Architettura di controllo di un 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

Lo studio e la realizzazione di sistemi robotici autonomi, in grado di operare anche in ambienti dinamici o non totalmente strutturati è una delle attività di ricerca che vedono impegnata l'Unità Operativa. Sistema robotici autonomi di nuova concezione devono 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.

Nella presente scheda viene descritta l'architettura di controllo di un robot manipolatore per compiti di assemblaggio in ambiente dinamico. Vengono in particolare descritti i flussi di controllo dell'architettura, i flussi dei dati e le attività preposte al controllo task-level, Inoltre, sono elencati i parametri temporali (tempi di esecuzione e periodi) relativi alle trame di controllo (thread) che realizzano l'architettura robotica in esame.

1 Il sistema robotico

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).

2 Architettura di controllo

L'architettura di controllo task-level del manipolatore Puma560 è realizzata mediante i supporti multi-thread/real-time offerti da Solaris 2.6.

In Tabella 1 sono elencati gli agenti che realizzano tale architettura, le loro caratterisitiche temporali e i loro parametri di scheduling. In particolare, "Instances" indica il numero di agenti di un certo tipo, "Timing" specifica se l'attività assegnata all'agente è periodica o aperiodica, mentre il campo "RT-Priority" mostra la priorità real-time Solaris assegnata alla trama di controllo che realizza l'agente. RM indica che la priorità è assegnata mediante politica Rate Monotonic. WCET è il worst case execution time relativo alla trama di controllo che esegue l'agente, infine T specifica il range dei periodi ammissibili.

Agents	Instances	Timing	RT-Priority	WCET (ms)	T (ms)
Motion Planner	n	Aperiodic	-	39E+3	-
Motion Executor	2	Aperiodic	1	7	-
Supervisor	1	Periodic	RM	0.04	1000-100
Timing Monitor	1	Periodic	RM	0.2	400-60
Sensor (Sonar, IR)	1	Periodic	RM	0.3	200-30
Sensor (Camera)	1	Periodic	RM	0.8	200-30
Trajectory Control	1	Periodic	59	10	50-30

Tabella 1: Caratterisitiche temporali e parametri di scheduling degli agenti dell'architettura [**N.B. I parametri sono stati corretti in data 5-11-1998**].

Per il *Motion Planner* si è indicato tuttavia il tempo medio di esecuzione; la durata della pianificazione è peraltro soggetta ad una forte variabilità. Oltre alle attività indicate in Tabella, il sistema esegue drivers e moduli kernel i cui fattori di utilizzo sono difficilmente quantificabili ma significativi. Rispetto ai dati in [1] alcuni valori risultano modificati a causa delle evoluzioni della versione del sistema operativo, delle librerie di supporto e della applicazione stessa.

La comunicazione fra agenti avviene mediante i meccanismi a memoria condivisa tipici della programmazione multi-threading. La mutua esclusione nelle fasi di lettura e scrittura delle strutture dati condivise è garantita da una opportuna collezione di mutex con priority inheritance. I tempi di accesso alle strutture dati condivise (regioni critiche) sono brevi.

2.1 Agenti dell'architettura di controllo

Trajectory Control
Periodicamente calcola e trasmette al controllore Mark II il setpoint relativo al controllo assi del manipolatore attraverso un'interfaccia dedicata. Attua in tal modo le richieste di moto elementari inoltrate da altri agenti presenti nell'architettura e depositate in una coda opportuna.

Motion Executors
Attuano movimenti parametrizzati o pianificati spezzandoli in una sequenza di azioni ognuna delle quali specifica una sequanza di controlli relativi allo stato sensoriale e di moti elementari che vengono inseriti nella coda del *Trajectory Control*. In particolare sono stati realizzati due *Motion Executor* denominati *Grasp* e *Put* rispettivamente. *Grasp* esegue le operazioni di scarico del robot mobile mentre *Put* attua i piani calcolati dal *Motion Planner* [2]. I *Motion Executor* sono normalmente bloccati su di una variabile condizione in attesa di un evento. Ad esempio, *"è a disposizione un nuovo piano"* oppure *"sono presenti n oggetti sul robot mobile"* sono due fra i possibili eventi attesi. In caso di contesa, è data priorità all'esecutore *Grasp*, mentre *Put* viene ritardato (si vuole infatti scaricare il robot mobile nel più breve tempo possibile). In generale, quando uno dei due *Executor* è attivo e si verificano le condizioni per l'esecuzione dell'altro, questi viene differito, cioè bloccato nuovamente sulla variabile condizione. Sarà l'*Executor* attivo a risvegliarlo al termine della sua esecuzione.

Motion Planner
Calcola il percorso che consente al manipolatore di depositare gli oggetti nei contenitori a partire dall'area di deposito. A tale scopo esegue un pianificatore di moti che si avvale di tecniche a campo di potenziale per la determinazione di un percorso libero in uno spazio 3D con ostacoli. I parametri necessari alla determinazione del percorso, cioè posizioni di partenza/arrivo, numero e posizione degli oggetti e dei contenitori, morfologia dell'ambiente, sono in parte predefiniti e in parte rilevati attraverso la sensorialità del robot. Ogni percorso pronto viene introdotto nella coda di esecuzione dell'agente *Put* e l'evento *"è a disposizione un nuovo piano"* è generato dallo stesso *Motion Planner* che l'ha calcolato. I *Motion Planner* sono istanziati durante la fase di attivazione del sistema. Normalmente bloccati, vengono risvegliati dal *Supervisor* quando è necessario un nuovo piano, cioè quando un nuovo oggetto è prelevato dal robot mobile.

Supervisor
Periodicamente analizza le informazioni sensoriali e attiva i *Motion Agent (Executors e Planners)* in modo

opportuno al fine di eseguire il compito di assemblaggio. Nel caso specifico descritto in questa scheda, il *Supervisor* è un semplice sequenzializzatore di azioni per cui il suo carico computazionale è modesto.

Sensor Agents
Periodicamente leggono i dati sensoriali (telecamera, infrarossi ed ultrasuoni) dai dispositivi periferici, eseguono sugli stessi alcuni filtraggi e li depositano in opportune aree di memoria condivise. I tempi sono brevi perché tutta l'elaborazione è affidata alle unità esterne dedicate. Inoltre, a causa delle limitate prestazioni di tali unità, l'informazione non viene prodotta al rate massimo di interesse.

Timing Monitor
Periodicamente calcola il fattore di utilizzo del sistema e memorizza le statistiche relative ai parametri real-time (tempi di esecuzione, deadline misses, skips, etc.) dell'architettura di controllo.

Riferimenti bibliografici

[1] G. Beccari, S. Caselli, M. Reggiani, F. Zanichelli, "A Real-Time Library for the Design of Hybrid Robot Control Architectures", IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'98, Victoria, BC, Canada, October 1998.

[2] "Robot manipolatore per compiti di assemblaggio in ambiente dinamico", MOSAICO Scheda I, Unità di Ricerca *Università di Parma* Dipartimento di Ingegneria dell'Informazione Viale delle Scienze - 43100 Parma - Italy.