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

Laboratorio di Robotica Avanzata
Advanced Robotics Laboratory

Corso di Robotica Mobile
(Prof. Riccardo Cassinis)

Guida all'installazione di Aria versione 2.7.0 su Acer Aspire One e collaudo su robot Tobor

Elaborato di esame di:

Luca Ceriani, Alessandro Caffi, Samuele Gatti, Ngassa Mosongo Tomson

Consegnato il: **16 giugno 2009**

Sommario

Il lavoro, che di seguito viene presentato, ha avuto come obiettivo la realizzazione di una vera e propria guida all'installazione di Aria nella versione 2.7.0 su mini calcolatore Acer Aspire One. A seguito di ciò è stato testato il collegamento del mini calcolatore al robot Tobor presente nel Laboratorio di Robotica Avanzata dell'Università degli Studi di Brescia. La valutazione ha riguardato inizialmente la connessione tramite radio modem e successivamente la connessione diretta via cavo usb-seriale.

1. Introduzione

La seguente relazione riporta il lavoro svolto durante il corso di Robotica Mobile per l'anno accademico 2008/2009. L'obiettivo centrale dell'elaborato ha riguardato la realizzazione di un manuale nel quale venissero riportati tutti i passi necessari per eseguire un'installazione di successo della suite Aria nella sua più recente versione (2.7.0) su calcolatore Acer Aspire One.

In relazione a tale obiettivo centrale, sono stati perseguiti degli obiettivi di natura secondaria, che di seguito si riportano:

- Descrizione delle novità presenti nella nuova versione di Aria (2.7.0) rispetto alla versione precedente.
- Collegamento tra il minicomputer Acer Aspire One e il robot Tobor.
- Testing della nuova suite Aria 2.7.0 sul robot Tobor.

2. Il problema affrontato

In questo capitolo vengono descritti i problemi affrontati durante lo svolgimento del lavoro, mentre le relative soluzioni, verranno prese in esame nel capitolo successivo.

L'intero lavoro è stato scomposto nei seguenti sottoproblemi indipendenti:

- Installazione del software sul calcolatore Acer Aspire One, in particolare:
    - Strumenti di compilazione.
    - Driver adattatore porta USB-Seriale.
    - Libreria Aria-2.7.0
    - Simulatore MobileSim-0.5.0
- Riadattamento della configurazione del robot Tobor.
- Installazione "on-board" del calcolatore Acer Aspire One sul robot Tobor.
- Testing.

2.1. Installazione del software sul calcolatore Acer Aspire One

L'obiettivo connesso all'installazione del software è stato quello di rendere il minicalcolatore un ambiente completo, sia dal punto di vista dell'esecuzione dei programmi che utilizzano la libreria Aria, sia per quanto concerne lo svolgimento dei compiti e delle simulazioni connesse al robot stesso.

Gli strumenti di compilazione installati sono quelli classici necessari per compilare programmi scritti in linguaggio C/C++ in ambiente Linux. In particolare, è stata necessaria l' installazione del compilatore "gcc-c++" e del programma "make", utilizzati successivamente sia per la compilazione dei moduli kernel necessari all'adattatore USB-Seriale, sia per la libreria Aria (oltre che per compilare programmi utente).

L'installazione del driver per l'adattatore USB-Seriale è stata indispensabile al fine di poter utilizzare la relativa periferica che permette il collegamento tra il calcolatore (dotato di porte USB) ed il robot (dotato di una porta seriale RS-232 femmina).

La parte centrale di tutto il processo di installazione del software ha riguardato il setup della libreria Aria. L'obiettivo da raggiungere ha previsto la sostituzione della vecchia versione Aria-2.5.1, con la più recente (Aria-2.7.0) nonché la verifica delle nuove funzionalità offerte e il testing del suo funzionamento.

Infine, è stata installato il simulatore MobileSim-0.5.0, dal momento che le funzionalità che esso offre hanno un particolare utilità, specialmente in fase di testing/sviluppo. Sul calcolatore viene eseguito un programma utente (pensato per funzionare con un robot reale), in grado di simulare il comportamento del robot, il tutto in modo completamente trasparente per il programma che riceve dal simulatore gli stessi feedback che riceverebbe dal robot reale.

2.2. Riadattamento configurazione Tobor

Durante la fase di posizionamento del minicalcolatore sulla schiena di Tobor è stato necessario operarne un riadattamento, focalizzando l'attenzione su due aspetti:

- Riadattamento della struttura del robot.
- Riadattamento della modalità di comunicazione con il calcolatore.

2.2.1. Riadattamento della struttura di Tobor

Nella figura sottostante è visibile la configurazione originale di Tobor, ovvero prima dell'elaborato in oggetto.

Dall'immagine si nota come sulla schiena del robot sia fissata una "coppa trofeo" che rende impossibile l'installazione del calcolatore "on-board"; pertanto, si è resa necessaria la rimozione dell'oggetto in questione.

2.2.2. Riadattamento del modalità di comunicazione con il calcolatore

In origine, la comunicazione del robot con il minicalcolatore, era realizzata tramite una coppia di radio modem: uno collegato al minicalcolatore e l'altro presente all'interno della Console di Tobor come si evince dalla figura (antenne sulla parte anteriore sinistra del robot). L'obiettivo è stato quello di permettere la comunicazione tra il robot ed il calcolatore attraverso la porta seriale 232 presente sul pannello superiore del robot stesso. Testato il funzionamento della porta, si è potuto montare l'Acer Aspire One direttamente sulla schiena del robot e realizzarne il collegamento attraverso un cavo adattatore USB-Seriale.

2.3. Installazione on-board del calcolatore Acer Apire One su Tobor

Per poter realizzare un collegamento tra la porta seriale del robot e la porta USB del calcolatore, è stato impiegato un apposito cavo-adattatore. E' stato escluso l'impiego di cavi eccessivamente estesi; le "prolunghe" sono state impiegate solamente in fase di test/verifica sotto attento controllo dell'operatore, dal momento che avrebbero potuto ostacolare il robot e mettere a rischio l'integrità del calcolatore. Il posizionamento del calcolatore direttamente sulla schiena del robot ha dato origine a due problematiche:

- individuazione sul robot dello spazio fisico necessario al posizionamento ottimale del calcolatore.
- ancoraggio del calcolatore alla struttura del robot in modo tale da evitare che le due apparecchiature potessero subire dei danni.

2.4. Testing

Terminata l'installazione del software ed effettuati i collegamenti opportuni tra il minicalcolatore ed il robot, è stato opportuno effettuare una serie di test per valutare il corretto funzionamento degli elementi installati. In particolare, è stato verificato:

- che i programmi scritti, i quali utilizzano le funzioni della nuova libreria Aria-2.7.0, venissero correttamente compilati sul calcolatore.
- che il robot si comportasse effettivamente come imposto dal programma utente, ovvero che il tutto venisse corretto a tempo di esecuzione. In tal senso, si sono evidenziati due scenari:
    1. testing della libreria attraverso il simulatore: se la simulazione è corretta, è possibile trarre conclusioni positive sia in relazione al funzionamento del simulatore che in relazione al funzionamento della libreria. In caso contrario, è opportuno analizzare il problema per determinare le cause del malfunzionamento.
    2. testing della libreria attraverso il robot reale: se il robot si comporta correttamente, possono essere tratte conclusioni positive sia sul funzionamento della libreria che sul funzionamento del collegamento tra il robot e il minicalcolatore. In caso negativo, ancora una volta è necessario analizzare il problema per determinarne le cause, che ovviamente possono essere differenti rispetto al precedente scenario.

L'impiego del robot reale ha avuto come diretta conseguenza il testing della solidità di ancoraggio del minicalcolatore alla struttura del robot. La verifica di stabilità del minicalcolatore è stata eseguita in relazione soprattutto a movimenti bruschi e inconsueti che possono potenzialmente minare la stabilità dell'intera struttura.

L'esecuzione di tutti i test sopra citati e la loro conclusione con esito positivo, consente di attuare in maniera permanente la sostituzione definitiva della vecchia libreria con la nuova versione Aria-2.7.0. In questo modo tale strumento può essere utilizzato in maniera completamente efficiente ed efficace per la scrittura di programmi sull' Acer Aspire One #3 e per la loro esecuzione (sia direttamente su Tobor che attraverso l'impiego del simulatore).

3. La soluzione adottata

In questo paragrafo, vengono indicate le strategie risolutive relative alle problematiche introdotte nel paragrafo precedente. Per ogni punto vengono delineate le procedure dettagliate e le problematiche riscontrate durante tali operazioni.

3.1. Installazione del software sul calcolatore Acer Aspire One

Di seguito vengono descritti i passi che sono stati compiuti per installare il software necessario. L'intento è stato quello di produrre una guida passo-passo all'installazione di tutto il software necessario ed in particolare di Aria nella sua versione più recente (2.7.0).

Durante le varie fasi d'installazione, è opportuno che l'utente acquisisca i privilegi di root, ovvero privilegi di più alto livello. Questo viene riportato nei comandi con il simbolo '#'.

Per quanto riguarda l'installazione degli strumenti di compilazione, è stato utilizzato il sistema di gestione dei pacchetti .*rpm. L'installazione di gcc ha richiesto l'esecuzione del seguente comando da shell:

```
# yum install gcc-c++
```

A seguito di tale comando è stato installato il compilatore GNU C++ (versione 4.3) che a sua volta ha richiesto obbligatoriamente l'installazione di altri elementi:

- libreria standard `libstdc++-4.3`
- compilatore GNU C (versione 4.3)
- libreria standard `libc-4.3`

Tali dipendenze sono state risolte automaticamente dal sistema di gestione dei pacchetti, il quale ha prelevato il software installato direttamente dai repository standard della distribuzione Linux Linpus. Successivamente, si è passati all'installazione del programma "make" mediante il seguente comando:

```
# yum install make
```

L'installazione di `make` permette la compilazione automatica di file sorgente, attraverso la chiamata di compilatori opportuni e la risoluzione delle dipendenze tra i diversi file sorgente. Tale programma è particolarmente utile per compilare programmi costituiti da più file. Anche in questo caso il programma è stato prelevato direttamente dal repository standard della distribuzione Linux Linpus.

Da questo momento in poi, il software installato è stato scaricato direttamente dai siti dei produttori (Mobile Robots e Acer) poiché nel repository della distribuzione Linux Linpus non era presente. I file scaricati sono stati inseriti in una cartella appositamente creata e raggiungibile tramite il seguente percorso "/home/user/installed_sw_robotica" sul calcolatore Acer Aspire One #3.

Prima di procedere all'installazione della libreria Aria è stato opportuno aggiungere al sistema un nuovo gruppo ed un nuovo utente appartenente al nuovo gruppo, entrambi di nome 'reed' mediante i seguenti comandi:

```
# groupadd reed

# useradd –g reed reed
```

La loro aggiunta non è stata obbligatoria. Tuttavia, si è preferito creare tale gruppo e aggiungervi un nuovo utente dal momento che il sistema di gestione della libreria stampava a video una consistente quantità di warning in relazione alla mancanza di tali elementi.

La nuova libreria (Aria 2.7.0) è stata installata nella cartella "/usr/local/Aria", lanciando i seguenti comandi:

```
# cd /home/user/installed_sw_robotica

# rpm -i ARIA-2.7.0.i386.rpm
```

Il pacchetto rpm installato fornisce sia la versione binaria della libreria compilata con gcc-c++-3.4, sia i file sorgente nel caso si desideri effettuare una ricompilazione. Dal momento che gcc-c++-4.3 è il compilatore con cui l'Acer Aspire One viene equipaggiato, è stata effettuata la ricompilazione della libreria con i seguenti comandi:

```
# cd /usr/local/Aria/

# make clean

# make
```

La ricompilazione della libreria è servita a verificare che il compilatore precedentemente installato funzionasse correttamente. Come risaputo, una versione più aggiornata di un compilatore fornisce maggiori garanzie per quanto riguarda la sicurezza e l'affidabilità.
In modo analogo spostandosi nella sottodirectory examples/ si è proceduto alla compilazione dei programmi di esempio scritti utilizzando la libreria in questione:

```
# cd /usr/local/Aria/examples/

# make clean

# make
```

Il Makefile presente nella directory examples/ è in grado di compilare qualsiasi file *.cpp che faccia uso della libreria Aria purché esso risieda nella stessa directory.

In seguito, è stato installato il simulatore MobileSim-0.5.0 attraverso i seguenti comandi:

```
# cd /home/user/installed_sw_robotica

# rpm -i MobileSim-0.5.0-0.i386.rpm
```

Infine, è stato installato il driver per il convertitore USB-Seriale necessario per utilizzare l'apposito cavo che collega il minicalcolatore al robot.

Dal punto di vista logico il processo di installazione è stato il seguente:

- dal sito del produttore sono stati scaricati i sorgenti del kernel Linux utilizzato dall'Acer Aspire One.
- si è proceduto alla riconfigurazione del kernel, in particolare è stato aggiunto il modulo per il supporto al cavo adattatore USB-Seriale.
- sono stati ricompilati tutti i moduli .
- sono stati copiati i moduli di interesse per il dispositivo in questione nelle apposite directory di sistema.

Di seguito viene riportata la sequenza dei comandi che realizza l'installazione del driver:

```
# yum install ncurses-devel

# cd /home/user/installed_sw_robotica
# unzip linux-2.6.23.9lw

# cd linux-2.6.23.9lw

# cp config_lawson_080516v1 .config

#make menuconfig
```

Modificare la voce Device Drivers -> USB Support ->USB SerialConverter support -> USB Prolific 2303 Single Port Serial Driver

Uscire da menuconfig e salvare la nuova configurazione del kernel

```
# rm -rf include/asm

# ln -sv include/asm-i386 include/asm

# make modules

# cp drivers/usb/serial/usbserial.ko
/lib/modules/2.26.23.9lw/kernel/drivers/usb/serial

 # cp drivers/usb/serial/pl2303.ko
/lib/modules/2.26.23.9lw/kernel/drivers/usb/serial

# depmod –ae
```

I moduli ora vengono automaticamente utilizzati dal sistema operativo nel momento in cui si inserisce l'adattatore nella presa USB del minicalcolatore Acer Aspire One. A seguito dell' inserimento del cavo nella porta USB del calcolatore è possibile verificare l'effettivo funzionamento del driver appena installato eseguendo il seguente comando shell:

```
# dmesg
```

Tale comando stampa a video i messaggi lanciati dal kernel relativamente al riconoscimento della periferica inserita. Una volta riconosciuto l'adattatore e caricato il driver appena installato, viene creato il file di dispositivo `/dev/ttyUSB0`. Attraverso il comando:

```
# ls –l /dev/ttyUSB0
```

si può notare come il dispositivo, sia di proprietà dell'utente `root` ed appartenga al gruppo `uucp`.
Per permettere all'utente di default `user` di utilizzare il driver in questione è stato necessario aggiungerlo al stesso gruppo `uucp` cui appartiene il dispositivo. Per fare ciò è stato sufficiente editare il file `/etc/groups` (servono il privilegi di `root`), salvare le modifiche e riavviare il sistema affinché le modifiche avessero effetto.

3.2. Riadattamento configurazione Tobor

In questo paragrafo verranno discusse le operazioni che fisicamente sono state compiute su Tobor per adattarlo ai nostri scopi.

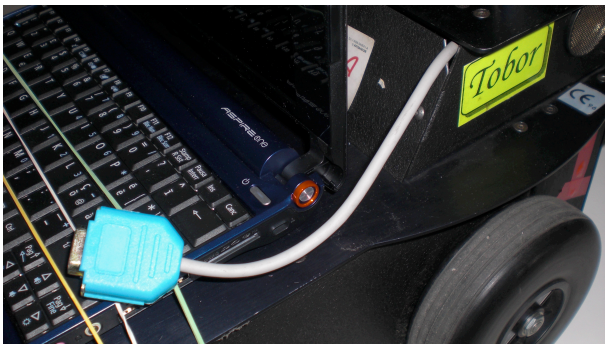
3.2.1. Riadattamento della struttura di Tobor

La "coppa trofeo" montata sulla parte superiore della struttura era fissata con del nastro adesivo ed è stato relativamente semplice rimuoverla. La rimozione ha liberato la superficie superiore del robot che è apparsa perfettamente adatta ad ospitare l'Acer Aspire One. Tuttavia, prima di poter posizionare il PC su di essa, è stato necessario pulire il robot dai residui di nastro adesivo. Tale pulizia è stata più difficoltosa della rimozione della coppa, ed è stata eseguita raschiando manualmente la superficie metallica con uno straccio imbevuto di alcool isopropilico.

3.2.2. Riadattamento della modalità di comunicazione con il calcolatore

Il radio modem presente sul robot era collegato direttamente al micro controller ed era causa di problemi di natura elettrica, che rendevano inutilizzabile la porta seriale che il robot mette a diposizione come ulteriore canale di input. Dovendo montare il minicalcolatore su Tobor e volendo utilizzare la porta in questione è stato necessario risolvere il conflitto. La Console del robot è stata aperta e il connettore femmina che collegava il radio modem alla presa seriale maschio situata all'interno della Console del robot è stato scollegato e portato all'esterno della cavità.

E' stato valutato opportunamente dove far passare tale cavo, dal momento che nella progettazione originale del robot Pioneer non è stata prevista una modifica di questo tipo. Con un'apposita trancia per ferro è stato operato un taglio nell'angolo superiore destro del pannello posteriore della Console (come visibile nella figura riportata in basso). In questo modo è stato possibile portare all'esterno il cavo del radio modem e richiudere la struttura attraverso le apposite viti.



Una volta fatta questa operazione ci si è resi conto che il connettore portato all'esterno era femmina; dal momento che tale connettore doveva essere inserito all'interno della presa seriale posta sul pannello superiore della Console di Tobor, è stato necessario realizzare un adattatore maschio-maschio, poiché tale presa seriale era femmina anch'essa.

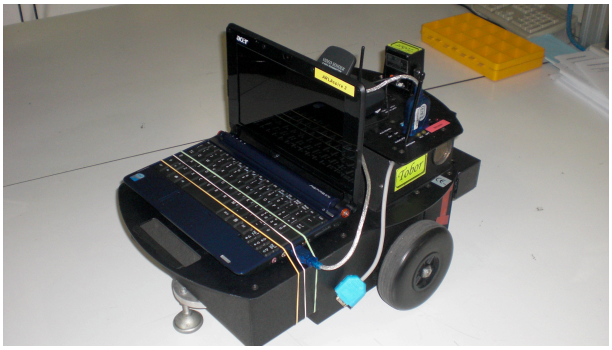
A questo punto il robot può essere gestito attraverso due possibili configurazioni:

- Tramite radio modem: il dispositivo radio situato all'interno della Console comunica con un altro dispositivo radio posizionato all'interno del laboratorio. Per fare in modo che si utilizzi questa modalità è sufficiente inserire il cavo seriale (provvisto di adattatore maschio) all'interno della presa seriale femmina situata sulla parte superiore di Tobor.
- Tramite calcolatore esterno: il collegamento tra calcolatore e robot viene realizzato attraverso un cavo USB-seriale. Un'estremità di tale cavo deve essere inserita nella porta USB del minicalcolatore mentre l'altra estremità (maschio) deve essere inserita all'interno della presa seriale femmina situata sulla parte superiore del robot.

Questo lavoro ha permesso di passare agevolmente da una configurazione all'altra semplicemente eseguendo un' operazione di plug/unplug di un connettore seriale.

3.3. Installazione on-board bel calcolatore Acer Aspire One su Tobor

Per sfruttare il collegamento seriale tra il robot ed il calcolatore è stato necessario fissare l'Acer Aspire One sulla superficie superiore del robot, opportunamente liberata come descritto in precedenza. Grazie alle ridotte dimensioni del calcolatore in oggetto la superficie si è rivelata sufficientemente adatta ed è bastato semplicemente appoggiare il calcolatore sopra di essa. L'attrito tra i piedini in gomma del calcolatore, e la superficie metallica di Tobor era già di per sé sufficiente a tenere fermo il calcolatore anche con il robot in movimento, tuttavia per escludere possibili cadute si è provveduto ad utilizzare degli elastici che di fatto hanno legato insieme i due componenti. Su suggerimento del docente sarebbe più opportuno fissare il calcolatore al robot attraverso delle apposite squadrette. La figura seguente illustra la configurazione da noi realizzata,



3.4. Cambiamenti introdotti nella libreria ARIA (2.7.0)

In questa sezione abbiamo deciso di elencare gli aggiornamenti principali apportati nella libreria Aria 2.7.0 e 2.6.0 rispetto alla versione adottata in precedenza nel laboratorio, cioè la 2.5.2 .

Dal momento che la macchina montata sul robot esegue in ambiente Linux programmi scritti in C++ abbiamo tralasciato di elencare anche gli aggiornamenti riguardanti Java e il sistema operativo Windows, che possono comunque essere trovati, insieme ai bugfix,sul documento originale dei cambiamenti (in inglese) inserito in Appendice.

3.4.1. Aggiornamenti della versione ARIA 2.7.0

In questa versione, rilasciata il 30 aprile 2009, si consiglia di aggiornare MobileEyes alla versione 2.2.4 o superiore per evitare problemi con le operazioni a distanza del robot. Nella versione 2.7.0 di Aria rispetto alla versione 2.6.0 sono state apportati i seguenti aggiornamenti, che elenchiamo suddivisi in:

- Aggiornamenti di carattere generale.
- Aggiornamenti specifici per la classe dei dispositivi di misurazione laser.

3.4.2. Aggiornamenti generali

Gli aggiornamenti principali sono:

- Nuove classi ArRobotConnector ed ArLaserConnector sostituiscono la vecchia ArSimpleConnector che, per mantenere la compatibilità con il vecchio codice, resta comunque disponibile come classe legacy che utilizza le nuove classi.

  - ArRobotConnector ha tutte le funzionalità precedenti ma può fare il parsing automatico degli argomenti quando si chiama blockingConnect.

  - ArLaserConnector crea i laser, li aggiunge al robot e li configura con gli argomenti passati alla riga di comando (estraendo comunque le informazioni sulla connessione dei dispositivi dal file .p) . I laser si configureranno da soli con le informazioni nel file .p se queste non vengono impostate da linea di comando.

- Sono state aggiunte e modificate classi di supporto per la camera RVision PTZ, per movimenti laterali su un robot Seekur e per altre periferiche, in particolare:

  - la classe ArUrg fornisce supporto per i laser Hokuyu URG (nello specifico per URG-04LX e per il protocollo SCIP 1.1).

  - la classe ArGPS ora legge e provvede alcuni dati aggiuntivi dal GPS (se il ricevitore li fornisce).

- Supporto migliore per dispositivi multipli eterogenei di misurazione laser delle distanza in ArLaserConnector, ArRobot e in generale in ARIA. E' possibile configurare quali dispositivi sono disponibili su un robot, e le opzioni dei dispositivi, nel file dei parametri del robot, nel file Aria.args o tramite le opzioni della riga di comando del programma.

- Nella classe Aria è stata aggiunto il metodo Aria::remExitCallback che verifica di non terminare prima di aver rimosso una callback, dal momento che ciò interromperebbe il ciclo di comunicazione tra calcolatore e robot. Il metodo Aria::addExitCallback, analogamente verifica di non terminare prima di aggiungere una callback.

- Nella classe ArUtil è stata aggiunta la chiamata di funzione setFileCloseOnExec(). Questa funzione fa in modo che una chiamata a fork() o exec() chiuda un file o un socket.

- Sempre la classe ArUtil ora dispone di funzioni fopen, open, creat, e popen che chiameranno la nuova funzione setFileCloseOnExec() per il nuovo file o socket. Si raccomanda che queste funzioni vengano sempre usate per evitare perdite di risorse se un programma utilizza exec() o fork().

- Modificata la classe ArKeyHandler così che possa ricevere un puntatore FILE * invece di usare sempre stdin; ora ha l'opzione di non ricevere i tasti nel costruttore (ciò si ripercuoterà solamente su piattaforme non Windows).

- Nella classe ArNetworking ora ci sono semplici comandi server per microcontrollerCommand e microcontrollerMotionCommand; microcontrollerMotionCommand sospenderà la normale riflessione dello stato in modo che possano essere inviati anche solamente comandi di movimento diretto (come la modalità 'd' nell'esempio demo).

- In ArDPPTU è stato ggiunto il supporto per il più recente DP PTU-D47. Selezionare il D47 fornendo PANTILT_PTUD47 come secondo argomento al costruttore di ArDPPTU.

3.4.3. Aggiornamenti per le classi dei dispositivi di misurazione laser

Gli aggiornamenti per il Laser range device sono i seguenti:

- Reso il supporto per i laser molto più generico. Ci sono classi di compatibilità che permettono di compilare il vecchio codice, l'unica eccezione sono i parametri 'second laser' nel file .p che non verranno più letti perciò dovranni essere cambiati in 'Laser 2 settings' .

- Inclusa una nuova classe ArLaser che svolge quasi tutto ciò che era svolto da ArSick in termini di filtraggio e callBacks, anche se i nomi e gli argomenti dei metodi a volte variano leggermente. ArLaser è un superset di tutto ciò che si può impostare su qualunque laser così che possa essere configurato genericamente.

- La classe ArSick è stata sostituita con la classe ArLMS2xx. ArSick è ora una classe legacy le cui funzioni sono ancora valide per mantenere la compatibilità del vecchio codice.

- Sistemate le informazioni sui parametri e tipo di laser in modo che siano immagazzinate nel file .p e che possano essere fornite in riga di comando.

- Fatto in modo che il robot debba essere collegato (con ArRobotConnector) prima della chiamata Aria::parseArgs. Il codice legacy funziona senza questo perché utilizza solamente un tipo di laser.

- Nella classe ArUtil sono state aggiunte le seguenti chiamate, principalmente per essere utilizzate dalla nuova ArLaserConnector:

  - createLaser che crea un Laser del tipo fornito.

  - getCreateLaserTypes che elenca i tipi che possono essere creati.

  - createDeviceConnection che crea una connessione ad un dispositivo (device connection) del tipo fornito e prova a collegarlo alla porta fornita.

  - getCreateDeviceConnectionTypes che elenca i tipi di connessioni ai dispositivi che possono essere creati.

- Il massimo raggio di azione del laser e la dimensione cumulativa del buffer possono essere impostati nel *.p file*, la dimensione cumulativa del buffer è impostata a 200 per default.

- Ai laser non viene fornita la dimensione attuale del buffer, viene impostata a qualunque sia il massimo numero di letture rilevate quando il laser è connesso (basandosi sui gradi e sull'incremento). In questi casi ArSick rimane uguale a prima.

- Modificata ArModeLaser (la modalità laser in demo) affinché operi con laser multipli.

- I laser vengono simulati in modo totalmente diverso da prima (a meno che non stiate usando la classe legacy ArSick dove il comportamento è rimasto invariato). Ora c'è una nuova classe chiamata ArSimulatedLaser che prende il posto della normale classe laser di quel tipo e può essere configurata nello stesso modo, ma prende le letture dal simulatore. Ciò viene gestito in maniera trasparente da ArLaserConnector. Al momento funziona solo con un laser, ma in futuro ci saranno modifiche a MobileSim e ad Aria per permettergli di lavorare con laser multipli.

- Il parametro LaserIgnore nel file *.p* e gli argomenti corrispondenti della riga di commando (-laserAdditionalIgnoreReadings) ora possono ricevere dei range (inizio-fine), gli argomenti multipli possono essere separati da virgole (con o senza spazi extra) o possono essere separati solamente dagli spazi come nella vecchia versione (sebbene non funzioni da riga di comando).

- Modificata ArRobot in modo che mappi i laser con dei numeri così che le classi possano estrarli più facilmente. Ora i laser vengono automaticamente aggiunti al robot come dispositivi a distanza da ArLaserConnector. Le chiamate per fare ciò sono ArRobot::addlaser, ArRobot::remLaser, ArRobot::findLaser e ArRobot::getLaserMap.

3.4.4. Aggiornamenti della versione ARIA 2.6.0

Nella versione 2.6.0 della libreria (non rilasciata al pubblico), datata 3 agosto 2007, sono state apportate le seguenti modifiche rispetto alla versione 2.5.2:

- La classe ArSimpleConnector ora ha argomenti per fare il log di: pacchetti ricevuti e spediti, movimento ricevuto e spedito, velocità ricevuta e azioni.

- Realizzati i timeouts di ArNetworking per To come per From, che servono per prevenire un uso estremo della memoria.

- ArRobot: aggiunti "odometer" e "trip odometer" alla classe ArRobot. Servono per tener traccia della distanza totale percorsa e delle svolte lungo il percorso durante l'esecuzione del programma. "Trip odometer" può essere resettato a differenti eventi.

- ArRobot: il nuovo firmware fornisce un valore di ritardo cinematico (perché usando il giroscopio può accadere che il microcontrollore spedisca dati più vecchi ma un' odometria più corretta). ArRobot ora tiene conto di questo ritardo quando inserisce le informazioni per l'interpolazione della posizione. Questo cambiamento dovrebbe essere trasparente a tutti, ma ci potrebbero essere più proiezioni anticipate nei dati restituiti da ArRobot::getPoseInterpPosition() e da ArRobot::getEncoderPoseInterpPosition(). ArSick tiene già conto di ciò. La proiezione anticipata su ArInterpolation è anche più indulgente delle proiezioni future (ammette una proiezione del 50% invece che del 3%)

- ArRobot, Actions, ArNetworking: aggiunto supporto per la velocità laterale di Seekur (movimento sul fianco) nei server per le operazioni a distanza di ArRobot, Actions e ArNetworking.

- ArActionDeceleratingLimiter: cambiamenti al costruttore così che funzioni con il movimento laterale, ora richiede un tipo enum invece di un booleano, il vecchio true diventa ArActionDeceleratingLimiter::FORWARDS ed il vecchio false diventa ArActionDeceleratingLimiter::BACKWARDS.

- ArTCM2: aggiunto il supporto per una bussola TCM2 connessa direttamente al computer. La classe ArTCM2 è ora un'interfaccia per entrambi i tipi di compasso. Si suggerisce di usare un ArCompassConnector per creare un oggetto bussola basato sugli argomenti forniti da riga di comando (il default è ArTCM2Robot), e chiamare blockingConnect() per connettere la bussola (vedere demo.cpp per un esempio). Chiunque dovrebbe utilizzare ArTCM2Robot (tipo "robot").
- ArGPS: aggiunto il supporto per il Trimble AgGPS e migliorata l'interfaccia della classe.
- Informazioni riguardanti l'altezza del laser ed un possibile secondo laser possono essere fornite nei parametri del robot (in genere non vengono usate per la maggior parte dei robot).

3.5. Testing

A causa della natura "sistemistica" dell'elaborato in questione, rivolta principalmente all'installazione di componenti hardware e software nonché alla verifica del relativo funzionamento, il testing in oggetto è inteso sia nell'ottica con cui è stato introdotto nel sottoparagrafo 2.4, che come metodologia con cui è stato approcciato il lavoro. Infatti, dal momento che non esiste una rigida sequenza logica alla quale attenersi nello sviluppo dei vari sottoproblemi, si è proceduto "per tentativi" e successivamente sono stati testati i risultati ottenuti.

Per quanto riguarda l'installazione del software, il problema principale si è verificato nel tentativo di installare il simulatore a partire dal pacchetto in formato sorgente. Eseguendo il comando di compilazione `make` infatti, si sono manifestati errori relativamente a librerie grafiche necessarie che non risultavano essere presenti. Nel repository della distribuzione Linpus i pacchetti non erano disponibili e pertanto si è proceduto all'installazione manuale dei singoli pacchetti .*rpm dopo averli scaricati dal sito rpmfind.net. Inoltre, la versione specifica per la distribuzione in uso non era disponibile, pertanto sono state utilizzate le versioni per Fedora (da cui Linpus deriva). Terminata l'installazione di tali librerie, è stato rieseguito il comando `make` con esito positivo ed il simulatore è stato installato e utilizzato correttamente. Tuttavia, al successivo avvio la macchina presentava problemi; in particolare, veniva eseguito il boot del sistema ma non era possibile caricare l'ambiente grafico che rimaneva bloccato presentando all'utente una schermata di colore nero. A seguito di tale problema, è stata necessario eseguire il ripristino del sistema operativo attraverso il software fornito con il minicalcolatore. Molto probabilmente le cause di questo problema sono da imputare a sovrascritture (a seguito delle installazioni effettuate) di alcuni file fondamentali per il boot della macchina ed in particolare per il funzionamento dell'ambiente grafico.
Una volta eseguito il ripristino della macchina si è deciso di reinstallare il simulatore utilizzando il pacchetto in formato .rpm. Con questa procedura alternativa, non si sono verificati problemi ed il simulatore, testato con i programmi `demo` e `wander`, presenti nella cartella degli esempi della libreria Aria, è risultato funzionare correttamente.

Durante l'installazione del driver per il cavo adattatore USB-Seriale, il comando `make modules` non compilava i moduli, a meno di un'acquisizione di privilegi meggiori (privilegi di root); infatti, il compilatore segnalava a video di non essere in grado di reperire determinate librerie nonostante esse fossero installate. Probabilmente, il problema si sarebbe potuto risolvere in modo alternativo, aggiungendo i percorsi assoluti di tali librerie alla variabile d'ambiente `PATH` dell'utente `user`.

La nuova modalità di comunicazione robot-minicalcolatore tramite porta seriale era inutilizzabile a causa di conflitti di natura elettrica con il radio modem presente all'interno di Tobor. Infatti, anche con il modem del robot spento, il cavo seriale correttamente inserito, ed il programma utente in esecuzione, il robot rimaneva costantemente inerte come se non ricevesse alcun segnale, e contemporaneamente, l'output del programma stampava un messaggio d'errore a video che specificava il non rilevamento del robot.

L'ipotesi di conflitto è stata formulata dopo aver collegato, tramite l'adattatore USB-Seriale, l'Acer Aspire One con il radio modem presente sul tavolo del laboratorio ed aver verificato che il programma `wander` veniva correttamente eseguito da Tobor. Questo indicava chiaramente che il ponte radio modem-modem era funzionante così come l'adattatore utilizzato e la libreria Aria installata. Pertanto rimaneva, come unica possibile causa, il conflitto di natura elettrica.
Apportando le modifiche descritte nel sottoparagrafo 3.2.2 il problema è stato correttamente risolto.

Nella fase finale, si è proceduto a testare effettivamente il funzionamento della libreria sul robot reale, sempre tramite l'esecuzione dei due programmi d'esempio `demo` e `wander`. A seguito di tali test, osservando il comportamento di Tobor, è stato possibile stabilire come tutto funzionasse per il meglio. Tuttavia, l'esecuzione dei suddetti programmi senza disporre dei privilegi dell'utente root sembrava non produrre alcuna reazione sul robot. Ancora una volta è stato possibile dedurre correttamente la causa del problema – i permessi – grazie all'impiego del simulatore. Infatti, i programmi che l'utente `user` eseguiva con il simulatore, venivano correttamente simulati, mentre quando si passava attraverso la connessione con il robot reale, esso rimaneva inerte, cosa che non accadeva se lo stesso programma veniva eseguito con i privilegi di root. In uno scenario del genere è stato facile intuire che l'utente normale non avesse i privilegi sufficienti per l'utilizzo dell'adattatore.

4. Modalità operative

In questo paragrafo vengono riportati i passi necessari per verificare il corretto funzionamento del sistema sia con il simulatore che con il robot reale. Di seguito si ipotizza che sia l'utente root ad eseguire i comandi.

4.1. Esempio di utilizzo del simulatore

Per l'avvio del simulatore è sufficiente eseguire il seguente comando da shell:

```
# MobileSim
```

A seguito di tale comando appare una finestra all'interno della quale è possibile selezionare il tipo di robot che si desidera simulare ed inoltre è possibile selezionare l'ambiente in cui il robot virtuale deve muoversi, caricando la relativa mappa. Cliccando su OK il simulatore si mette in ascolto sulla porta TCP 8101. Per lanciare il programma d'esempio è necessario spostarsi nella cartella in cui esso è contenuto ed eseguirlo. Segue un esempio con riferimento al programma `wander`:

```
# cd /usr/local/Aria
# ./wander
```

Come ogni programma scritto utilizzando le funzionalità della libreria Aria, anche il programma in questione, effettuerà un primo tentativo di connessione al simulatore e se quest'ultimo non è presente cercherà successivamente di connettersi con il robot reale. Nell'esempio in questione, se il simulatore è stato attivato correttamente, nella relativa schermata sarà possibile visualizzare i movimenti del robot in funzione del programma lanciato e della mappa caricata.

4.2. Esempio di utilizzo di Tobor

Per effettuare un test con il robot reale è opportuno:

- Fissare saldamente il calcolatore al robot tramite gli appositi elastici, come descritto nella sezione 3.3
- Spegnere qualsiasi eventuale istanza del simulatore in funzione
- Inserire il cavo adattatore USB-Seriale nella presa USB del calcolatore
- Inserire l'altra estremità del cavo nella porta seriale del robot e verificare che la spia di collegamento si accenda
- Verificare che il robot sia acceso ed i motori siano abilitati
- Eseguire i seguenti comandi da shell:

```
# cd /usr/local/Aria
# ./wander -rp /dev/ttyUSB0
```

Il programma in questione, tramite la libreria Aria, non trovando alcun simulatore in ascolto tenterà di connettersi al robot reale tramite il dispositivo `ttyUSB0` (l'adattatore) come specificato nel comando attraverso l'opzione –rp.

4.3. Componenti necessari

4.3.1. Il calcolatore Acer Aspire One A110L

Il calcolatore Acer Aspire One è un Netbook (dispositivo portatile di dimensioni più piccole rispetto ad un Notebook e di costo contenuto) dotato di sistema operativo Linpus Linux, una versione "lite" del sistema operativo Fedora. La scelta di un calcolatore di questo tipo è giustificata da alcuni aspetti molto importanti:

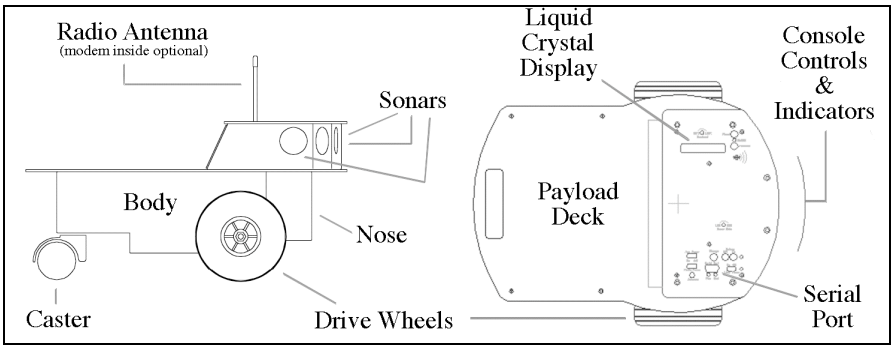
- Riduzione dei costi: il calcolatore ha un costo estremamente ridotto in relazione alle prestazioni che è in grado di fornire. Il Netbook, essendo un dispositivo di ultima generazione, possiede una potenza di calcolo di gran lunga superiore a quella richiesta per eseguire i programmi del robot.
- Dimensioni contenute: le dimensioni estremamente ridotte del calcolatore gli consentono di collocarsi perfettamente sulla schiena del robot Tobor realizzando un assemblaggio perfetto.
- Elevata autonomia energetica: il calcolatore è un dispositivo dotato di una batteria in grado di fornire molte ore di autonomia energetica; in questo modo il robot può eseguire le proprie attività per una notevole quantità di tempo prima di dover rientrare nella stazione di ricarica.

Di seguito vengono riportate le principali caratteristiche tecniche del minicalcolatore.

- Processore Intel Atom N270 (1.60 GHz, 533 Mhz FSB, 512 KB L2 cache)
- 512 MB di RAM (DDR2 533 MHz SDRAM)
- 8 GB di unità SSD
- Schermo 8.9" (TFT LCD, risoluzione 1024x600 – 262,000 colori supportati)
- Webcam integrata Acer Crystal Eye, 0.3 megapixel
- WLAN: Acer InviLink 802.11 b/g Wi-Fi
- LAN: 10/100 Mbps Fast Ethernet
- Sistema operativo Linpus Linux

4.3.2. Il robot Tobor

Tobor è un robot Pioneer la cui struttura è visualizzata nella figura sottostante.



I componenti principali che costituiscono il robot sono:

- Console: situata sulla schiena del robot e posizionata nella parte anteriore, è il "cervello" del robot e al suo interno sono alloggiati il microcontrollore ed il radio modem.
- Deck: le dimensioni di questa zona sono state pensate per ospitare un calcolatore, ed assicurarne la sua protezione.
- Body: zona inferiore adibita a contenere la batteria e i motori per il movimento delle ruote.
- Nose: zona inferiore all'interno della quale è situato l'experimenter module. Al suo interno possono essere alloggiati accessori e sensori aggiuntivi.

Il sistema di guida del robot è gestito da motori reversibili in corrente continua. Ogni motore include un encoder ad alta risoluzione per garantire estrema precisione.

Il robot è provvisto di 7 sonar, 5 situati nella parte anteriore e 2 laterali (uno per lato) la cui posizione è fissa sulla Console. I 5 sensori anteriori sono in grado di rilevare ostacoli lungo un angolo di 75°. I due sensori laterali, invece, sono utili per rilevare la presenza di le pareti laterali (per esempio quelle dei corridoi).

La porta seriale del robot Pioneer è compatibile con lo standard RS232 e consente la comunicazione dati tra il microcontrollore del robot ed un calcolatore esterno. Sotto la porta seriale sono situati due led il cui lampeggiamento indica rispettivamente la trasmissione/ricezione dei dati tra il robot e il calcolatore.

E' presente poi un interruttore che consente di abilitare/disabilitare il radio modem alloggiato all'interno della Console. Tale interruttore non interferisce con le funzioni della porta seriale, ma è consigliabile spegnere il radio modem se si intende utilizzare la porta seriale per connettervi un calcolatore esterno.

5. Conclusioni e sviluppi futuri

Grazie al lavoro svolto è possibile disporre della nuova versione della libreria Aria-2.7.0 sul robot Tobor e del simulatore MobileSim-0.5.0. Inoltre la comunicazione tra il robot ed il minicalcolatore avviene attraverso una linea seriale.
Gli sviluppi futuri che si possono ipotizzare in relazione al lavoro svolto finora sono principalmente inerenti l'uso della libreria e circa le funzionalità aggiuntive offerte, specialmente per quanto riguarda il supporto fornito ai dispositivi ottici laser range.

5.1. Considerazione dei membri del gruppo

L'elaborato in questione ha permesso ai membri del gruppo di avere un approccio esauriente relativamente alle modalità di funzionamento basilari dei robot, alla sua architettura, che permette di raggiungere alti livelli di astrazione in un modo differente rispetto all'ambito puramente informatico incentrato sull'uso del calcolatore tradizionale, almeno dal punto di vista di studenti che non sono abituati ad operare su macchine come i robot.

Per quanto riguarda la parte di programmazione, la libreria Aria ha permesso di trattare il robot con estrema flessibilità e semplicità, garantendo allo stesso tempo la possibilità di sviluppare programmi relativamente complessi, specialmente per chi non è abituato a questo genere di programmazione.

Inoltre, il lavoro svolto ci ha permesso di valutare quanto le piccole difficoltà incontrate nei vari sottoproblemi incidano, specialmente in termini di tempo, anche quando vengono svolti compiti che potrebbero apparire semplici, ma che analizzati nel dettaglio si rivelano tanto più complessi quanto più è necessaria la collimazione dei vari risultati intermedi che portano all'obiettivo comune.

Bibliografia

[1] ActivMedia: "Pioneer Mobile Robot Operation Manual", Edition 2, January 1998.

[2] J. Borenstein, H.R. Everett, L. Feng: "Where am I? Sensors and methods for mobile robot positioning", April 1996.

Indice