Robot manipolatore per compiti di assemblaggio in ambiente dinamico

Unità di Ricerca *Università di Parma*
Dipartimento di Ingegneria dell'Informazione
Viale delle Scienze - 43100 Parma - Italy
e-mail responsabile UR: conte@ce.unipr.it

1 Requisiti

Obiettivo della attività di ricerca è lo studio e la realizzazione di sistemi robotici autonomi, in grado di operare anche in ambienti dinamici o non totalmente strutturati. A tale scopo un sistema robotico deve disporre di adeguata sensorialità e mobilità, per percepire i cambiamenti nell'ambiente e reagire ad essi, e di capacità di adattamento ed eventualmente di ragionamento, per intraprendere un appropriato insieme di azioni in risposta agli eventi.

Il sistema specifico descritto in questa scheda è rivolto in particolare a compiti di manipolazione. Il problema di riferimento consiste nella costruzione di un kit di oggetti su un vassoio destinazione. Gli oggetti arrivano con tempi, ordine, e disposizione arbitrari all'interno dell'area di lavoro del manipolatore. Un piccolo robot mobile, che non viene considerato come parte del sistema da controllare, emula le funzioni di un nastro trasportatore. Il comportamento del robot mobile non è tuttavia noto al manipolatore, in particolare per ciò che riguarda la durata della sosta nello spazio di lavoro.

Il manipolatore provvede a scaricare gli oggetti dal robot mobile nel più breve tempo possibile e ad elaborare il piano per la costruzione del kit e il trasporto degli oggetti. Il robot mobile, infatti, dopo un breve lasso di tempo lascia la cella di lavoro del manipolatore per ritornare nell'area di rifornimento pezzi, da cui poi si porterà nuovamente verso il manipolatore per essere scaricato. Il deposito nel vassoio destinazione delle diverse parti fornite al manipolatore comporta la determinazione di un piano di lavoro e di un percorso che consenta al manipolatore stesso di trasportare gli oggetti dal punto di scarico al vassoio evitando tutti gli ostacoli presenti nell'ambiente.

2 Architettura di realizzazione

Il sistema robotico è centrato attorno ad un manipolatore industriale Puma 560 a 6 gradi di libertà. Il manipolatore è dotato, oltre che di sensorialità interna, di una telecamera in posizione fissa, di un sensore a ultrasuoni montato sul polso e di fotocellule.

Il Puma560 è connesso al suo controllore industriale Mark II che tuttavia ospita unicamente gli anelli di controllo assi e l'I/O relativo all'attuazione e alla sensorialità propriocettiva. Infatti il controllore industriale originario è interfacciato mediante linee parallele e seriale ad un sistema di controllo esterno "aperto" che funge da controllore principale. Il controllore principale è costituito da un sistema VMEbus con CPU MicroSPARC (Force 3CE) che opera in ambiente Solaris 2.6 ed ospita il pacchetto di controllo RCCL (Robot Control C Library), ben noto in ambito accademico. La sensorialità eterocettiva è affidata a controllori dedicati (schede a microprocessore e sistema di visione), che forniscono le informazioni al sistema di controllo principale mediante linee seriali.

RCCL fornisce sofisticati generatori di traiettorie che in assenza di ostacoli consentono di attuare moti nello spazio dei giunti o nello spazio cartesiano a partire dalle configurazioni iniziale e di arrivo. Esso supporta inoltre numerose procedure di controllo del moto del manipolatore che costituiscono i blocchi elementari per la costruzione di compiti complessi. RCCL è una libreria multi-piattaforma inizialmente progettata in ambiente Unix BSD e successivamente adattata ad altri sistemi operativi (Solaris, VxWorks, SGI-Irix, Linux).

L'architettura di controllo task-level del sistema è totalmente realizzata in ambiente Solaris 2.6. La gestione delle informazioni sensoriali, la definizione del moduli di controllo di alto livello relativi a movimenti parametrici o pianificati del manipolatore, la sincronizzazione degli stessi, sono realizzati mediate i supporti multithreading e real-time offerti dal sistema operativo impiegato.

3 Problemi principali nello sviluppo

Lo sviluppo di architetture logiche per il controllo di sistemi robotici autonomi in ambienti dinamici richiede di affrontare molteplici problemi. I principali tra essi sono i seguenti:

- la progettazione e lo sviluppo di sistemi concorrenti caratterizzati da attività sia indipendenti che interagenti;
- il rispetto di vincoli temporali sia per le attività di tipo periodico, il cui peso è preponderante nel controllo dei manipolatori, sia per le attività aperiodiche caratterizzate da statistiche di arrivo;
- la limitatezza delle risorse computazionali rispetto alle esigenze complessive di elaborazione; tale limitatezza suggerisce l'adozione di meccanismi a tempo di esecuzione in grado di allocare le risorse ai processi in maniera variabile nelle diverse fasi del compito;
- l'estendibilità del sistema di controllo, tipicamente per accrescere la dotazione sensoriale o integrare moduli di ragionamento;
- la disponibilità di ambienti di sviluppo e debugging sul target che consentano di realizzare l'architettura logica di controllo definita e di analizzare l'esecuzione del compito in relazione ai vincoli sia logici che temporali.

3.1 Soluzioni parziali adottate

L'architettura logica utilizzata per l'organizzazione del controllo è di tipo ibrido, e cioè cerca di coniugare le esigenze di reattività del sistema robotico con l'esecuzione di attività deliberative.

Per assicurare estendibilità e agevole integrazione di moduli standard di ragionamento e pianificazione il controllore principale del sistema robotico si basa sul sistema operativo general purpose Solaris. Oltre ai supporti allo sviluppo del software, Solaris offre alcune importanti funzionalità orientate alla elaborazione in tempo reale e alla programmazione multithreaded (kernel revocabile, ridotta latenza all'interrupt, lock della memoria, priority inheritance sui semafori binari e tre classi di scheduling distinte (user, sys e real-time)). Il livello di prestazioni conseguibili caratterizza il sistema come soft real-time e tuttavia appare adeguato al tipo di applicazione. RCCL-Solaris, allo stato attuale, sfrutta solo parzialmente i supporti per la programmazione real-time e multithread offerti da questo sistema operativo. È in corso presso la UR una parziale revisione della libreria RCCL orientata ad ottenere una maggiore integrazione dei supporti real-time e multithread di Solaris 2.6.

Un problema delle architetture di governo dei robot autonomi è l'adattamento della priorità e delle risorse computazionali attribuite ai processi alle esigenze del compito e agli stimoli sensoriali. È allo studio presso la UR una politica di adattamento del carico computazionale indotto dalla parte reattiva dell'architettura robotica.

4 Pubblicazioni e riferimenti

Dei riferimenti che seguono, [1, 2, 3] descrivono il set up sperimentale e le architetture di governo sviluppate presso la UR. [4]-[8] sono rappresentativi dei problemi e delle evoluzioni delle architetture di governo dei robot autonomi. In [9, 10] viene proposto l'adattamento delle caratteristiche dei processi in relazione alle condizioni operative del compito osservate dal livello deliberativo dell'architettura. Il riferimento [11] illustra, tra l'altro, le prospettive di convergenza dei sistemi operativi general purpose e dedicati. I lavori [12, 13] presentano un esempio di sistema operativo dedicato ai sistemi robotici ma concepito secondo una filosofia soft real-time.

Relativamente alle problematiche di tempo reale sono in corso di stesura presso la UR due lavori, da sottoporre a sedi scientifiche. Il primo sarà dedicato alla illustrazione di una libreria di ausilio alla progettazione dei task reattivi e con esigenze di tempo reale di un'architettura di controllo ibrida. Il secondo descriverà un meccanismo di adattamento del carico basato sul rilassamento dei vincoli di attivazione per i task soft real-time. Si prevede che questi due lavori potranno essere resi disponibili in bozza alle altre UR del progetto entro il prossimo mese.

References

[1] F. Zanichelli, S. Caselli, A. Natali, A. Omicini, "A Multi-Agent Framework and Programming Environment for Autonomous Robotics," IEEE Int. Conf. on Robotics and Automation (ICRA'94), San Diego, CA, Aprile 1994.

[2] F. Zanichelli, G. Beccari, A. Bigi, S. Caselli, "Experiences on Prototyping Robot Manipulation Tasks: Putting a Hybrid Multiagent Architecture at Work," Int. Conf. on Advanced Robotics (ICAR'95), Barcellona, Spagna, Settembre 1995.

[3] F. Zanichelli, "Programmazione di Sistemi Robotici Autonomi", Rivista di Informatica, Vol. XXVII, n. 1 gennaio-aprile 1997.

[4] R.G. Simmons, "Structured Control for Autonomous Robots", IEEE Trans. on Robotics and Automation, Vol. 10, No. 1, February 1994.

[5] D.M. Lyons and A.J. Hendriks, "Plannig for Reactive Robot Behavior", IEEE Int. Conf. on Robotics and Automation, ICRA'92, Nice, France, May 1992.

[6] R.P. Bonasso, R.J. Firby, E. Gat, D. Kortenkamp, D. Miller, M. Slack, "Experiences with an Architecture for Intelligent, Reactive Agents", J. of Experimental and Theoretical Artificial Intelligence, Vol. 9, No. 1, 1997.

[7] R.J. Firby, "An Investigation into Reactive Planning in Complex Domains", Nat. Conf. on Artificial Intelligence, AAAI'87, 1987.

[8] F.F. Ingrand, R. Chatila, R. Alami, F. Robert, "PRS: A High Level Supervision and Control Language for Autonomous Mobile Robots", IEEE Int. Conf. on Robotics and Automation, ICRA'96, Minneapolis, MN, April 1996.

[9] D.J. Musliner, E.H. Durfee, K.G. Shin, "CIRCA: A Cooperative Intelligent Real-Time Control Architecture", IEEE Trans. on Systems, Man, and Cybernetics, Vol. 23, No. 6, November/December 1993.

[10] M. Schoppers, "A Software Architecture for Hard Real-Time Execution of Automatically Synthesized Plans or Control Laws", Conf. on Intelligent Robotics in Field, Factory, Service, and Space (CIRFFSS'94), AIAA, March 1994.

[11] J.A. Stankovic, et al., "Strategic Directions in Real-Time and Embedded Systems", ACM Computing Surveys, Vol. 28, No. 4, December 1996

[12] D.B. Stewart, P.K. Khosla, "Mechanisms for Detecting and Handling Timing Errors", Communications of the ACM, Vol. 40, No. 1, January 1997.

[13] D.B. Stewart, D.E. Schmitz, P.K. Khosla, "The Chimera II Real-Time Operating System for Advanced Sensor-Based Control Applications," IEEE Trans. on Systems, Man, and Cybernetics, Vol. 22, No. 6, November/December 1992.