una stanza alternando comportamenti di wall-following, di evitamento delle pareti attraverso dei comportamenti di "rimbalzo" e movimenti a spirale.

Figura 2.14. Sinistra: iRobot Create con il "command module". Destra: Irobot Roomba, il robot aspirapolvere autonomo.

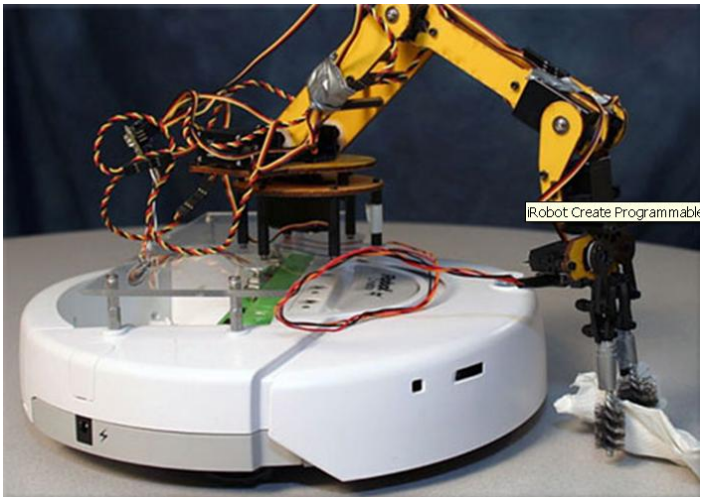


Figura 2.15. iRobot Create con un braccio robotico.

Recentemente la Prof.ssa. Maja Mataric, direttore del "Center for Robotics and Embedded Systems" della University of Southern California, USA, ha sviluppato del materiale curricolare sulla robotica autonoma pensato per studenti di livello universitario e per amatori basato sulla piattaforma ICreate. Tale materiale consiste in particolare del volume "The Robotics Primer Book" pubblicato da MIT Press (Maataric, 2007) che copre la parte teorica e da un WorkBook consultabile gratuitamente su internet (http://roboticsprimer.sourceforge.net/workbook/Main_Page) organizzato in una serie di unità di apprendimento che trattano i stessi temi descritti nel primo volume dal punto di vista pratico sperimentale. Dal punto di vista software, il workbook si basa su strumenti pre-esistenti sviluppati in ambito di ricerca o per la ricerca, come URBI (una libreria software gratuita basata sul linguaggio di programmazione C sviluppata per il controllo di robot) e il Microsoft Robotics Developer Studio, un ambiente per lo sviluppo e la simulazione di robot sviluppato per la ricerca e per la robotica industriale.

L'obiettivo dell'autrice, per la verità, era quello di rendere la robotica insegnata a livello universitario accessibile anche ai livelli di istruzione precedenti e ai non specialisti, come per es. gli insegnanti delle scuole superiori. Tale obiettivo tuttavia è stato raggiunto solo in parte. Il volume teorico, che illustra i concetti basilari della robotica moderna e i diversi approcci utilizzati per sviluppare robot, è effettivamente

di facile lettura. In particolare, il volume introduce al lettore in modo semplice e chiaro i componenti fondamentali dei robot (sistema di controllo, sensori, e motori) e le architetture (reattiva, behavior-based, e deliberativa) che vengono utilizzate nella per progettare il sistema di controllo dei robot. Al contrario, il workbook, che si pone come obiettivo quello di tradurre in attività pratiche i concetti illustrati a livello teorico nel primo volume, risulta lacunoso, complesso, e di difficile consultazione. Tale complessità si deve in primo luogo alla necessità di padroneggiare strumenti hardware e software eterogenei, spesso poco documentati, e in secondo luogo alla lacunosità delle spiegazioni e degli esercizi proposti.

3. EduBot: Un ambiente di apprendimento basato su robot Lego e reti neurali artificiali

3.1 Introduzione

Una interessante applicazione della robotica educativa consiste nell'utilizzare il robot come modello di un organismo naturale, sia pure molto semplice, dotato di un corpo, un sistema sensoriale, un sistema motorio, e un sistema nervoso. Utilizzare questo tipo di prospettiva, significa costruire degli artefatti robotici bio-ispirati che condividono cioè alcune caratteristiche fondamentali con gli organismi naturali. Tale scelta ha un vantaggio duplice: da un lato consente di conciliare l'attività didattica con la visione moderna della robotica, dall'altro può consentire di utilizzare la robotica educativa per veicolare conoscenze in ambiti disciplinari diversi da quelli trattati abitualmente.

In questo capitolo descriviamo i presupposti teorici di questa scelta e Edubot, l'ambiente di apprendimento sviluppato adattando dei tool software e hardware sviluppati precedentemente per altri scopi. In particolare, nella sezione 3.2 descriviamo brevemente la robotica bio-inspirata e i veicoli di Braitenberg, che hanno rappresentato le fonti di ispirazione più importanti per il sistema sviluppato. Nella sezione 3.3 descriviamo BestBot e Evorobot*, due sistemi che sono alla base del nuovo ambiente. Nella sezione 3.4 descriviamo EduBot, l'ambiente educativo sviluppato nell'ambito di questa tesi. Infine, nella sezione 3.5 discutiamo le differenze tra il sistema sviluppato e gli altri ambienti di apprendimento disponibili.

3.2 La robotica bio-ispirata e i veicoli di Braitenberg

L'idea che costruire dei robot in grado di esibire dei comportamenti in un ambiente possa contribuire a capire l'intelligenza animale o umana risale almeno alla seconda metà del 900, prima dell'avvento del computer, in cui si affermò la cibernetica. E' nell'ambito di queste ricerche che si afferma l'idea di unire lo studio degli organismi naturali e delle macchine sviluppando degli artefatti in grado di esibire delle capacità

comportamentali e cognitive basate sui principi sottostanti al comportamento animale.

> *Le prime ricerche in questo ambito mettono immediatamente in luce la forte interdipendenza tra macchina e ambiente e la natura emergente del comportamento esibito da una macchina (cioè il fatto che macchine semplici dal punto di vista del corpo e del sistema nervoso possono produrre comportamenti che appaiono complessi dal punto di vista di un osservatore esterno).*
>
> Nolfi (2010), pp. 20.

L'avvento del computer e l'affermarsi dell'Intelligenza Artificiale alla fine degli anni 60 producono un temporaneo disinteresse per queste idee. L'intelligenza (naturale e artificiale) viene concepita sostanzialmente come un processo di manipolazione di simboli. Tuttavia, l'interesse verso lo studio di sistemi bio-inspirati, in robotica, si riafferma con forza a partire dagli anni 80, con l'avvento della robotica behavior-based (Brooks, 1986) e dell'approccio cosiddetto "embodied" nelle Scienze Cognitive (Clark, 1997) . In sostanza, tali approcci tendono a riaffermare l'importanza di considerare il comportamento e la cognizione come un fenomeno emergente dell'interazione tra l'agente e l'ambiente piuttosto che il risultato di un processo di manipolazione di rappresentazioni simboliche.

Particolarmente interessante, dal nostro punto di vista, è il piccolo volume "I Veicoli Pensanti: saggio di psicologia sintetica" pubblicato da Valentino Braitenberg nel 1984 del quale riportiamo qui di seguito parte della sezione introduttiva:

> *Per anni mi sono occupato di certe strutture che si trovano nei cervelli degli animali, e che sembrano interpretabili come componenti di un calcolatore elettronico a causa della loro semplicità e regolarità, ma temo che gran parte di questo lavoro possa essere affascinante solo per chi lo fa. Mi è capitato però mentre contavo le fibre del secondo ganglio visivo della mosca, o le sinapsi nella corteccia celebrale del topo, di accorgermi che distinzioni e nodi non risolti, problemi e difficoltà che avevo incontrato nei primi miei ingenui contatti con il problema filosofico della mente, andavano scomparendo. Questo processo di purificazione è stato per anni una deliziosa esperienza, e questo libro è stato scritto per comunicarla almeno in parte se siete pronti a seguirmi,*

non attraverso un mondo di cervelli veri, ma in un mondo di giocattoli che proveremo a creare insieme.

In queste pagine non troverete un riassunto del mio lavoro scientifico (tuttavia rimando il lettore all'appendice, in cui cercherò di dimostrare che alcune delle riflessioni che seguiranno valgono anche per i cervelli reali).

Il nostro discorso sarà limitato a certe macchine con struttura interna molto semplice, così semplice da non essere troppo interessante ne dal punto di vista dell'ingegneria meccanica ne da quello dell'ingegneria elettronica.

L'interesse nascerà invece quando proveremo ad immaginare queste macchine (o veicoli) come animali nel loro ambiente naturale. Nelle pagine che seguono saremo ogni tanto tentati di usare un linguaggio psicologico per descrivere il comportamento di questi veicoli, pur sapendo che in quelle macchine non c'é nulla che non sia stato messo dentro da noi. Sarà un gioco avvincente e istruttivo.

Braitenberg (1984), pp. 17.

L'autore illustra poi una serie di veicoli, semplici robot autonomi dotati di sensori, motori, e di un semplice sistema nervoso, immaginando e descrivendo il comportamento che tali veicoli esibirebbero in alcune condizioni ambientali. A titolo esemplificativo, possiamo considerare il veicolo N.2 (Figura 3.1).

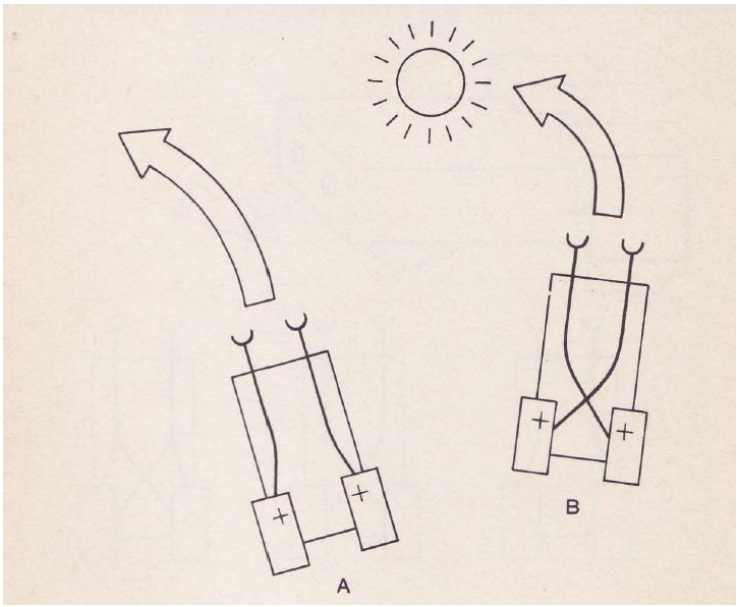


Figura 3.1. Il veicolo N. 2. Braitenberg (1984).

Tale veicolo ha due sensori, uno su ogni lato, connessi a due motori, uno a destra e uno a sinistra. Più i sensori sono eccitati, più forte è la spinta del motore. Nel veicolo 2A ogni sensore è collegato al motore posto sullo stesso lato. Nel veicolo 2B invece ogni sensore è collegato al motore posto sul lato opposto. Immaginiamo di situare i due veicoli in un ambiente dove la quantità di sostanza che eccita i sensori (per es. la luce) varia. Il veicolo 2A passerà la maggior parte del tempo nei posti dove la quantità della luce è poca e accelera invece dove l'intensità della luce è maggiore. Nei pressi di una sorgente luminosa il veicolo 2A vi ci si precipiterà contro, se orientato esattamente nella sua direzione. Se invece la sorgente luminosa è posta su un lato, il comportamento cambia perché il sensore posto sul lato più vicino alla luce sarà attivato in maniera maggiore. Il risultato sarà una maggiore velocità del motore corrispondente che farà allontanare il veicolo dalla sorgente luminosa. Il veicolo 2B, analogamente al 2A, tenderà a muoversi rapidamente nelle zone illuminate e lentamente nelle zone scure. Tuttavia quando la sorgente è posta di lato, esso si dirigerà verso la sorgente di luce fino a raggiungerla e colpirla. Entrambi i veicoli dunque, si comportano come se non amassero la luce, ma esibiscono due comportamenti diversi. Il primo, che potremmo chiamare codardo, quando incontra una sorgente luminosa accelera e gira in modo da evitarla e raggiungere una zona meno illuminata. Il secondo, che potremmo definire aggressivo, accelera e si dirige a velocità massima contro la sorgente luminosa come se volesse distruggerla, o superarla prima possibile.

Dunque i due veicoli esibiscono dei comportamenti apparentemente orientati ad uno scopo (evitare la luce) anche se tale scopo non è rappresentato all'interno del loro semplice sistema nervoso. Anche i comportamenti stessi, accelerare in presenza di luce girando nella direzione opposta della luce o verso la direzione della luce, non sono riconducibili solo alla struttura dei veicoli ma piuttosto all'interazione tra i veicoli e l'ambiente. Per es., la capacità del veicolo 2B di girare verso la luce fino ad averla di fronte procedendo poi in modo rettilineo contro la sorgente di luce, si spiega con il fatto che la percezione di intensità di luce diversa sui due sensori produce una risposta differenziata sui due motori che riduce il disallineamento tra il robot e la

sorgente luminosa. La riduzione della discrepanza tra l'attivazione dei due sensori di luce produce una attivazione dei motori meno differenziata che a sua volta produce una modifica meno forte dell'orientamento del robot. La reiterazione di tale tipo di interazioni conduce, dopo un po' di tempo, ad una situazione in cui il veicolo 2B percepisce la stessa intensità di luce sui due sensori. Questo tipo di stimolazione sensoriale elicita una risposta omogenea dei due motori che non modifica l'orientamento tra il robot e la sorgente luminosa. Conseguentemente il veicolo non modifica più il proprio orientamento e comincia a muoversi in modo rettilineo contro la sorgente luminosa.

Il volume di Braitenberg illustra 14 diversi veicoli, di complessità crescente, che servono all'autore per illustrare una serie di aspetti, come ad es., il ruolo delle connessioni attivatorie e inibitorie tra i neuroni (veicolo N. 3), e la capacità di un semplice processo di selezione e variazione analogo all'evoluzione naturale di generare veicoli in grado di svolgere efficacemente una certa funzione (veicolo N. 6).

L'obiettivo che ci siamo proposti in questa tesi è quello di sviluppare un ambiente di apprendimento che possa consentire a degli studenti di effettuare degli esperimenti analoghi a quelli immaginati da Braitenberg utilizzando le tecnologie software e hardware oggi disponibili piuttosto che il pure esercizio mentale. In particolare, abbiamo cercato di realizzare un ambiente integrato software e hardware che consenta di creare facilmente robot di questo tipo, di osservarne il comportamento nel mondo reale (e in simulazione), e di modificarne le caratteristiche al fine di ottenere uno scopo desiderato. Tale processo consente ai ragazzi di acquisire una conoscenza sistemica delle varie componenti coinvolte, sensori, motori, corpo, sistema nervoso, ambiente. Inoltre dovrebbe consentire ai ragazzi di acquisire la capacità di identificare e capire sistemi con una natura complessa, cioè sistemi il cui comportamento complessivo dipende in modo cruciale dalle interazioni tra le parti di cui sono costituiti.

3.3 Strumenti software & hardware utilizzati

Per sviluppare un ambiente di apprendimento con le caratteristiche descritte nella sezione precedente, siamo partiti da BestBot (un prototipo hardware basato su componenti Lego NXT sviluppato per attività di edutainment) e da Evorobot* (un software sviluppato per scopi di ricerca) dei quali riassumiamo brevemente le caratteristiche nelle due sottosezioni seguenti.

Bestbot

BestBot (http://eutopia.unina.it/bestbot2/) è un serious game robotico che è stato sviluppato dal Laboratorio per lo Studio dei Processi Cognitivi Naturali ed Artificiali (NAC), Dipartimento di Scienze Relazionali "G. Iacono" dell'Università di Napoli "Federico II" e dal Laboratorio di Robotica Autonoma e Vita Artificiale dell'Istituto di Scienze e Tecnologie della Cognizione, del CNR. BestBot è stato istallato e utilizzato con successo da centinaia di utenti presso la Mostra "Futuro Remoto 2009 …. E creò il robot a sua immagine e somiglianza" che si è tenuta alla Citta della Scienza di Napoli da 20 novembre all'8 dicembre 2009 e presso il Festival della Scienza di Genova, dal 29 Ottobre al 7 novembre 2010.

L'obiettivo di questo ambiente è quello di introdurre i visitatori alla robotica autonoma e alla possibilità di sviluppare dei robot in grado di svolgere un compito attraverso un processo di apprendimento. BestBot comprende una piattaforma robotica basata su componenti LEGO NXT e compatibili, e più precisamente: il computer di bordo Lego NXT, due servomotori Lego NXT, tre sensori di prossimità ad infrarosso DIST-Nx-Short-v2 della Mindsensor.com, e una telecamera Vision Subsystem v2 for NXT della Mindsensor.com. Tali componenti sono assemblati su due supporti circolari di vetroresina disegnati appositamente per aumentare la robustezza del robot e consentire il posizionamento e l'orientamento dei sensori.

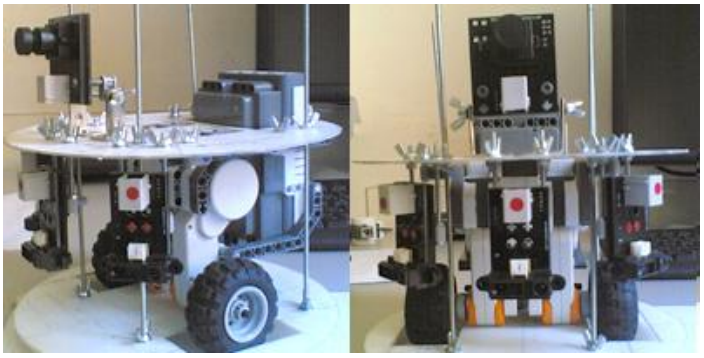


Figura 3.2 La piattaforma hardware BestBot.

BestBot inoltre comprende un ambiente software che consente agli utenti di osservare il comportamento dei robot in simulazione, addestrare i robot attraverso un processo di selezione guidato dall'utente, trasferire il sistema di controllo su dei robot reali posti in un ambiente analogo a quello simulato.

Figura 3.3 L'ambiente software BestBot.

Il compito dei robot consiste nel raggiungere l'oggetto target rosso nel più breve tempo possibile evitando gli ostacoli costituiti dalle mura e dagli altri robot. I comportamento iniziale dei robot è casualmente diverso. L'utente tuttavia ha la possibilità di selezionare e far "riprodurre" i robot più promettenti (creando cioè dei

nuovi robot dotati di un sistema di controllo uguale a quello del robot selezionato al quale tuttavia vengono aggiunte delle variazioni casuali) eliminando contemporaneamente i robot meno promettenti. La ripetizione di tale processo, consente di ottenere robot progressivamente più efficaci. L'utente inoltre ha la possibilità di variare la complessità del compito inserendo nell'ambiente degli ostacoli (i cilindri verdi posti inizialmente sul lato esterno destro dell'ambiente, si veda la Figura 3.3). Una volta ottenuto un robot efficiente, e soprattutto dopo aver osservato come sia possibile evolvere dei robot in grado di svolgere un compito dato attraverso la combinazione di un processo di riproduzione selettiva e variazione casuale, l'utente ha possibilità di far mettere il robot scelto in competizione con altri robot "allevati" da altri utenti.

I robot scelti, infatti, vengono inseriti all'interno di un torneo, e si confrontano i simulazione al fine di identificare i robot più performanti (Figura 3.4), ovvero i robot in grado di raggiungere in media il target rosso nel più breve tempo possibile senza urtare gli altri robot presenti nell'arena che naturalmente cercheranno di ostacolare gli avversari. Nel caso di istallazioni un cui i robot fisici sono disponibili, come nel caso delle istallazioni effettuate presso la Citta della Scienza di Napoli e il Festival della Scienza di Genova, le gare vengono realizzate nell'ambiente reale con i robot fisici.

BestBot presenta diverse caratteristiche interessanti dal nostro punto di vista. Innanzi tutto la piattaforma hardware è robusta e ha un costo piuttosto contenuto. Inoltre, dal punto di vista software, è molto semplice da utilizzare. Nella versione attuale tuttavia, il software non consente di mettere in relazione il sistema nervoso del robot e il comportamento risultante. Inoltre non consente di approfondire la relazione tra le modifiche inserite durante il processo di adattamento, il sistema nervoso, e il comportamento risultante.

Figura 3.4. La classifica dei bot più performanti con i relativi punteggi.

Evorobot*

Evorobot* (Nolfi & Gigliotta, 2001; http://laral.istc.cnr.it/evorobotstar/) è un software sviluppato per scopi di ricerca all'Istituto di Scienze e Tecnologie della Cognizione del CNR per realizzare degli esperimenti di robotica evolutiva (Nolfi, 2009). In pratica questo software consente di sviluppare dei robot in grado di svolgere un compito dato (come per es. raccogliere oggetti in un ambiente depositandoli in una particolare zona oppure trovare e riconoscere un oggetto target) attraverso un processo di evoluzione artificiale.

Evorobot* è costituito da cinque moduli integrati: (1) un editor grafico che consente di definire il tipo di robot che si vuole utilizzare, i sensori e i motori di cui il robot è dotato, l'architettura del sistema nervoso del robot, e (qualora si voglia osservare il comportamento del robot in simulazione, le caratteristiche dell'ambiente nel quale il robot viene posto), (2) una libreria che consente emulare il funzionamento della rete neurale artificiale che costituisce il sistema nervoso del robot (che consente di aggiornare cioè lo stato di attivazione dei neuroni sensoriali, interni, e motori del sistema nervoso del robot), (3) una libreria che consente di evolvere i parametri della rete neurale attraverso un algoritmo evolutivo, (4) un simulatore del robot e

dell'ambiente che consente di calcolare lo stato di attivazione dei sensori del robot in base alla posizione corrente del robot, gli effetti dell'attivazione dei motori del robot sull'ambiente o sulla posizione relativa del robot nell'ambiente, e gli effetti delle interazioni fisiche tra il robot e l'ambiente (per es. gli effetti di una collisione tra il robot e l'ambiente), (5) una interfaccia grafica che consente di analizzare il comportamento esibito da uno o più robot nell'ambiente reale o in simulazione (si veda la Figura 3.5).

Evorobot* supporta due tipi di robot di piccole dimensioni: i robot Khepera e i robot E-puck entrambi dotati di due ruote motorizzate e di 8 sensori ad infrarosso che possono essere utilizzati anche come sensori di luce ambientale. Il robot Khepera è dotato di un sistema di comunicazione via cavo RS232. Il robot E-puck è dotato anche di una telecamera, di 2 microfoni, un altoparlante, e un sistema di comunicazione wireless bluetooth. Sebbene il robot e-Puck sia stato sviluppato anche per fini educativi, entrambi i robot risultano meno robusti dei robot Lego.

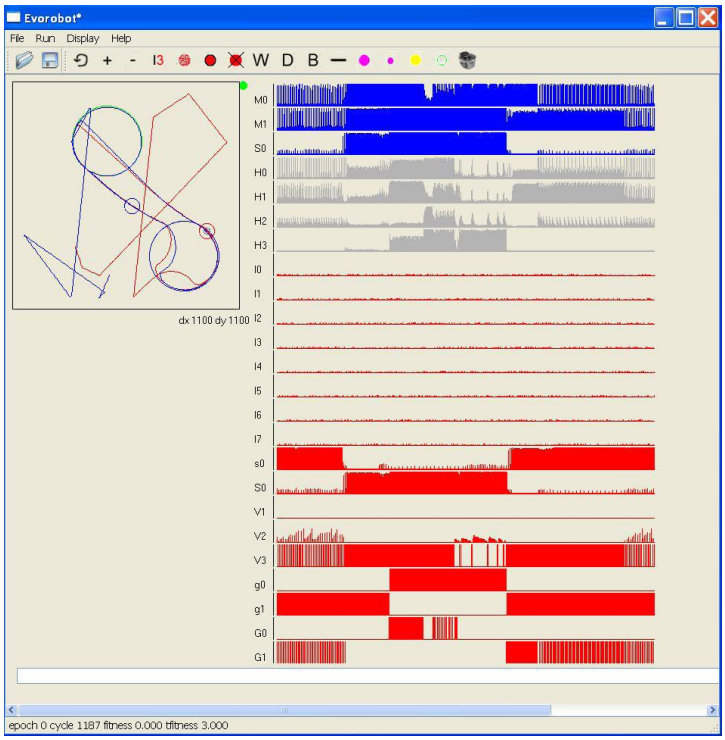


Figura 3.5. L'interfaccia grafica del programma Evorobot*. L'interfaccia del programma prevede la possibilità di eseguire una serie di comandi selezionando i menù a tendina situati in alto e le icone, posizionate immediatamente sotto. L'area in alto a sinistra viene utilizzata per visualizzare l'ambiente e i robot in movimento. La

porzione destra della finestra viene utilizzata per visualizzare lo stato dei sensori, dei motori, e dei neuroni interni del sistema di controllo del robot.

Si tratta dunque di un software molto potente ma non sufficientemente semplice ed intuitivo per essere utilizzato in ambito educativo.

3.4 Edubot

Al fine di sviluppare un ambiente software/hardware che possa consentire agli utilizzatori di approfondire le relazioni funzionali tra sensori, attuatori, sistema nervoso, comportamento, e apprendimento si è deciso di adattare la piattaforma hardware di BestBot e l'ambiente software Evorobot*. In questo capitolo descriviamo le caratteristiche hardware e software di questo nuovo ambiente e le ragioni sottostanti alle scelte di design effettuate. E' importante considerare che tale lavoro è stato portato avanti di pari passo allo sviluppo del supporto didattico/curricolare che, per chiarezza di esposizione, verrà descritto nel capitolo successivo.

Piattaforma hardware

Per quanto riguarda la piattaforma hardware Edubot è basato sostanzialmente sulla piattaforma sviluppata precedentemente per BestBot. Il processo di re-design ha riguardato infatti solo due aspetti: la variazione della configurazione sensoriale del robot e l'utilizzo del display del computer di bordo per visualizzare lo stato corrente dei sensori e dei motori del robot ed eventualmente lo stato dei neuroni interni del sistema di controllo.

Per quanto riguarda il primo aspetto si è deciso di montare sulla configurazione di base del robot due sensori di luminosità Lego NXT posizionati sul lato frontale sinistro e destro del robot, un sensore di luminosità Lego NXT orientato verso il pavimento posto sul lato frontale del robot, e opzionalmente la telecamera Mindstorm.com. La telecamera è opzionale nel senso che viene utilizzata solo per effettuare delle estensioni degli esperimenti base previsti.

Per quanto riguarda il secondo aspetto, è stato variato il firmware da istallare sul computer di bordo del robot al fine di consentire a questo di visualizzare lo stato

corrente dei sensori e dei motori del robot ed eventualmente lo stato dei neuroni interni. Come evidenziato nella Figura 3.6, lo stato corrente viene visualizzato in modo numerico accanto al simbolo del sensore/motore corrispondente. Inoltre, ciascuno stato viene visualizzato nel tempo in modo grafico, così da apprezzare le variazioni dello stato e la relazione tra le variazioni dello stato e la variazione della posizione del robot nell'ambiente.

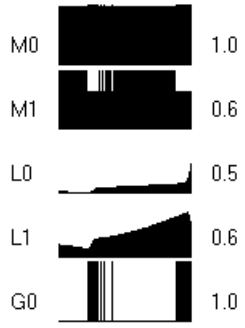


Figura 3.6. Visualizzazione numerica e grafica dello stato dei sensori, dei motori, ed eventualmente dei neuroni interni sul display del computer di bordo del robot. M0 e M1 rappresentano il motore della ruota sinistra e destra. I sensori L0 e L1 i due sensori di luce montati sul lato frontale sinistro e destro. Il sensore G0, il sensore di luce montato sul lato frontale orientato verso il pavimento.

Tale sistema di visualizzazione, come illustreremo meglio nel capitolo successivo, consente di illustrare all'utilizzatore in modo semplice ed efficace il funzionamento dei sensori, dei motori, e del sistema di controllo (implementato sul computer di bordo). L'utente infatti ha la possibilità di variare la posizione del robot nell'ambiente o la struttura dell'ambiente osservando direttamente sul display del robot le variazioni conseguenti dello stato dei sensori. Inoltre ha la possibilità di osservare gli effetti dello stato dei sensori sulla velocità delle ruote tenendo il robot sospeso da terra. Consente inoltre all'utente di cominciare ad acquisire nozioni sulla relazione tra robot e ambiente

Ambiente software

Come detto precedentemente, l'ambiente software di Edubot è stato sviluppato partendo dal software Evorobot* descritto sopra. In questo caso tuttavia, il processo di re-design è stato piuttosto significativo in quanto ha comportato lo sviluppo di

diverse funzionalità nuove e il re-design quasi totale dell'interfaccia grafica. In questa sezione descriviamo le caratteristiche dell'ambiente sviluppato, le funzionalità aggiunte, e le variazioni effettuate per rendere il software utilizzabile in ambito educativo.

Il software prevede la possibilità di preparare e utilizzare una serie di esperienze di apprendimento guidate che consentono di utilizzare l'ambiente in modo efficace riducendo al minimo il costo di ingresso (cioè il numero di operazioni preparatorie e il la quantità di informazioni introduttive da acquisire per poter utilizzare efficacemente l'ambiente). Ciascuna esperienza guidata di apprendimento corrisponde ad una cartella che contiene i relativi file di configurazione. In questa sezione descriveremo in particolare il caso dell'esperienza di apprendimento denominata "ESPLORA" che è stata sviluppata al fine di consentire agli utilizzatori di familiarizzare con l'ambiente e di esplorare le aree tematiche introdotte nelle sezioni precedenti di questo capitolo e descritte in modo dettagliato nel capitolo successivo.

Una volta eseguito, il programma si presenta attraverso 4 finestre indipendenti che corrispondono a 4 moduli del programma stesso: (1) Una finestra **Edubot** che contiene una serie di icone che permettono di eseguire i comandi fondamentali e uno spazio che viene utilizzato per visualizzare lo stesso tipo di informazioni visualizzate sul display del computer di bordo del robot (vale a dire lo stato dei sensori, dei motori, e eventualmente degli stati interni del robot mentre questo interagisce con l'ambiente), (2) una finestra **Sistema Nervoso** che visualizza il sistema di controllo del robot, (3) una finestra **Robot/Ambiente** che visualizza il robot e l'ambiente in simulazione, (4) una finestra **Adattamento** che visualizza come le prestazioni del robot variano, rispetto ad un compito dato, durante il processo di adattamento. Nel caso dell'esperienza guidata di apprendimento ESPLORA, il programma si presenta nel modo illustrato nella Figura 3.7.

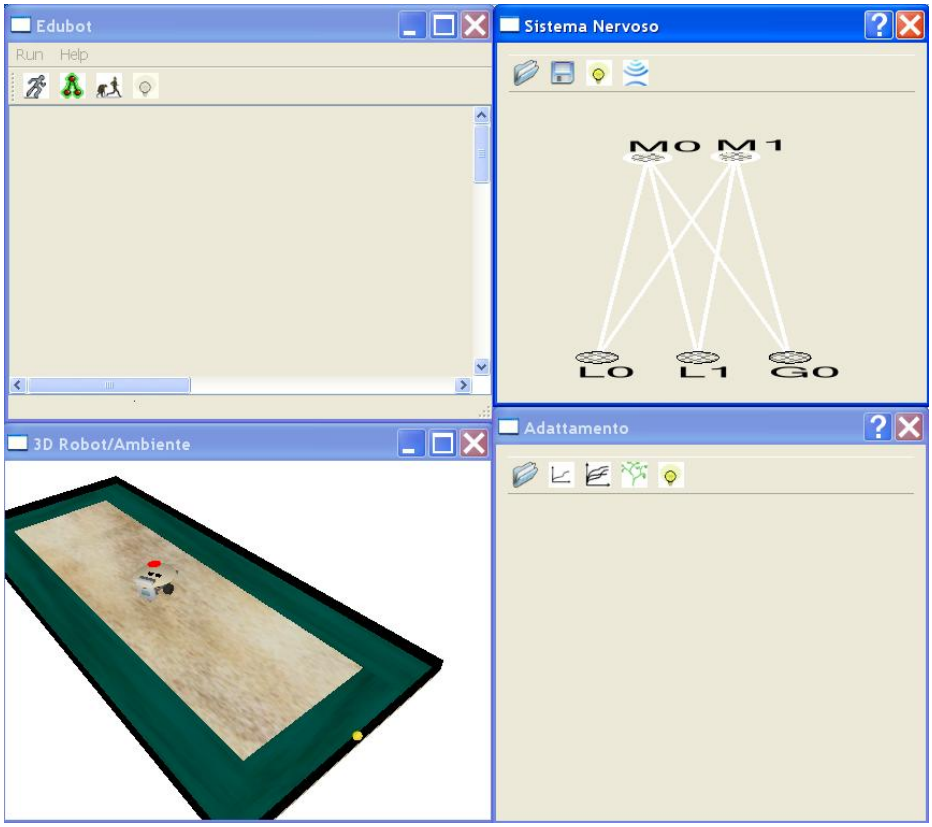


Figura 3.7. L'ambiente software EduBot al lancio dell'esperienza di apprendimento ESPLORA.

Il **sistema nervoso** del robot è costituito, come possiamo vedere nella finestra in alto a destra della Figura 3.7, da una semplice rete neurale in cui i tre neuroni sensoriali, che corrispondono ai due sensori di luce passivi posizionati sul lato frontale sinistro e destro del robot (L0 e L1) e al sensore del colore del pavimento posizionato sul lato frontale del robot (G0) sono connessi ai due motori (M0 e M1) che controllano la velocità e la direzione di rotazione della ruota destra e sinistra. Il colore bianco delle connessioni sta tuttavia che si tratta di connessioni con peso sinaptico pari a 0, vale a dire connessioni ininfluenti. Analogamente, il fatto che il colore dei neuroni motori è bianco, sta ad indicare il fatto che l'attività spontanea di tali neuroni è nulla. Tutto questo implica che, dato questo sistema nervoso iniziale, i due motori rimarranno fermi e lo stato dei tre neuroni sensoriali non influenzerà lo stato dei motori.

L'utente ha la possibilità di trasferire il sistema nervoso visualizzato nella finestra corrispondente sul robot cliccando sul pulsante  e osservare dunque il comportamento di un robot dotato di tale sistema nervoso nel mondo reale. La forza e

il segno delle connessioni (cioè il fatto che una connessione può essere attivatoria oppure inibitoria) può essere modificato semplicemente selezionando la connessione corrispondente con il mouse e premendo ripetutamente i tasti "+" o "-" sulla tastiera. Analogamente l'attività spontanea di un neurone motorio può essere modificata selezionando il neurone (cliccandoci sopra con il mouse) e utilizzando i tasti "+" o "-". Il colore (rosso per le connessioni attivatorie e blu per le connessioni inibitorie) e l'intensità del colore (più o meno intenso a seconda del valore corrispondente) permette all'utente di osservare immediatamente gli effetti delle modifiche effettuate e l'insieme delle caratteristiche del sistema nervoso corrente. L'utente ha inoltre la possibilità di osservare il comportamento del robot in simulazione cliccando sul pulsante  posto nella finestra EduBot, variare i valori delle connessioni del sistema nervoso mentre il robot è in movimento, osservando dunque immediatamente gli effetti delle variazioni sul comportamento risultante.

Una volta che il robot è in movimento nell'ambiente reale o nell'ambiente simulato, l'utente ha anche la possibilità di osservare come lo stato dei sensori e dei motori varia nel tempo mentre il robot interagisce con l'ambiente. Nel caso del robot reale, tali informazioni vengono visualizzate sul display del computer di bordo del robot. Nel caso del robot simulato, lo stesso tipo di informazioni vengono visualizzate nella Finestra Edubot. Ciò consente all'utente di osservare il comportamento del robot non solo dal punto di vista esterno ma anche dal punto di vista del robot, cioè dal punto di vista dello stato dei sensori e dei motori del robot.

Complessivamente, tali funzionalità consentono di creare robot analoghi ai veicoli di Braitenberg con poche operazioni elementari e di osservare il comportamento di tali robot (e del relativo sistema nervoso) nel mondo reale e in simulazione.

Il modulo di **adattamento** del robot consente all'utente di addestrare il robot a svolgere un determinato compito, per es. esplorare il maggior numero di porzioni di un ambiente in un tempo determinato. Al fine di consentire agli utenti di sperimentare con tale tipo di processo nel modo più semplice ed efficace possibile, l'ambiente EduBot presenta la possibilità di utilizzare sia un metodo evolutivo basato su riproduzione selettiva e variazioni, sia un metodo di apprendimento per prove ed errori basato su un algoritmo di hill-climbing stocastico (Russell e Norvig, 2003). In

particolare, cliccando sul pulsante  , l'utente può sottoporre il sistema di controllo corrente del robot ad un processo di apprendimento in cui: (1) le prestazioni del robot con il sistema nervoso corrente vengono valutate rispetto a un compito dato (per es., nel caso di un compito di esplorazione, rispetto alla percentuale di ambiente esplorata dal robot durante una serie di prove effettuate in simulazione); (2) i valori dei pesi delle connessioni e dei biases (i valori che determinano l'attività spontanea dei neuroni) del sistema nervoso del robot vengono variati casualmente, (3) le prestazioni del robot dopo le variazioni vengono ri-valuatate, (4) qualora le variazioni abbiamo prodotto un peggioramento delle prestazioni, esse vengono eliminate, (5) si ripetono le fasi 1-4 fino ad ottenere un robot in grado di svolgere il compito in modo efficace. Durante il processo di apprendimento che può essere interrotto o ripreso in qualsiasi istante, l'utente ha la possibilità di osservare come variano le caratteristiche del sistema nervoso nella finestra corrispondente, come variano le prestazioni del robot durante il processo di apprendimento nella finestra adattamento, e come varia il comportamento risultante del robot in simulazione o nell'ambiente reale attraverso i comandi descritti sopra. L'utente ha anche la possibilità di combinare insieme la progettazione esplicita delle caratteristiche del sistema nervoso e l'adattamento, impostando a mano il valore delle connessioni sinaptiche prima del processo di apprendimento, oppure modificando i valori manualmente dopo aver sottoposto il robot ad un processo di apprendimento o evoluzione. Infine, l'utente ha la possibilità di osservare in tempo reale come i valori delle connessioni e le prestazioni dei robot rispetto al compito dato variano durante il processo di apprendimento/evoluzione (la finestra **Sistema Nervoso** infatti viene utilizzata per visualizzare graficamente il tipo e il valore delle connessioni del robot allo stato corrente del processo di apprendimento).

Il modulo di simulazione corrispondente alla finestra **robot/ambiente** consente di osservare il comportamento del robot in simulazione nonché di manipolare la posizione del robot nell'ambiente e le caratteristiche dell'ambiente. Tale modulo svolge una serie importante di funzioni. Innanzi tutto rappresenta un elemento essenziale per poter effettuare il processo di apprendimento. Infatti, valutare il comportamento del robot in simulazione consente di realizzare in pochi minuti o

poche ore un processo che nell'ambiente reale durerebbe ore o giorni. In secondo luogo consente di fare degli esperimenti in modo più semplice e rapido rispetto all'uso del robot fisico, per es. di sperimentare un particolare tipo di ambiente senza doverlo necessariamente costruire fisicamente. In terzo luogo consente di osservare la relazione tra il comportamento del robot e lo stato del sistema nervoso in modo più semplice ed efficace. Infine, considerata la stretta corrispondenza tra il robot reale e il robot simulato, non pregiudica troppo la concretezza delle sperimentazioni fatte rispetto ad un ambiente unicamente software.

Complessivamente, tali funzionalità consentono di sottoporre in modo estremamente semplice un robot ad un processo di apprendimento/evoluzione e, soprattutto, di poter osservare la dinamica di tale processo da diversi punti di vista (e in particolare, dal punto di vista della capacità del robot di svolgere la funzione richiesta, dal punto di vista del tipo di comportamento esibito, e dal punto di vista delle caratteristiche del sistema nervoso).

Riassumendo, lo sviluppo dell'ambiente software di EduBot ha comportato un re-design quasi completo dell'interfaccia grafica, lo sviluppo di strumenti di visualizzazione e editing semplici ed intuitivi, lo sviluppo di nuove funzionalità non presenti in Evorobot* quali l'algoritmo di apprendimento.

L'ambiente EduBot preserva inoltre tutte le funzionalità di Evorobot*, come per es. la possibilità di sperimentare con comportamenti collettivi, la possibilità di utilizzare piattaforme robotiche diverse, la possibilità di replicare e variare una ricca serie di esperimenti, la possibilità di creare altre esperienze guidate di apprendimento. A titolo esemplificativo, la Figura 3.8 mostra come sia possibile effettuare esperimenti con 10 robot situati nello stesso ambiente semplicemente cambiando uno dei parametri.

Figura 3.8. Un esempio di esperimento condotto utilizzando 10 diversi robot situati nello stesso ambiente. I cerchi di colore diverso sopra i robot servono a distinguere i singoli individui.

3.5 Caratteristiche innovative e ambiti di applicazione

EduBot presenta una serie di caratteristiche innovative rispetto agli ambienti di robotica educativa disponibili che consentono di ampliare lo spettro di applicazione della robotica educativa allo studio del comportamento (come caratteristica emergente dall'interazione tra il robot e l'ambiente), del sistema nervoso, e dell'apprendimento.

La maggior parte degli ambienti di apprendimento passati in rassegna nel capitolo precedente partono dall'assunzione che il comportamento di un robot non sia altro che la manifestazione di ciò che è contenuto all'interno del sistema di controllo del robot. In altre parole i robot sono concepiti come dei meri esecutori di istruzioni compilate dal programmatore e contenute all'interno del computer di bordo del robot. Per dimostrare quanto sia forte questa assunzione, riportiamo qui di seguito il testo di una delle lezioni introduttive alla robotica contenute nei corsi basati su Robot LEGO NXT creati e distribuiti dalla Robotics Academy della Carnegie Mellon University, USA. Si tratta del testo di commento di un video introduttivo del corso:

> *Robots are made to perform useful tasks. Each one is designed to solve a specific problem in a specific way. This robot* [le immagini mostrano un trattore che si muove in modo in un campo] *solves the problem of safely driving a tractor through a field by moving towards the destination, and making small detours if any obstacles get in its way. This robot* [le immagini mostrano ora un robot lego che si muove all'interno di un labirinto con delle line disegnate sul pavimento] *solves the problem of getting through this maze by moving in timed segments. Let's take a closer look at this last robot. How does it do that ? How does it know to do that ? Creating a successful robot takes a team effort between humans and machines. The human is responsible for identifying the task, planning out a solution, and then explaining to the robot what it need to do to reach the goal. The machine is responsible for following the instruction it is given, and thereby carrying out the plan. Because humans an machines don't not normally speak the same language a special language must be used to translate the necessary instructions from human to robot. There are many such languages, with ROBOTC being one of them. These human-to-robot languages are called "programming languages and instructions written in them are called "programs". The human who write the program is called the programmer. The programmer's job therefore, is to identify the problem that the robot must solve, to create a plan to solve it, and to write that plan in a program that the robot will be able to understand. The program is loaded onto the robot and then the robot runs the program, and accomplish the task it was given. Finally, take note: the robot only follow the program, it does not think for itself. Just a seat can be no stronger than it is built, the robot can be no smarter than the program that the human programmer gave it. You as programmer, will be responsible for planning and describing to the robot exactly what it need to do to accomplish its task.*

L'idea di base che si evince chiaramente da questo testo è che il robot è una macchina che si limita ad eseguire delle istruzioni. L'ambiente software sviluppato e utilizzato per programmare i robot è dunque un normale ambiente di programmazione esteso con alcune istruzioni in grado di leggere lo stato dei sensori e determinare lo stato dei

motori. Il robot è concepito come una macchina programmabile, analoga a un computer, piuttosto che come un sistema ontologicamente diverso; un sistema dotato di un corpo e situato in ambiente esterno con il quale interagisce più simile ad un insetto piuttosto che a un computer o ad una lavatrice.

Tale assunzione implica che per sviluppare un robot in grado di esibire un comportamento desiderato l'utente dovrà: (1) immaginare il comportamento desiderato (cioè il comportamento adatto svolgere la funzione o le funzioni stabilite), (2) descrivere tale comportamento in linguaggio naturale, (3) definire ciascuna parola o frase utilizzata attraverso una serie di parole/istruzioni più semplici fino ad arrivare a istruzioni sufficientemente semplici da poter essere espresse attraverso i comandi disponibili nel linguaggio di programmazione:

> *"Behaviors" are very convenient way to talk about what the robot is doing, and what it must do. Moving forward, stopping, turning, looking for an obstacle, these are all behaviors. Some behaviors are big, "solve the maze." Some behaviors are small, "Turn on your left motor." As you will see in this vide, big behaviors are actually made up of the smaller ones. As you begin the task of programming, you should also begin thinking about the robot's actions in terms of behaviors. Now recall: as programmer, your primary responsabilities are first to formulate a plan for the robot to reach the goal, and then, to translate that plan into a program that the robot can follow. The plan will simply be the sequence of behaviors that the robot need to follow, and the program will just be those behaviors translated into the programming language. To find a solution, start by examining the problem. Try to see what the robot need to do, at a high level to accomplish the goal. Having the robot follow this path, for example, would solve the problem. Bingo, you've just identified the first behavior you need. Write it down. Now. start trying to break that behavior down into smaller parts. Following this path involves moving forward here, then turning, then moving forward again, turning the other way, and so on. Each of these smaller actions is also a behavior. Write them down as well, taking care to keep them in their correct sequence. Break those down again, and you'll get smaller and smaller behaviors with more and more detail. Keep track of the way you go. Eventually, you'll reach commands that you can express directly in the programming language. For example. ROBOTC has a command to turn on one motor. When you reach a behavior that says*

to turn on one motor you can stop breaking it down, because it's now ready to translate. When all the pieces have reached this level of detail take a look at the lst you've made. These behaviors, in the order and way you've specified them, represent the plan that the robot needs to follow in order to accomplish its goal. Because the steps are still written in English, or if you prefer as a diagram, they should be relatively easy to understand for the human programmer. As the programmer becomes more experienced, the organization of the behaviors in English will start to include important techniques from the programming language itself like if-else statements and loops. This hybrid language, halfway between English and the programming language, is called pseudocode and is an important tool in helping to keep larger program understandable. By starting with a very large solution behavior and breaking it down into smaller and smaller sub-behaviors, you have a logical way to figure out what the robot needs to do in order to accomplish the task. By recording the behaviors in English, you have taken the first step toward good pseudocoding practice, allowing you easily review the behaviors and their organization as you prepare to translate them to program code. The only step remaining is to translate your behaviors from English pseudocode in ROBOTC programming language.

EduBot al contrario parte dall'assunzione che il comportamento di un robot, e più in generale di un agente dotato di un corpo e situato in un ambiente esterno con il quale interagisce, sia il risultato emergente dell'interazione tra il robot e l'ambiente. Una impostazione è in linea con le ricerche di frontiera nell'ambito della robotica autonoma (Nolfi, 2009). Tale assunzione implica che il sistema di controllo di un robot, o meglio il sistema nervoso di un robot, non controlla il comportamento del robot ma piuttosto regola l'interazione tra robot e ambiente in modo tale che emerga il comportamento appropriato. Ciò ha una serie di implicazioni importanti. In primo luogo il fatto che il sistema nervoso del robot, che riceve informazioni quantitative dai sensori e impartisce ordini quantitativi ai motori, deve essere in grado di operare in modo quantitativo (e non semplicemente attraverso istruzione del tipo IF THEN). In secondo luogo, il fatto che il processo di design del sistema nervoso del robot deve essere condotto osservando il comportamento risultate dall'interazione tra il robot e l'ambiente.

L'utilizzo del formalismo delle reti neurali, per implementare il sistema di controllo del robot, e la possibilità di variare ripetutamente le caratteristiche della rete neurale (eventualmente attraverso un processo di adattamento automatico analogo all'evoluzione o ad una forma di apprendimento per prove ed errori) e dell'ambiente osservando contemporaneamente gli effetti di tali variazioni sul comportamento risultante rendono EduBot profondamente diverso dagli altri ambienti software sviluppati in ambito educativo.

Questo cambio di prospettiva presenta tre importanti ricadute dal punto di vista educativo. In primo luogo può consentire di utilizzare il robot come un modello semplificato di un organismo naturale aprendo dunque la possibilità di estendere l'area di applicazione della robotica educativa allo studio di aspetti relativi alla biologia e alla psicologia. In secondo luogo permette ai ragazzi di riflettere sulle somiglianze e le differenze tra gli organismi viventi e gli artefatti anche da un punto di vista bio-ingegneristico, in cui la natura rappresenta una sorgente di ispirazione fondamentale per lo sviluppo di soluzioni tecnologiche. In terzo luogo può consentire ai ragazzi di acquisire conoscenze teorico/pratiche sui sistemi complessi, cioè su sistemi che esibiscono un comportamento che è il risultato emergente di un gran numero di interazioni.

4. Edubot: Esperienza guidata di apprendimento

4.1 Introduzione

In questo capitolo descriviamo il materiale curricolare sviluppato. Si tratta di una esperienza di apprendimento guidata, della durata complessiva di 120 minuti, rivolta ai ragazzi della scuola media inferiore. L'esperienza è organizzata in una sessione introduttiva e 5 sezioni sperimentali (descritte nelle sezioni 4.2-4.7).

Ciascuna sezione (con l'esclusione della sessione introduttiva) prevede una fase sperimentale (che riguarda l'effettuazione di esperimenti esplorativi o volti al raggiungimento di uno scopo), una fase di analisi del processo in corso, e una fase di riflessione/consolidamento.

4.2 Cosa è un robot

I robot sono oggetti fisici dotati di **corpo**, **sensori**, **attuatori**, e **sistema di controllo.**

I **sensori** sono dei componenti che, analogamente alle cellule sensoriali localizzare nei nostri occhi, naso, bocca, e pelle, consentono ai robot di percepire delle informazioni dall'ambiente circostante. Questi robots (Figura 4.1), in particolare sono dotati di un sensore di luce "G0" posto sul lato frontale che è in grado di rilevare il colore del pavimento, di due sensori di luce "L0 e L1" posti sul lato frontale sinistro e destro in grado di rilevare l'intensità della luce ambientale.

Gli **attuatori** sono dei componenti che, analogamente ai nostri muscoli, consentono al robot di modificare la propria posizione rispetto all'ambiente. I nostri robot in particolare sono dotati di due motori "M0" e "M1" che controllano la velocità di rotazione di due ruote.

Il **sistema di controllo** è un programma software contenuto all'interno di un piccolo computer (NXT) collegato ai sensori e ai motori attraverso dei cavi che analogamente al nostro sistema nervoso determina lo stato degli attuatori sulla base dello stato corrente e precedente dei sensori.

Il **corpo** è la struttura rigida circolare che sostiene i sensori, gli attuatori, il sistema di controllo e altri componenti, come per es. la batteria contenuta all'interno dell'NXT, che svolgono un ruolo analogo ai nostri organi interni (cuore e fegato, per es.).

I robot si trovano in un **ambiente** con il quale interagiscono. Nel nostro caso l'ambiente è costituito da una arena rettangolare circondata da pareti. Il pavimento della porzione centrale dell'arena e' bianco mentre nelle zone adiacenti alle pareti è nero. L'ambiente inoltre contiene una lampada ad incandescenza di 25W posizionata sul lato sinistro dell'arena ad una altezza di circa 30cm dal pavimento.

Sul computer è attivo il programma Evorobot (Figura 2) che ci permette di programmare il sistema di controllo del robot, osservare il comportamento del robot in simulazione, o osservare come il robot sviluppa autonomamente le proprie capacità attraverso un processo di apprendimento per prove ed errori simile a quello con cui gli organismi naturali sviluppano le proprie capacità.

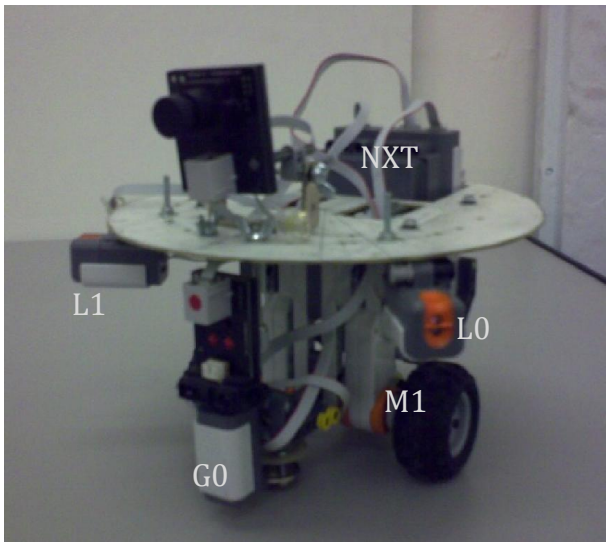


Figura 4.1. Il robot. Nell'immagine possiamo identificare il computer di bordo (NXT) che contiene anche le batterie ricaricabili, il sensore di colore del pavimento (G0), i due sensori di luce ambientale (L0 e L1), e i due motori che controllano le due ruote (M0 e M1). Il robot mostrato in questa immagine monta anche una sensore di prossimità ad ultrasuoni e una telecamera sopra il sensore G0, che tuttavia non utilizzeremo negli esperimenti descritti in questo documento.

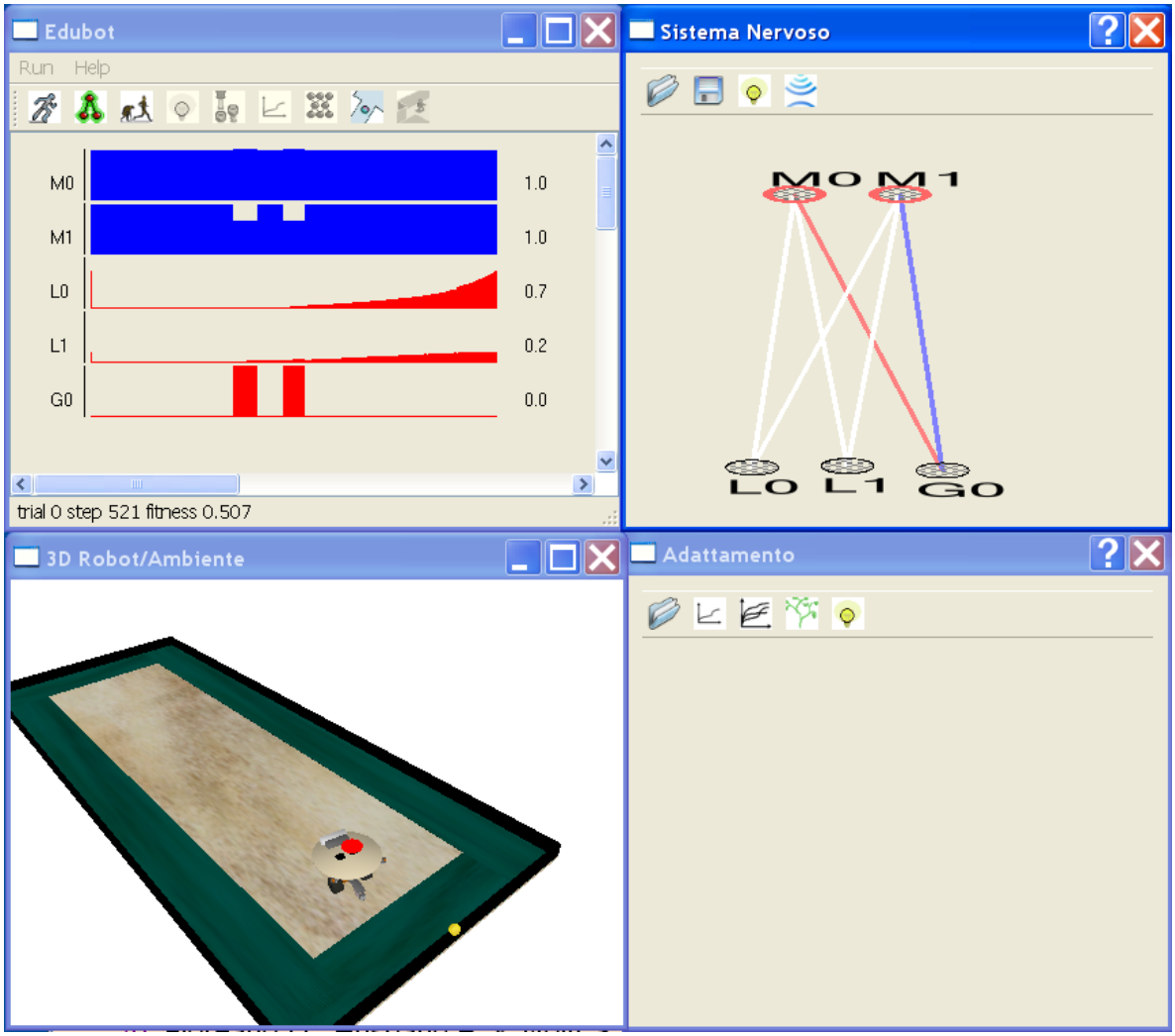


Figura 4.2. Il programma Evorobot come appare eseguendo il file "run" dalla directory "LEGO".

4.3 Percepire il mondo

Osserviamo ora come i sensori permettono al robot di ricevere informazioni dal mondo esterno.

Esperimento 1. Trasferiamo ora il sistema di controllo visualizzato nella finestra "controller" sul computer di bordo del robot cliccando sull'icona  *(dopo aver acceso il robot tenendo premuto il pulsante arancione posto sull'NXT per qualche secondo). Tale sistema di controllo, visualizzato nelle finestra controller, è costituito da tre neuroni sensoriali (L0, L1, G0) il cui stato di attivazione varia in base al livello di eccitazione dei*

sensori corrispondenti e da due neuroni motori (M0 e M1) che hanno un valore di attivazione costante pari a 0.5. Lo stato di attivazione dei sensori e dei motori viene visualizzato sul piccolo display posto sul lato posteriore del computer di bordo del robot (NXT). Spostiamo e ruotiamo il robot nell'ambiente e osserviamo come lo stato dei neuroni sensoriali cambia al variare della relazione spaziale tra robot e ambiente.

Alternativamente possiamo osservare il comportamento dei sensori del robot in simulazione nella finestra Evorobot cliccando sull'icona*  *, muovendo il robot nell'ambiente simulato con i tasti freccia, e ruotando il robot premendo il tasto "R". Il comando*  *(che consente di osservare il comportamento del robot in simulazione può essere interrotto cliccando sull'icona STOP). Durante il processo di simulazione il robot viene lasciato interagire con l'ambiente per 10 periodi (prove) ciascuna dei quali dura circa 3m. All'inizio di ciascuna prova il robot viene posto con un orientamento casuale in una posizione casuale dell'arena.*

<u>Osservazioni 1</u>. Ci sono alcune cose che il nostro robot può percepire e altre che non è in grado di percepire. Sapresti dire quali ? Proviamo ad osservare il mondo attraverso i sensori del nostro robot. Quali stati sensoriali identificano in modo univoco un oggetto o una situazione. Ci sono stati "ambigui" cioè stati sensoriali simili o identici che tuttavia vengono percepiti in situazioni diverse ? E se si, per quale ragione si verificano queste situazioni ambigue ?

4.4 Agire nel mondo

I robot non si limitano ad osservare il mondo attraverso i propri sensori, come abbiamo visto nell'esperimento precedente, ma eseguono delle azioni che modificano la posizione del robot rispetto all'ambiente o l'ambiente stesso. Il nostro robot è dotato di due motori che consentono di far ruotare le due ruote corrispondenti in avanti o all'indietro a diverse velocità.

Affinché le ruote del robot si muovano è necessario che i due neuroni motori "M0" e "M1" assumano uno stato di attivazione maggiore o minore di 0.5 (lo stato 0.5 infatti produce invece un movimento nullo). Per valori maggiori di 0.5, maggiore e'

l'attivazione maggiore è la velocità in avanti e viceversa per valori minori di 0.5 minore e' l'attivazione maggiore e' la velocità indietro. E possibile variare lo stato di attivazione di un neurone motorio modificandone il **bias**, un parametro che determina l'attività spontanea del neurone.

Esperimento 2. Per variare il bias dei neuroni motori, seleziona un neurone motorio ("M0" o "M1") nella finestra "controller" cliccando sopra il cerchio corrispondente e poi premi i tasti "+" o "-" ripetutamente. Infine deseleziona il neurone cliccando su una zona vuota della finestra. Poi eventualmente modifica il bias dell'altro neurone ripetendo la stessa procedura. Trasferisci ora il sistema nervoso cosi' modificato sul robot cliccando sull'icona  *e osserva il comportamento esibito dal robot. Alternativamente, osserva il comportamento del robot in simulazione, cliccando sull'icona*  *. Puoi variare il valore dei bias anche mentre il robot si sta muovendo in simulazione. Inoltre puoi spostare il robot indietro, quando questo rimane bloccato contro una parete. Puoi rallentare o accelerare la simulazione premendo i tasti "-" e "+". Prova a identificare i valori dei bias che consentono al robot di esibire i seguenti comportamenti: (1) andare dritto in avanti o indietro, (2) girare sul posto o muoversi lungo una traiettoria circolare piu' o meno grande.*

Osservazioni 2. Anche se il robot produce sempre la stessa azione, indipendentemente da dove si trova e dallo stato dei propri sensori, il comportamemto esibito dal robot è spesso il risultato dell'interazione tra robot e ambiente. Prova a identificare come le caratteristiche dell'ambiente e la posizione relativa del robot nell'ambiente influenzano il comportamento o i comportamenti esibiti dal robot.

4.5 Reagire agli stimoli sensoriali

La possibilità di modulare il tipo di azione eseguita sullo base dello stato dei sensori permette al robot di esibire comportamemnti più complessi e di variare il proprio comportamento sulla base delle caratteristiche della porzione dell'ambiente in cui si trova. In effetti, il sistema di controllo del robot serve proprio a questa funzione:

variare le azioni eseguite sulla base dello stato corrente (e eventualmente precedente) dei sensori.

Gli elementi che consentono ai neuroni sensoriali di influenzare i neuroni motori sono le **connessioni** (le 6 linee bianche che uniscono i tre neuroni sensoriali ai tre neuroni motori). Ciascuna connessione è caratterizzata da un **peso** (cioè un valore numerico che varia da -5.0 a + 5.0). Maggiore è il valore del peso maggiore è l'influenza dell'attività del neurone sensoriale sul corrispondente neurone motorio. Se il peso è positivo, lo stato di attivazione del neurone sensoriale produrrà un aumento dello stato di attivazione del neurone motorio al quale è connesso. Viceversa, se il peso è negativo, lo stato di attivazione del neurone sensoriale produrrà una diminuzione dello stato di attivazione del neurone motorio al quale è connesso.

Attualmente i pesi delle connessioni sono tutti 0.0. Questo significa che l'influenza dei neuroni sensoriali sui neuroni motori è nulla. Possiamo tuttavia modificare i pesi delle connessioni in modo da consentire al robot di reagire agli stimoli sensoriali.

Esperimento 3. Modifichiamo i pesi delle due connessioni che uniscono il neurone sensoriale "G0" ai motori "M0" e "M1" e i pesi dei bias dei neuroni motori cercando di scegliere dei valori che consentono al robot di muoversi nell'ambiente senza sbattere contro le pareti dell'arena. Possiamo selezionare una connessione cliccandoci sopra con il mouse e poi aumentarne/diminuirne il valore con i tasti "+" e "-". Trasferiamo ora il sistema nervoso modificato sul robot e osservarne il comportamento. Alternativamente, osserviamo il comportamento del robot in simulazione. Le modifiche dei pesi e dei bias influenzano immediatamente il comportamento del robot in simulazione. Per far si che le modifiche abbiamo effetto sul robot fisico dobbiamo invece trasferire il sistema di controllo modificato sul robot cliccando sul sull'icona . Continuiamo a modificare i valori dei pesi delle connessioni e i bias dei due neuroni motori finché non otteniamo il comportamento desiderato.

Osservazioni 3. La possibilità di modulare il tipo di azioni prodotte sulla base dello stato dei sensori consente al robot di esibire comportamenti diversi in situazioni diverse. In particolare un comportamento di evitamento degli ostacoli in prossimità delle pareti e

un comportamento di avanzamento lontano dalle pareti che vengono eseguiti rispettivamente quando il sensore di ground ha un valore pari a 1.0 o 0.0 rispettivamente. Si noti tuttavia che sebbene questi due comportamenti siano esibiti in situazioni diverse, essi interagiscono tra di loro. In particolare, l'angolo più o meno stretto con cui il robot "rimbalza" sui bordi dell'arena durante il comportamento di evitamento degli ostacoli influenza il comportamento esibito dal robot lontano dagli ostacoli. A seconda del modo in cui il robot evita gli ostacoli, infatti, il robot tenderà ad esplorare una zona più o meno ristretta dell'ambiente. Si noti inoltre come il comportamento esibito dal robot in una certa fase dipenda dalla posizione e dall'orientamento iniziale del robot che a sua volta dipende dai comportamenti esibiti precedentemente dal robot e, in ultima analisi, dalla posizione e orientamento con cui il robot è stato posto nell'ambiente.

4.6 Svolgere un compito

Cerchiamo ora di creare un robot in grado di svolgere un compito. Per es., immaginiamo di voler costruire un robot tagliaerba che tagli efficacemente l'erba del nostro giardino. Immaginiamo inoltre che il nostro giardino sia, analogo all'arena rettangolare illustrata sopra, e che il nostro robot sia dotato, come i comuni tagliaerba, di una lama rotante. Per svolgere efficacemente questo compito il robot dovrà visitare il maggior numero possibile di porzioni di "giardino" entro un tempo limite.

Quando testiamo il comportamento del robot in simulazione, il programma visulizza nella parte inferiore della finestra "Edubot" la prestazione (fitness) del robot rispetto a questo compito, cioè la percentuale delle porzioni di spazio esplorato nella prova corrente e nelle prove precedenti.

Esperimento 4. Proviamo ora a modificare i valori dei pesi delle connessioni e dei bias cercando di massimizzare la percentuale di spazio esplorata dal robot. Confrontiamo le prestazioni ottenute dal nostro robot e il tipo di comportamento esibito con quello degli altri.

Osservazioni 4. Per far si che il robot si muova senza urtare contro le pareti è necessario esibire due comportamenti, evitare gli ostacoli e muoversi in modo rettilineo o curvilineo, e scegliere il comportamento giusto al momento giusto. Ciò può essere ottenuto aumentando l'attività spontanea dei due neuroni motori (in modo che il robot tenda a muoversi in avanti) e connettendo il sensore di ground con una connessione attivatoria e inibitoria ai due neuroni. L'alternanza tra i due tipi di comportamenti viene realizzata grazie al fatto che quando il sensore di ground non è attivo (cioè quando il robot si trova su un'area bianca) l'influenza di tale sensore sui motori è nulla, e dunque il robot si muove in avanti grazie all'attività spontanea dei neuroni motori. Quando invece il robot si trova nell'area nera, il sensore di ground produce un aumento dell'attività del primo neurone e una diminuzione dell'attività del secondo neurone motorio che consentono al robot di girare fino ad uscire dall'area nera per poi ritornare al comportamento di movimento in avanti. Si noti come e' necessario che il robot si muova in avanti e non indietro perché i sensori sono posti sul lato frontale.
Si noti infine che per esplorare l'ambiente è opportuno evitare le pareti "rimbalzando" su di esse con un certo angolo. Tale angolo dipende dal rapporto quantitativo tra l'attività spontanea dei neuroni e l'intensità dei pesi delle connessioni.

4.7 Robot che apprendono

Fin qui abbiamo visto come sia possibile "programmare" un robot ad esibire uno o più comportamenti (evitare gli ostacoli, explorare l'ambiente) in grado di svolgere una funzione (rasare l'erba di un giardino). Durante il processo di programmazione, come abbiamo visto, l'utente modifica manualmente le caratteristiche del sistema di controllo del robot cercando di massimizzare la prestazione del robot rispetto alla funzione prestabilita.

Un metodo alternativo per sviluppare robot in grado di svolgere una certa funzione consiste nel consentire al robot stesso di migliorare la propria capacità attraverso un processo analogo ai processi di apprendimento che consentono a molti animali di sviluppare e di migliorare le proprie capacità nel corso della propria vita.

Un modo semplice per realizzare tale processo di apprendimento consiste nel dotare i robot delle capacità di modificarsi autonomamente attraverso un processo per prove ed errori in cui (1) i pesi dei bias e delle connessioni vengono modificati casualmente, (2) le modifiche che producono un aumento della prestazione rispetto al compito vengono conservate, (3) le modifiche che producono un peggioramento della prestazione rispetto al compito vengono scartate, (4) tale processo viene ripetuto per un gran numero di volte.

Per consentire ai robot di apprendere sviluppando e/o a migliorando la proprie capacità di esplorare l'ambiente clicchiamo sull'icona  . Durante l'esecuzione di questo comando, il programma Evorobot* simula la vita di un robot (che nel mondo reale avrebbe una durata di circa 30m) in meno di un secondo. Nella finestra "sistema nervoso" potremo osservare come i valori dei pesi e delle connessioni (indicati dal tipo e dall'intensità del colore) varia durante il processo di apprendimento.

Nella finestra "adattamento" possiamo vedere invece come a prestazione del robot varia, durante il processo di apprendimento. Infine nella parte inferiore della finestra "Edubot" possiamo vedere quante variazioni sono state fatte finora e il numero di variazioni preservate e scartate (cioe' il numero di variazioni che hanno prodotto miglioramenti o peggioramenti delle prestazioni).

<u>Esperimento 5.</u> Sottoponiamo ora il nostro robot, con i valori dei pesi e dei bias modificati manualmente da noi, al processo di adattamento. Osserviamo come cambiano le caratteristiche del sistema nervoso del robot e le prestazioni del robot rispetto al compito scelto durante tale processo. Poi interrompiamo il processo di apprendimento e osserviamo il comportamento del robot. Quali sono gli effetti delle modifiche introdotte durante il processo di apprendimento? C'e' stato un miglioramento ? A cosa è riconducibile tale miglioramento ? Confrontiamo il nostro robot con quello degli altri cercando di identificare somiglianze e differenze ?

<u>Osservazioni 5.</u> In primo luogo le modifiche introdotte durante il processo di apprendimento hanno consentito sia di migliorare il modo con cui il robot rimbalza sulle pareti (al fine di esplorare l'ambiente). Almeno in alcuni dei robot addestrati sembrano

aver scoperto una strategia per esplorare efficacemente l'ambiente che consiste nel muoversi in alto e in basso ripetutamente spostandosi progressivamente del lato destro al lato sinistra dell'ambiente e, una volta raggiunto il lato sinistro, ripartire dal lato destro. Perché questa strategia è efficiente ? Quali caratteristiche del sistema nervoso del robot permettono di produrre questo tipo di comportamento ?

Conclusioni

In questa tesi abbiamo illustrato che cosa è la robotica educativa, quali sono i fondamenti teorici sottostanti a questa area, quali sono gli ambienti di apprendimento più interessanti oggi a disposizione, quali sono i punti di forza e i problemi aperti nell'ambito di questa area di ricerca.

Nella maggior parte dei casi gli obiettivi formativi che vengono perseguiti attraverso lo sviluppo e l'utilizzo di ambienti di apprendimento robotici riguardano la robotica stessa o le cosiddette discipline STEM (Scienze, Tecnologia, Ingegneria e Matematica). Inoltre, nella maggior parte dei casi, il robot viene concettualizzato come un oggetto programmabile che si limita ad eseguire le istruzioni contenute nel programma.

In questa tesi, invece, prendendo spunto dal lavoro di Valentino Braitenberg e dalle più recenti tendenze nell'ambito della robotica autonoma, abbiamo focalizzato la nostra attenzione sull'uso dei robot come modelli semplificati di organismi biologici. Questa scelta implica un cambiamento piuttosto radicale del modo in cui il robot viene programmato, degli strumenti e del formalismo che vengono usati per svilupparlo, e del modo in cui il comportamento del robot viene interpretato. Il comportamento del robot non è riconducibile tanto a ciò che è contenuto nel sistema di controllo del robot ma piuttosto agli effetti delle interazioni tra il robot e l'ambiente. Il sistema di controllo del robot non è più costituito da una serie di istruzioni simboliche analoghe al linguaggio naturale, ma piuttosto da una serie di elementi che operano in modo quantitativo (i neuroni che formano il sistema di controllo del robot). Il processo di programmazione del robot non avviene attraverso l'identificazione del modo in cui il robot deve comportarsi, dal punto di vista dell'osservatore esterno, per risolvere un certo problema, ma piuttosto attraverso un processo incrementale per prove ed errori in cui le caratteristiche del sistema nervoso del robot vengono modificate in modo quantitativo sulla base dell'osservazione del comportamento esibito dal robot nell'ambiente.

Questo diverso modo di procedere consente di applicare la robotica educativa per acquisire una conoscenza diretta e sistemica dei fondamenti alla base del

comportamento e dell'intelligenza biologica: il ruolo del sistema sensoriale, del sistema motorio, e del sistema nervoso, la relazione tra agente e ambiente, il ruolo dell'apprendimento e dell'evoluzione biologica e soprattutto la relazione tra questi diversi fattori. A tal fine abbiamo sviluppato un ambiente di apprendimento basato su robot Lego, su reti neurali artificiali, e su algoritmi di evoluzione e apprendimento artificiali. Inoltre abbiamo sviluppato del materiale curricolare, rivolto ai ragazzi dei primi anni della scuola media inferiore, che consente di sperimentare con sistemi robotici di questo tipo con un costo iniziale di apprendimento dello strumento relativamente basso.

In futuro ci proponiamo di sperimentare tale ambiente in ambito scolastico e di espandere il materiale curricolare in modo da consentire di estendere il dominio dei temi trattabili.

Bibliografia

Battegazzore P. (2009). Fare robotica con un giocattolo programmabile a banalità limitata. In Andronico A. & Colazzo L. (Ed.) Atti del Convegno DIDAMATICA: Informatica per la didattica. Università degli Studi di Trento.

Beer R.D., Chiel H.J., Drushel R.F. (1999). Using Autonomous Robotics to teach science and engineering, Commun. ACM 42(6), 85–92.

Bransford J.D., Stein, B.S. (1993). The ideal problem solver. New York: Freeman.

Braitenberg, V. (1984). I veicoli pensanti: Saggio di psicologia sintetica. Milano: Garzanti.
Clark A. (1987). Being There: Putting Brain, Body and World Together Again. Cambridge, Ma: MIT Press. Trad. It. Dare corpo alla mente (1987). Milano: Mc-Graw Hill.

Brophy S., Klein, S., Portsmore, M., & Rogers, C. (2008). Advancing engineering education in P-12 classrooms. Journal of Engineering Education, 97(3), 369-387.

Brooks R. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2 (1): 14-23.

Clark A. (1997). Being There: Putting Brain, Body and World Together Again. Oxford University Press. Trad. It. (1999). Dare corpo alla mente. Milano: McGraw-Hill.

Druin A., Hendler J. (2000). Robots for Kids: Exploring New Technologies for Learning. Morgan, Kaufmann, San Francisco.

Han S., Bhattacharya K. (2001). Constructionism, learning by design, and project-based learning. In M. Orey (Ed). Emerging perspective on learning, teaching, and technology.

Department of Educational Psychology and Instructional Technology, University of Georgia

Imberman S.P. (2003) Teaching neural networks using lego handy board robots in an artificial intelligence course, SIGCSE '03: Proc. 34th SIGCSE Tech. Symp. Comp. Sci. Edu. (ACM, New York 2003) pp. 312–316

Martin F.G. (2001). Robotic explorations: A hands-on introduction to engineering. Prentice Hall, Upper Saddle River 2001.

Mataric M.J. (2004). Robotics education for all ages. Proceedings of the AAAI Spring Symposium on Accessible, Hands-on AI and Robotics Education. Palo Alto, CA Mar 22-24.

Mataric M. J. (2007). The Robotics Primer. Cambridge, MA: MIT Press.

Mataric M.J., Koenig N., Feil-Seifer D. (2007). Materials for Enabling Hands-on Robotics and STEM Education, Tech. Rep. SS-07–09, AAAI, Menlo Park.

Miller D.P. Nourbakhsh I.R., Siegwart S. (2008). Robots for Education. in Siciliano B., Oussama Khatib (eds.), *Handbook of Robotics*, Berlin: Springer Verlag,

Mindell D., Beland C., Chan W., Clarke D., Trupiano M. (2000). The Lego Mindstorm: Structure of Engineering Revolutions. Rapporto Tecnico.

Miglino O., Lund H.H., and Cardaci M. (1999). Robotics as an Educational Tool. In Journal of Interactive Learning Research 10:1, 25-48.

Miglino, O., Gigliotta, O., Ponticorvo, M., Nolfi, S. (2007). Breedbot: an edutainment robotics system to link digital and real world. In B. Apolloni, R.J. Howlett, L. Jain (Eds.) Knowledge-Based Intelligent Information and Engineering Systems. LNAI 4693, 74-81. Heidelberg: Springer

Nolfi S. (2009). Che Cos'è la Robotica Autonoma. Roma: Carocci Editore.

Nolfi S., Gigliotta O. (2010). Evorobot*: A tool for running experiments on the evolution of communication, in Nolfi S. & Mirolli M. (eds.), Evolution of Communication and Language in Embodied Agents, Berlin, Springer Verlag.

Papert S. (1980). *Mindstorms*: Children, Computers, and Powerful Ideas. New York: Basic Books. Trad. It. (1984) *Mindstorms. Bambini computer e creatività*. Emme ed.

Postmore M., Rogers C. & Pickering M. (2003). STOMP: Student Teacher Ourreach Mentorship Program. Proceedings of the 2003 American Society for engineering education annual conference & exposition, American Society for Engineering Education.

Russell S. J., Norvig P. (2003). Artificial Intelligence: A Modern Approach (2nd ed.), Upper Saddle River, New Jersey: Prentice Hall, pp. 111–114.

Risorse On-Line Selezionate

Piattaforme Hardware e Software

Bee-BOT: www.tts-group.co.uk

LEGO Mindstorm: www.lego.com/eng/education/mindstorms/default.asp

TETRIX: http://www.tetrixrobotics.com/Building_System/

VEX Robotics: http://www.vexrobotics.com/

Create: store.irobot.com/shop/index.jsp?categoryId=3311368

Cubelets: http://www.modrobotics.com/

BestBot: eutopia.unina.it/bestbot2/

PicoCrickets: http://www.picocricket.com

Evorobot*: laral.istc.cnr.it/evorobotstar/

RobotC: http://www.robotc.net/

NXT-G: http://www.ortop.org/NXT_Tutorial/index.html

LabView Education: www.ni.com/academic/education_edition/

Robotic Primer WorkBook: http://roboticsprimer.sourceforge.net/workbook/Main_Page

Microsoft Robotic Studio: msdn.microsoft.com/en-us/robotics/default

Materiale curricolare e formazione

Carnegie Mellon Robotic Academy: www.education.rec.ri.cmu.edu/

Center for Engineering Education and Outreach, **Tuft University:** www.ceeo.tufts.edu/

Student Teacher Outreach Mentorship Program at Tuft University: http://www.stompnetwork.org/tufts/

The European Project TERECoP: http://www.terecop.eu/

Scuola di Robotica: http://www.scuoladirobotica.eu/

Materiale Curricolare basato su LabView: http://zone.ni.com/devzone/cda/tut/p/id/10243

Risorse Educative (from iRobot): spark.irobot.com/educational_resources