ALMA MATER STUDIORUM
UNIVERSITÀ DEGLI STUDI DI BOLOGNA

Scuola di Ingegneria e Architettura
Corso di Laurea in Ingegneria Informatica, Elettronica e Telecomunicazioni

APPRENDIMENTO ROBOTICO

Elaborata nel corso di: Fondamenti di Informatica B

Tesi di Laurea di:
NICOLA CASADEI

Relatore:
Prof. ANDREA ROLI

ANNO ACCADEMICO 2012–2013
SESSIONE III

PAROLE CHIAVE

Apprendimento Automatico
Apprendimento per rinforzo
Apprendimento per dimostrazione
Robotica

Alla mia famiglia, per avermi sempre sostenuto.
A mio fratello, da sempre motivo d'orgoglio ed esempio da seguire.

Indice

Introduzione

La robotica è,oggi, uno dei campi di ricerca di maggiore interesse ed è sempre in continua evoluzione. L'inserimento dei robot nella società moderna è, sicuramente, uno degli obbiettivi della robotica.

Il raggiungimento di questo obbiettivo dipende fortemente dalle capacità che il robot presenta nell'interazione con l'ambiente. Quindi, dotare un robot di nuove capacità per aumentare la sua autonomia nell'ambiente esterno, è un problema fondamentale nella robotica.

I metodi per affrontare questo problema possono essere raccolti sotto il nome di *Apprendimento Robotico*. Quest'ultimo può essere ulteriormente suddiviso in tre grandi paradigmi: la programmazione diretta, l'apprendimento per rinforzo e l'apprendimento per dimostrazione.

Illustrare in modo completo ed esauriente tali tecniche e tutte le loro possibili implementazioni è un compito assai arduo per una singola tesi.

Lo scopo di questo elaborato consiste principalmente in una presentazione generale e teorica dei paradigmi più interessanti dell'apprendimento robotico.

In particolare,nel capitolo 1, dopo una breve introduzione sull'apprendimento automatico, si vogliono presentare le motivazioni e le basi teoriche che stanno dietro all'apprendimento per rinforzo. Nel dettaglio, vengono presentate le origini storiche del paradigma, l'algoritmo Q-Learning e i limiti attuali del paradigma.

Nel capitolo 2, invece, l'attenzione ricade su un altro paradigma di apprendimento, quello per dimostrazione. Similmente al capitolo precedente si vuole presentare in modo generale le tecniche di dimostrazione più utilizzate con i rispettivi pro e contro e si conclude, sempre, con i limiti del paradigma.

Nel capitolo 3, l'obbiettivo è presentare i possibili vantaggi e svantaggi dell'utilizzo di queste tecniche, insieme alla programmazione diretta, nel campo della robotica. In particolare, prendendo spunto dai lavori di Kober [25] e Kormushev [28] si vuole presentare come l'utilizzo congiunto di questi paradigmi può essere sfruttato per l'apprendimento di comportamenti molto complessi in un ambiente dinamico.

Capitolo 1

Apprendimento per rinforzo

1.1 Apprendimento Automatico

Attualmente la maggioranza dei calcolatori si basa su una elaborazione di tipo simbolico, ovvero il problema viene prima codificato in un insieme di variabili e successivamente elaborato tramite un algoritmo esplicito che, per ogni Input possibile del problema, offre un adeguato output. Esistono però problemi in cui la risoluzione tramite un algoritmo esplicito risulta inefficiente o addirittura innaturale, per esempio un riconoscitore vocale; affrontare questo genere di problema con l'approccio classico risulta inefficiente per i motivi sotto menzionati. Non è, infatti, sufficiente analizzare il segnale audio e successivamente eseguire su di esso un'elaborazione, quale il controllo del livello per separare parole o distinguere sillabe. In questo genere di problemi ci sono troppe variabili in gioco: velocità di pronuncia, prosodia e altre condizioni di rumore, tutte variabili impossibili da formalizzare completamente, senza considerare la difficoltà stessa del modellare un problema di questo genere per un programmatore. Questo e altri problemi simili, come la navigazione autonoma di un robot o la ricerca di un pattern comune in un'insieme di dati, fanno parte di un'insieme molto vario di problemi che non sono affrontabili direttamente tramite soluzioni basate sull'elaborazione simbolica.

L'Apprendimento Automatico [36], [3] offre un nuovo tipo di approccio, che consiste nella costruzione e nello studio di sistemi capaci di imparare dalle informazioni, queste possono essere un'insieme di

esempi osservati e dati raccolti, successivamente analizzati per trovare relazioni nella struttura dati o nelle variabili osservate. Soluzioni basate su algoritmi di apprendimento automatico, tra cui anche il riconoscimento vocale, danno risultati di gran lunga superiori rispetto alle soluzioni con algoritmi a elaborazione simbolica.

Definizione e paradigmi fondamentali

Una prima interessante definizione di Apprendimento Automatico o Machine Learning viene data da Arthur Samuel nel 1959 [49]:

Field of study that gives computers the ability to learn without being explicitly programmed

Questa definizione risulta piuttosto semplice e informale, ma da essa si capisce immediatamente la potenza di questo nuovo tipo di approccio. La possibilità di dotare una macchina di una limitata capacità di apprendimento permette a un programmatore di insegnare un nuovo compito alla macchina senza dover esplicitamente programmare la macchina per farlo.

Una definizione più formale viene fornita da T. Mitchell nel 1997 [36]:

Un programma apprende da una certa esperienza E se: nel rispetto di una classe di compiti T, con una misura di prestazione P, la prestazione P misurata nello svolgere il compito T è migliorata dall'esperienza E

Questa definizione è comune a tutte le tipologie di algoritmi di apprendimento, che hanno, infatti, quale caratterista comune più importante,la presentazione in Input alla macchina di una serie di esempi chiamata Training Pattern, che, per citare nuovamente Mitchell, non sono altro che istanze definite dell'esperienza E. La soluzione al problema si ottiene quando, dopo un numero sufficiente di istanze E, la macchina si comporta nel modo desiderato. Gli algoritmi di apprendimento automatico sono, quindi, classificati in base a due fattori: il

tipo di output emesso dall'algoritmo e quello delle modalità in cui vengono presentati gli esempi di apprendimento.

Gli algoritmi di apprendimento possono essere suddivisi in tre grandi classi:

- **Apprendimento non supervisionato**: questi algoritmi ricevono in input un vettore di dati o esempi, lo scopo dell'algoritmo e il relativo apprendimento consistono nell'individuare nel Training Pattern alcune caratteristiche distintive o comuni che permettono di eseguire un'operazione di classificazione sugli esempi di input. Il processo di apprendimento però non riceve alcun feedback dall'esterno, quindi l'utente non ha modo di prevedere a priori l'output dell'operazione di classificazione. In pratica l'algoritmo realizza delle operazioni di tipo statistico per individuare una struttura ricorrente nel pattern di ingresso che, successivamente, può essere riutilizzata per riconoscere e classificare altre istanze di input che non facevano parte del vettore di esempi. Questo genere di algoritmi trovano il loro massimo utilizzo nella risoluzione di problemi di *Clustering* o *Data Mining*.

- **Apprendimento supervisionato**: gli algoritmi di apprendimento supervisionato, invece, ricevono in ingresso un vettore di Input che rappresenta il Training Pattern, a cui, a differenza dell'apprendimento non supervisionato, a ogni input è associato un output desiderato. L'obbiettivo finale di questi algoritmi è estrapolare dagli esempi dati in ingresso, la funzione che lega la coppia di Input-Output e, dato un numero sufficiente di esempi, la funzione estrapolata offrirà una soluzione accettabile anche per nuovi Input che non facevano parte del training pattern.

  Ricapitolando con un linguaggio più formale: alla macchina vengono forniti in ingresso una serie di esempi di solito rappresentati come un vettore, a ogni input I viene associato l'output O desiderato. L'obbiettivo dell'algoritmo consiste nell'estrapolare e migliorare, da ogni coppia (I, O) , data come esempio, una funzione h_a chiamata anche *Ipotesi Induttiva*, funzione che deve avvicinarsi il più possibile alla ipotetica funzione h_b chiamata *funzione obbiettivo*. Se gli esempi dati come Training Pattern erano sufficienti sia in numero che in qualità, la funzione h_a estrapolata

dall'algoritmo sarà sufficientemente simile alla funzione obbiettivo h_a, permettendo così di approssimare in modo accettabile l'output fornito dall'algoritmo di fronte a un ingresso I non facente parte del Training Pattern iniziale. Un parametro F, di solito rappresentato come la differenza di output fra h_a e h_b, è utilizzato come stima di efficienza dell'apprendimento. Questi algoritmi generalmente sfruttano i principi della distribuzione matematica e della funzione di verosimiglianza. Una volta individuata la funzione di distribuzione che lega l'input all'output, si generano dei parametri tali che massimizzino la probabilità di generare la funzione di verosimiglianza appropriata. Anche questi tipi di algoritmo sono utilizzati spesso in operazioni di *Pattern Recognition*, *Speech recognition* e *Handwriting recognition*,operazioni in cui, sopratutto negli ultimi due casi, l'algoritmo deve essere guidato a fornire un output accettabile, cosa che non è possibile per gli algoritmi non supervisionati.

- **Apprendimento per rinforzo**: La letteratura scientifica è ancora molto confusa sulla classificazione dell'apprendimento per rinforzo come paradigma o meno. Inizialmente considerato come caso particolare dell'apprendimento supervisionato, trova però applicazioni in contesti diversi nei quali l'apprendimento supervisionato risulta inefficiente, i problemi di interazione con l'ambiente ne sono un chiaro esempio.

1.2 Apprendimento per rinforzo

1.2.1 Introduzione

L'apprendimento per rinforzo [52] fonda le sue basi su un'interessante teoria di psicologia [55]:

Applying a reward immediately after the occurrence of a response increases its probability of reoccurring, while providing punishment after the response will decrease the probability (Thorndike, 1911)

Se viene fornito un premio subito dopo l'esecuzione di un comportamento considerato corretto, aumenta la probabilità che questo comportamento si ripeta. Mentre a fronte di un comportamento non desiderato, l'applicazione di una punizione decrementa la probabilità di una ripetizione dell'errore.

Pertanto, definito un obbiettivo da raggiungere, l'apprendimento per rinforzo cerca di massimizzare le ricompense ricevute per l'esecuzione dell'azione o dell'insieme di azioni che permettono di raggiungere l'obbiettivo designato.

Considerare l'apprendimento per rinforzo un caso particolare dell'apprendimento supervisionato risulta, quindi, riduttivo: le due metodologie si differenziano non solo per il funzionamento dei rispettivi algoritmi ma anche per diverse problematiche da affrontare in fase di analisi e definizione del problema.

Per questi motivi si può considerare l'apprendimento per rinforzo un paradigma a parte di apprendimento automatico, che trova maggiore successo se applicato a problemi di interazione di un agente con l'ambiente esterno.

1.2.2 Cenni storici

La storia dell'apprendimento per rinforzo segue tre flussi di ricerca, distinti e quasi paralleli, che, solo verso la fine degli anni '80, confluirono nella moderna concezione di apprendimento per rinforzo.

Il primo di questi flussi cercava soluzioni al problema del controllo ottimo, ovvero un algoritmo che stabilizzasse un sistema dinamico. Uno degli approcci al problema fu sviluppato da Richard Bellman verso la metà degli anni '50 e consisteva nell'utilizzare i concetti di stato dinamico del sistema e di funzione di valutazione per definire un'equazione funzionale, oggi conosciuta come equazione di Bellman. I metodi utilizzati per la soluzione dei problemi di controllo ottimo vengono classificati come programmazione dinamica e introducono la versione discreta e stocastica del problema di controllo ottimo conosciuto anche come Markov Decision Processes (MDP), che pochi anni dopo, nel 1960, Ron Howard definì un processo di interazione per il MDP [9] [10].

Il secondo flusso basava le sue ricerche sulle possibili applicazioni pratiche del Trial-and-error Learning, inizialmente osservato in psicologia, dove venivano già studiate le teorie di apprendimento per rinforzo nel campo dell'apprendimento animale.

Quegli stessi studi di psicologia che permisero al già citato Thorndike di definire la sua *Law of Effect* [55], che gettò le basi del trail-and-error Learnig, ovvero permise di definire i concetti di selezione e associazione: il primo consiste nel poter scegliere tra un insieme di possibili comportamenti sulla base delle conseguenze che implicano, l'associazione ,invece, consiste nell'accoppiare una particolare scelta a una determinata situazione.

Risulta evidente che il concetto moderno di apprendimento per rinforzo è un esempio lampante di coesistenza di entrambi i concetti, mentre l'apprendimento con supervisione risulta associativo ma non selettivo, ulteriore conferma della differenza sostanziale dei due metodi di apprendimento.

Le prime applicazioni di trial-and-error learning nel campo dell'intelligenza artificiale furono studiate, nel 1954, da Minsky e da Farley e Clark. Entrambe le ricerche portarono allo sviluppo di un modello di calcolatore che utilizzava un processo di apprendimento ispirato al trial-and-error [34].

A partire dalla metà degli anni '60, i termini rinforzo e apprendimento per rinforzo fecero le prime apparizioni nella letteratura ingegneristica e.g. Waltz and Fu (1965)[58], Mendel (1966), Fu (1969)[17], Mendel and McClaren (1970) [31]. Particolarmente innovativo fu l'articolo di Minsky *Steps Towards Artificial Intelligence* che introdusse per la prima volta il problema di *credit assignment*, ovvero come potevano essere distribuiti crediti a seguito di un comportamento corretto per stimolare la sua rioccorenza [35].

Degno di nota risulta, anche, il lavoro di Donald Michie, il quale tra il 1961 e il 1963 [32] [33] descrisse un semplice sistema, chiamato MENACE, che, con l'esperienza, migliorava la sua abilit1à nel gioco del tris. L'apprendimento di MENACE si basava in pratica sul fornire o meno una ricompensa a seguito di una particolare mossa.

Nel 1973, i ricercatori Widrow, Gutpa e Matria modificarono l'algoritmo Least Mean Square, sviluppato dallo stesso Widrow. La nuova versione dell'algoritmo risolveva lo stesso problema grazie all'in-

vio di segnali di successo o di fallimento. Questo particolare approccio fu battezzato *apprendimento con un critico*, sinonimo ancora oggi dell'apprendimento per rinforzo [61] .

I lavori di Widrow, come quelli di Farly e Clark [13], furono però isolati e il loro maggiore contributo andrà, invece, all'apprendimento con supervisione, fatto che sicuramente ha influenzato l'attuale confusione sulla classificazione degli algoritmi di apprendimento per rinforzo.

Il terzo e ultimo flusso di ricerca cercava applicazioni nell'intelligenza artificiale del *temporal-difference learning*. Questo metodo di apprendimento si basava sulla stima, in instanti di tempo successivi, di una stessa quantità, per maggiori informazioni si faccia riferimento a [54] [53].

Anche queste ricerche rimasero indipendenti fino al 1972, quando Klopf propose di applicare al trial-and-error learning alcuni concetti del temporal-difference learning. Egli sviluppò, infatti, il concetto di *rinforzo generalizzato*, in cui ogni sottocomponente di un sistema di apprendimento poteva rinforzare un qualsiasi altro componente dello stesso sistema. I segnali di input, in pratica, venivano trattati come ricompense se erano segnali eccitatori o come punizioni se erano segnali inibitori.

Nel 1981, le idee di Klopf [23] [24], portarono allo sviluppo del primo metodo che univa i concetti principali di temporl-difference e trail-and-error learning, conosciuto meglio come actor-critic architecture [8]. Questa architettura innovativa fu utilizzata da Sutton (1984)[51] come nuova soluzione al problema di bilanciamento dell'asta ed estesa da Anderson (1986)[4] per utilizzarla nell'algoritmo di *backpropagation* nelle reti neurali.

Fu solo nel 1989 che i risultati ottenuti indipendentemente da questi tre flussi di ricerca furono utilizzati da Chris Watkins nello sviluppo dell'algoritmo cardine dell'apprendimento per rinforzo il *Q-Learning* [60] [59].

Oggi le applicazioni di apprendimento per rinforzo sono innumerevoli, interessanti sono gli ultimi lavori di Kormushev e Billiard [26] [11], i quali utilizzano tecniche di apprendimento per rinforzo per migliorare l'esecuzione di alcuni comportamenti appresi, da un robot,

tramite dimostrazione.

1.2.3 Analisi del problema

L'interfaccia Agente-Ambiente

Il problema dell'apprendimento per rinforzo può essere visto come un caso particolare del problema di interazione per il raggiungimento di un obbiettivo o goal. L'entità che deve raggiungere l'obbiettivo è chiamata agente. L'entità con cui l'agente deve interagire viene denominata ambiente, che corrisponde con tutto ciò che è esterno all'agente stesso. L'interazione agente-ambiente è continua, l'agente sceglie un' azione da intraprendere e in risposta l'ambiente cambia stato presentando una nuova situazione da affrontare.

Nel caso particolare dell'apprendimento per rinforzo, l'ambiente fornisce all'agente una ricompensa, è fondamentale che la sorgente della ricompensa sia l'ambiente onde evitare il formarsi, all'interno dell'agente, di un meccanismo di rinforzo personale che comprometterebbe l'apprendimento.

Il valore della ricompensa è proporzionale all'influenza che ha l'azione nel raggiungimento dell'obbiettivo, quindi risulta positiva o elevata nel caso si tratti di un'azione corretta oppure negativa o bassa per un'azione scorretta.

Infatti, caratteristica particolare del paradigma di apprendimento per rinforzo è il considerare il problema nella sua interezza; la pratica comune di suddividere il problema in sotto problemi è ancora applicabile ma non strettamente necessaria al fine di trovare una soluzione.

Secondo un linguaggio più formale, l'agente e l'ambiente interagiscono a intervalli discreti nel tempo, $t = 0, 1, 2, , n$. A ogni intervallo l'agente riceve una rappresentazione dello stato s_t dell'ambiente.

Ogni elemento $s_t \in S$, dove S è l'insieme dei possibili stati, una volta riconosciuto lo stato l'agente deve intraprendere un'azione $a_t \in A(s_t)$, dove $A(s_t)$ è l'insieme delle azioni possibili nello stato s_t. La scelta dell'azione da intraprendere dipende dall'obbiettivo da raggiungere e viene mappata attraverso la *politica* o *policy* indicata con π, la quale associa per ogni stato s l'azione $a \in A(s)$. La formulazione $\pi_t(s, a)$ rappresenta la probabilità che l'azione a venga effettuata

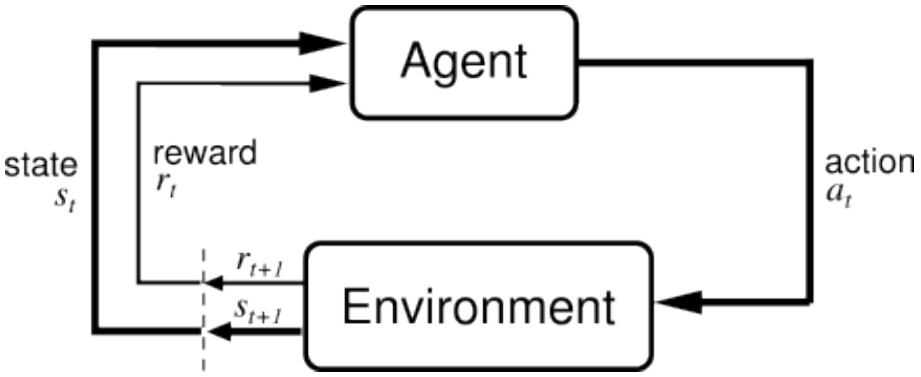

Figura 1.1: *interazione agente-ambiente nell'apprendimento per rinforzo, immagine tratta da Reinforcement Learning:An Introduction Richard S. Sutton and Andrew ,G. Barto, A Bradford - Book The MIT Press Cambridge, Massachusetts London, England*

nello stato s. Durante il successivo intervallo temporale $t+1$, come parte della conseguenza dell'azione s_{t+1}, l'agente riceve una ricompensa numerica $r_{t+1} \in \Re$ corrispondente all'azione precedentemente intrapresa a_t. La conseguenza dell'azione rappresenta, invece, il nuovo stato s_t , a questo punto l'agente deve nuovamente codificare lo stato ed eseguire la scelta dell'azione. Questa iterazione si ripete fino al raggiungimento dell'obbiettivo da parte dell'agente. Nel caso generale, la definizione dello stato s_{t+1} dipende dalla successione degli stati precedenti e dall'azione intrapresa, ovvero:

$s_{t+1} = \delta(s_t, s_{t-1}, ..., s_1, s_0, a_t)$ dove δ rappresenta la funzione di stato.

Per ovvi problemi di carico e difficoltà computazionali questa formulazione non è applicabile. Per risolvere questo problema si deve assumere che l'ambiente goda della *proprietà di Markov*, ovvero che lo stato s_{t+1} dipenda esclusivamente dallo stato precedente s_t e dall'azione eseguita a_t, in altre parole si presuppone che la scelta dell'azione possa essere effettuata tramite l'applicazione del Markov Decision Process MDP [30] .

La nuova formulazione risulta pertanto:

$$s_{t+1} = \delta(s_t, a_t)$$

L'ambiente può essere, quindi, descritto tramite la logica *one step*, caratterista fondamentale per la risoluzione di un problema di apprendimento per rinforzo.

Tecniche di rinforzo

Una possibile soluzione al problema consiste nell'associare a ogni singolo stato s l'azione che fornisce la ricompensa r maggiore , ovvero si deve determinare una policy ottima tale che:

$$\max\{R_t = \sum_{i=0}^{i=T} r_{t+i} = r_t + r_{t+1} + ... + r_T\}$$

dove r_T rappresenta la ricompensa dell'azione che porta l'ambiente nello stato terminale s_T, chiamato anche *stato terminale*.

Questa, però, rappresenta solamente una soluzione particolare, applicabile solo per i cosiddetti *episodic tasks*, quei problemi risolvibili in un insieme finito di azioni.

Per i problemi, chiamati *continuing tasks*, che non raggiungono l'obbiettivo o stato terminale in un numero finito di passaggi, la formulazione prima esposta risulterà:

$$\max\{R_t = \sum_{i=0}^{i=\infty} r_{t+i} = r_t + r_{t+1} + ... + r_n + r_{n+1} + ...\}$$

da cui risulta evidente:

$$R_t \to \infty$$

Questa formulazione non è applicabile, la somma delle ricompense che si vuole massimizzare diverge all'infinito, è necessario, perciò, sviluppare una tecnica di rinforzo alternativa; una possibile soluzione consiste nella *tecnica di rinforzo con premio medio*:

$$\max\{R_t = \sum_{i=0}^{i=\infty} r_{t+i} = r_t + r_{t+1} + ... + r_n + r_{n+1} + ...\} \text{ con } r_{t+i} \geq r_{t+i-1} \forall i$$

Quindi $r_{t+i} \to 0$ con $i \to \infty$.

La ricompensa diminuisce a ogni istante di tempo, le azioni compiute per prime ricevono le ricompense maggiori, mentre le ultime diminuiscono diventando trascurabili con il passare del tempo.

Il rinforzo con premio medio risulta, però, ancora imperfetto: sintetizzare un problema di interazione, considerando importanti solo le prime azioni intraprese, risulta troppo limitativo e in alcuni casi profondamente errato.

La tecnica che più si adatta al paradigma risulta essere il *rinforzo con premio scontato*:

$$\max \{R_t = \sum_{i=0}^{i=\infty} \gamma^i r_{t+i} = r_t + \gamma r_{t+1} + ... + \gamma^n r_n + \gamma^{n+1} r_{n+1} + ...\} \text{ con } 0 \leq \gamma \leq 1$$

il parametro γ viene chiamato *fattore di sconto* e rappresenta l'importanza per le future ricompense. Se $\gamma < 1$ la successione r_t convergerà a un valore finito, se $\gamma = 0$ l'agente non avrà interesse nelle ricompense future, ma tenterà di massimizzare la ricompensa solo per lo stato attuale. Infine, se $\gamma = 1$ l'agente cercherà di aumentare le ricompense future anche a discapito di quelle immediate. Il fattore di sconto può essere modificato durante il processo di apprendimento per evidenziare o meno azioni o stati particolari.

Esplorazione vs Sfruttamento

Idealmente, l'agente deve associare a ogni azione $a_t \in A(s_t)$ la rispettiva ricompensa r per poter scegliere poi il comportamento più ricompensato per il raggiungimento dell'obbiettivo. Questo approccio risulta, però, impraticabile per problemi complessi in cui il numero di stati è particolarmente elevato e di conseguenza le associazioni possibili aumentano in modo esponenziale. Questo problema è riconducibile all'*Exploration-Expoitation dilemma* [43] .

Idealmente, l'agente deve esplorare (exploration) tutte le possibili azioni per ogni stato, trovando quella effettivamente più ricompensata per sfruttarla (exploitation) nel raggiungimento del suo obbiettivo. In pratica, però, in problemi molto complessi, la convergenza a una strategia ottima sarebbe troppo lenta.

Una buona soluzione al problema risulta trovare un equilibrio tra esplorazione e sfruttamento:

- un agente che si limita a esplorare, agirà sempre in modo casuale a ogni stato ed è evidente che la convergenza a una strategia ottima è impossibile;
- d'altro canto, se un agente esplora poco, utilizzerà sempre le solite azioni che potrebbero non essere quelle ottimali.

1.2.4 Q-Learning

La funzione di valutazione *V*

Una soluzione generale al problema di apprendimento per rinforzo consiste nello stimare, grazie al processo di apprendimento, *una funzione di valutazione*. Questa funzione deve poter valutare, tramite la somma delle ricompense, la convenienza o meno di una particolare politica π.

Ricordando che la politica π associa alla coppia (s, a) la probabilità $\pi(s, a)$ che venga effettuata l'azione a nello stato s. Si può definire una prima funzione di valutazione $V^\pi(s)$ come il valore atteso del rinforzo totale R_t seguendo la politica π a partire dallo stato s:

$$V^\pi(s) \equiv \sum_{i=0}^{i=\infty} \gamma^i r_{t+i}$$

dove la successione di r_t viene generata seguendo la politica π a partire dallo stato s. In altre parole gli esempi del *training pattern* devono guidare il processo di apprendimento verso la valutazione della politica ottimale π^*:

$$\pi^* \equiv \arg\max_\pi V^\pi(s) \ \forall s$$

Ricordando la funzione $\delta(s, a)$, la quale determina il nuovo stato generato dalla coppia (s, a), si può operare una ricerca *lookahead* per scegliere l'azione migliore a partire dallo stato s poiché si può esprimere:

$$\pi^*(s) = \arg\max_a \{r(s, a) + \gamma V^{\pi^*}(\delta(s, a))\}$$

dove $r(s, a)$ rappresenta la ricompensa r ottenuta dall'esecuzione dell'azione a nello stato s.

Questa soluzione però è accettabile solo nel caso siano note le funzioni:

$$\delta : S \text{ x } A \to S$$

$$r : S \text{ x } A \to \Re$$

questa condizione non è, però, sempre rispettata.

La funzione di valutazione *Q*

Quando questo non accade è necessario definire una nuova funzione simile a V^{π^*}:

$$Q(s, a) \equiv r(s, a) + \gamma V^{\pi^*} \delta(s, a)$$

Se l'agente riesce a stimare la funzione Q risulta possibile scegliere l'azione ottimale s anche senza conoscere la funzione δ:

$$\pi^*(s) = \arg\max_a Q(s, a)$$

La funzione Q viene di solito indicata come *funzione di valore di azione* o *action-value function*.

Le funzioni di valutazione Q e V^{π^*} sono strettamente legate, infatti si può scrivere:

$$V^{\pi^*}(s) = \max_{a'} Q(s, a')$$

da cui è possibile esprimere in modo ricorsivo Q:

$$\begin{aligned} Q(s_t, a_t) &= r(s_t, a_t) + \gamma V^{\pi^*}(\delta(s_t, a_t)) \\ &= r(s_t, a_t) + \gamma \max_{a'} Q(s_{t+1}, a') \end{aligned}$$

L'apprendimento consiste nell'aggiornare, a ogni azione, la funzione approssimata $\tilde{Q}$ secondo la seguente *regola di training*:

$$\tilde{Q}(s, a) \leftarrow r + \gamma \max_{a'} \tilde{Q}(s', a')$$

dove s' è lo stato risultante dall'esecuzione dell'azione a nello stato s. Questa regola di training risulta, però, applicabile solo per ambienti deterministici, ovvero quando l'agente riesce sempre a determinare come reagirà l'ambiente di fronte all'esecuzione di una particolare azione.

Nel caso di ambienti non deterministici le funzioni V e Q possono essere riformulate considerando i valori attesi, in particolare Q risulterà:

$$Q(s,a) \equiv E\{r(s,a) + \gamma V^*(\delta(s,a))\}$$

dove $E\{\}$ rappresenta il valore atteso, stimato, della ricompensa R_t ottenuta eseguendo l'azione a nello stato s.

La regola di training può essere modificata per un'applicazione anche in casi non deterministici in questo modo:

$$\tilde{Q}_n(s,a) \leftarrow (1-\alpha_n)\tilde{Q}_{n-1}(s,a) + \alpha_n[r + \max_{a'} \tilde{Q}_{n-1}(s',a')]$$

dove α è il parametro di *learning rate*, compreso tra i valori $0 \leq \alpha \leq 1$ e rappresenta l'importanza delle nuove informazioni acquisite con l'apprendimento: $\alpha = 0$ significa che l'agente smette di apprendere dall'esperienza con $\alpha = 1$, invece, l'agente sovrascrive le vecchie informazioni con quelle nuove acquisite, anche se peggiori.

La prima parte:

$$\tilde{Q}_n(s,a) \leftarrow (1-\alpha_n)\tilde{Q}_{n-1}(s,a)$$

rappresenta il vecchio valore che, come suesposto, tramite il learning rate, può essere più o meno modificato.

La seconda parte:

$$\alpha_n[r + \max_{a'} \tilde{Q}_{n-1}(s',a')]$$

rappresenta, invece, il nuovo valore appreso grazie alla n-esimo aggiornamento.

La regola di training sopra esposta è utilizzata nell'algoritmo base dell'apprendimento per rinforzo il *Q-Learning*, il quale è alla base di quasi tutte le varianti degli algoritmi di questo paradigma.

La convergenza di $\tilde{Q} \to Q$ viene idealmente assicurata dal seguente teorema di convergenza:

Si consideri un ambiente modellato come un processo di decisione Markoviano, tale che $\forall s \in S, a \in A \ \ |r(s,a)| \leq C$. *L'algoritmo del Q-Learning converge a* $\tilde{Q}$ *con probabilità = 1, sotto l'ipotesi che ogni*

coppia (s, a) *venga visitata infinite volte e il fattore di apprendimento* α *tenda a zero adeguatamente.*

È evidente che la coppia (s, a) non può essere visitata infinite volte, quindi il processo di apprendimento riesce a dare solo una stima della soluzione, che è sempre più precisa iterazione dopo iterazione.

Il processo di apprendimento

Uno degli esempi più utilizzati per implementare l'algoritmo Q-Learning, prevede l'utilizzo di una tabella.

Ogni cella della tabella rappresenta $\tilde{Q}(s, a)$ ed è inizializzata a 0. L'agente può eseguire una qualsiasi azione $a \in A$ dove A è l'insieme totale delle azioni conosciute dall'agente. L'algoritmo segue questi passi fondamentali:

1. si decodifica la stato s
2. si seleziona un'azione $a \in A(s) \subset A$, dove $A(s)$ è il già definito insieme delle azioni eseguibili in s;
3. l'azione a viene eseguita e si riceve la ricompensa r
4. l'elemento della tabella $\tilde{Q}(s, a)$ viene aggiornato con la regola di training:

$$\tilde{Q}_n(s, a) \leftarrow (1 - \alpha_n)\tilde{Q}_{n-1}(s, a) + \alpha_n[r + \max_{a'} \tilde{Q}_{n-1}(s', a')]$$

5. l'esecuzione dell'azione a porta l'ambiente nello stato s'
6. $s \leftarrow s'$
7. ricominciare dal punto 1.

In formulazioni più complesse ed efficienti è possibile sostituire la tabella, la cui iterazione risulta comunque inefficiente per problemi complessi, con una rete neurale dove il processo di apprendimento andrà a modificare i pesi delle connessioni sinaptiche, per maggiori informazioni si faccia riferimento a: [14], [50].

1.2.5 Limiti dell'apprendimento per rinforzo

L'apprendimento per rinforzo risulta un paradigma molto solido per quanto riguarda la risoluzione di problemi di interazione con l'ambiente, l'assunzione che questo ambiente rispetti la proprietà Markoviana è sicuramente una limitazione, ma non delle peggiori, infatti, nella maggior parte dei problemi di interazione, l'ambiente gode o può essere ricondotto con una semplificazione accettabile a un *ambiente markoviano*. I veri limiti che pesano sul futuro dell'apprendimento per rinforzo sono i seguenti:

- *Curse of dimensionality*: l'applicazione in problemi reali è di solito difficoltosa, il grande numero di stati possibili, unito ad un sempre maggior numero di azioni richieste, aumenta in modo esponenziale la difficoltà di apprendere una politica ottima.

- *(Temporal) Credit Assignment Problem*: conseguenza della Curse of dimensionality, è spesso necessario assegnare una ricompensa a un insieme di azioni e non ad ogni singola azione eseguita. Perciò si tende a ricompensare un comportamento piuttosto che una singola azione. Ne risulta, quindi, che il processo di apprendimento è tanto lento quante sono le azioni che formano il comportamento ricompensato. Ciò è dovuto al fatto che gli effetti della ricompensa si devono propagare attraverso le varie azioni che lo compongono.

- *Partial Observability Problem*: nei problemi reali la codifica esatta dello stato in cui si trova l'ambiente è spesso impossibile, risulta, pertanto, necessario accontentarsi di un'approssimazione della codifica dello stato tramite una percezione parziale di esso. La codifica parziale dello stato determina solo un sottoinsieme degli stati dell'ambiente reale, quindi, un qualsiasi agente non sarà mai autosufficiente in questo genere di ambienti.

- *State-Action Space Tilling*: la scelta della granularità dello spazio Stato-Azione determina direttamente l'influenza sul problema della Curse of Dimensionality e della Temporal Credit Assignment. Infatti, una granularità troppo fine risentirà mag-

giormente della Curse of Dimensionality, mentre una granularità troppo grossa della Temporal Credit Assignment.

- *Non-Stationary Environments*: la convergenza degli algoritmi di apprendimento per rinforzo è molto lenta, quindi efficace solo in ambienti stazionari o al limite dinamicamente molto lenti. Un ambiente reale risulta, però, non stazionario e nella maggior parte dei casi molto dinamico. L'apprendimento per rinforzo non riuscirebbe, perciò, a convergere verso una politica ottima a causa dei continui cambiamenti dell'ambiente.

- *Exploration-Exploitation Dilemma* esposto già in analisi del problema, viene qui riportato per completezza.

Capitolo 2

Apprendimento per dimostrazione

2.1 Introduzione

Il grande sviluppo che l'uomo ha raggiunto in ambito tecnologico, in tempi relativamente brevi, ha permesso la diffusione della tecnologia stessa in ogni aspetto della sua vita; basti pensare alla grande diffusione del computer, nato come un costoso e ingombrante calcolatore utilizzato da un ristretto numero di ricercatori come supporto al loro lavoro.

Ora, grazie al rapido sviluppo tecnologico, la dimensione e il costo di queste macchine è stato abbattuto rendendole un bene accessibile alla maggior parte della popolazione. La stessa potenza di calcolo è in continuo aumento permettendo un uso sempre maggiore di questi calcolatori, ormai utilizzati dall'uomo in ogni sua attività: dal lavoro, al tempo libero.

Gli stessi avvenimenti si stanno ripetendo per quanto riguarda la diffusione dei robot, inizialmente presenti solo in un contesto di ricerca. Le loro potenzialità stanno rapidamente aumentando e un futuro dove l'uomo è affiancato da un robot nell'esecuzione di una qualsiasi attività non è poi così lontano.

I robot, infatti, sono già presenti nelle catene di montaggio di molte aziende piccole o grandi, i primi semplici robot domestici hanno fatto la loro comparsa nelle abitazioni e hanno già sostituito l'uomo

in compiti per lui troppo pericolosi o impossibili, basti pensare alla missione NASA di esplorazione di Marte, possibile solamente grazie ai due robot esploratori *Spirit* e *Opportunity*.

L'interazione uomo-robot e la futura integrazione di quest'ultimo nella società risulta più che mai evidente; così, come lo sviluppo delle prime interfacce grafiche è risultato un passo fondamentale nella diffusione dei computer, la ricerca in ambito robotico è quanto mai indirizzata nello sviluppo di nuove e flessibili interfacce per semplificare l'interazione e l'integrazione dei robot nella società moderna.

Queste motivazioni hanno ispirato la nascita dei primi metodi di *Apprendimento per dimostrazione* [5] [11] [44] [45] , l'idea di insegnare a un robot un nuovo compito, tramite una dimostrazione diretta di esso nasce appunto nella speranza di minimizzare o eliminare la programmazione esplicita del robot da parte di un esperto, orientandosi quindi un interfaccia meno formale e più sociale con l'obbiettivo futuro di semplificare l'interazione con queste macchine, anche per persone esterne a questo campo di studio.

L'apprendimento per dimostrazione ricade nell'ambito dell'apprendimento automatico, visto come un caso particolare del paradigma di apprendimento supervisionato, risulta però un approccio unico nel suo genere, dove la presentazione in input di una serie di dimostrazioni, velocizza e guida passo passo il processo di apprendimento del robot. Questo approccio è spesso affiancato da altri algoritmi di apprendimento, affiancamento necessario per sopperire agli attuali limiti dell'apprendimento per dimostrazione.

2.1.1 Cenni storici

L'apprendimento per dimostrazione è nato all'inizio degli anni '80 nell'ambito della robotica industriale con l'obbiettivo di trovare un'alternativa al complesso e lungo processo di programmazione necessario per i robot utilizzati nelle industrie. Inizialmente, i principali progressi in questo campo furono ispirati dai primi lavori di Intelligenza Artificiale del tempo, basati, quindi, su un approccio simbolico. La dimostrazione veniva effettuata dall'esperto, di solito tramite teleoperazione del robot e il compito era suddiviso in azioni primitive che venivano registrate codificando le relazioni spaziali fra il robot e l'ambiente

e successivamente elaborate tramite processi computazionali Lozano-Perez(1982) [12], Dufay e Latombe (1983)[29] , Segre e DeJong (1985) [48].

Pertanto le prime soluzioni di apprendimento per dimostrazione si potrebbero formalizzare come una macchina a stati finiti basata su un'insieme di blocchi *if-then*. Quando uno degli stati esplorati durante la dimostrazione si ripresentava, il processo rieseguiva i movimenti effettuati dall'esperto in modo autonomo. Risulta evidente il primo limite a questo approccio: se l'ambiente esterno cambiava o si presentavano stati che non erano previsti durante la dimostrazione, il robot non sapeva come reagire, costringendo quindi l'esperto a considerare più stati possibili durante la dimostrazione. La soluzione a questo problema avvenne a partire dagli anni '90; si era, infatti, pensato di affiancare all'apprendimento per dimostrazione anche tecniche di apprendimento automatico per generalizzare le azioni apprese dal robot e poterle applicare in contesti più vasti e.g. Ikeuchi(1993) [21] e Dillmann, Kaiser e Ude (1995) [15].

Contemporaneamente a questi studi, furono effettuate le prime ricerche sull'imitazione di comportamenti e abilità umane esplorate inizialmente da Kawato, Gandolfo, Gomi e Wada (1994) [22] e due anni dopo Miyamoto (1996) [37], insegnò a un braccio meccanico antropomorfo compiti complessi tramite l'apprendimento per dimostrazione.

L'interesse per l'apprendimento per dimostrazione è aumentato notevolmente negli ultimi vent'anni, tanto da diventare un terreno di studio interdisciplinare che interessa non solo la robotica, ma anche la psicologia e le neuroscienze. Attualmente l'apprendimento per dimostrazione risulta l'approccio più semplice e intuivo come interfaccia uomo-robot, poiché minimizza la programmazione esplicita da parte di un esperto e apre le porte alla diffusione dei robot come parte integrante della società. Per questi e altri motivi, l'apprendimento per dimostrazione è tutt'ora argomento di discussione nelle maggiori conferenze di robotica.

2.2 Definizione del problema

2.2.1 Analisi del problema

L'apprendimento per dimostrazione può essere visto come un caso particolare di apprendimento supervisionato, l'insegnante o supervisore fornisce sia l'input che il relativo output desiderato che è il concetto fondamentale del paradigma di apprendimento supervisionato. La differenza da un problema classico di apprendimento con supervisione consiste, però, nel fatto che, tramite la dimostrazione, viene fornito al robot anche la strategia per la risoluzione del problema. In pratica il robot imita il comportamento del supervisore, motivo per cui l'apprendimento per dimostrazione è anche chiamato *apprendimento per imitazione*.

Nel 2001 Nehavin e Dautenhahn formalizzarono i problemi da affrontare nell'apprendimento per dimostrazione con una serie di domande chiave [39] [40] :

- **Cosa Imitare?**
- **Come Imitare?**
- **Quando Imitare?**
- **Chi imitare?**

Attualmente solo le prime due domande sono considerate fondamentali nell'apprendimento per imitazione, in quanto per le altre non sono stati ancora definiti riscontri pratici interessanti.

Cosa Imitare?

Il primo problema dell'imitazione è sicuramente quello di analizzare quali aspetti della dimostrazione devono essere presi in considerazione, questo problema ha connessioni con la psicologia e in particolare nei metodi di apprendimento dei bambini (Gergely et al. 2002, Carpenter et al. 2002), fatto che dimostra ancora una volta come l'apprendimento per dimostrazione trova ispirazione in diversi e svariati campi di studio. Determinare ciò che è rilevante e cosa non lo è, dipende dal supervisore che, durante le dimostrazioni, deve evidenziare gli aspetti

importanti da imitare e tralasciare quelli secondari o non interessanti. Una cattiva soluzione a questo problema, potrebbe causare tempi di apprendimento molto lunghi o addirittura l'apprendimento di strategie ambigue che potrebbero dare risultati indesiderati in particolari situazioni o stati. Limitare invece il problema agli aspetti fondamentali, permette una convergenza dell'apprendimento più rapida e meno soggetta al rumore.

Come Imitare?

Una volta deciso cosa è importante e rilevante imitare, risulta fondamentale il modo in cui viene effettuata l'imitazione. Anche se l'apprendimento per dimostrazione offre i maggiori vantaggi se applicato a robot umanoidi, non è da escludere il suo utilizzo anche con altri robot. Risulta, quindi, necessario da parte del supervisore determinare un comportamento equivalente che porti allo stesso risultato, ad esempio: se la dimostrazione consiste nel colpire un pallone con un calcio, un robot non umanoide potrebbe considerare equivalente colpire il pallone con il corpo o con una ruota.

Mentre il problema del Cosa imitare trova delle similitudini con la psicologia, il Come imitare trova una correlazione molto forte col problema delle corrispondenze (Nehavin 2007) [2], pertanto le differenze fisiche e di percezione fra uomo e robot devono essere prese seriamente in considerazione all'atto dell'imitazione. Sottovalutare questo problema, potrebbe portare all'apprendimento di comportamenti anomali rispetto alla dimostrazione effettuata e all'output desiderato.

2.2.2 Definizione generale del problema

Un problema generale di Apprendimento per Dimostrazione può essere formalizzato come segue: sia S l'insieme degli stati, sia A l'insieme delle azioni, sia Z l'insieme degli stati di interesse rispetto al totale S, risulta essere quindi

$$Z \subset S$$

Il problema fondamentale dell'apprendimento consiste nel determinare una strategia π, tale che:

$$\pi : Z \to A$$

Ovvero, la serie di dimostrazioni serve a insegnare al robot il comportamento da replicare quando percepisce un particolare stato Zn. Di solito la strategia π è funzione di diversi parametri,

$$\pi = \pi(z(t), \alpha)$$

Dove:

- $z(t)$ è lo stato attuale in funzione del tempo
- α è il vettore di parametri che devono essere impostati dall'apprendimento.

La strategia viene appresa grazie alle dimostrazioni, per ogni dimostrazione $d_i \in D$ insieme delle dimostrazioni, viene definita come la coppia $d_i = (z_i^j, a_i^j)$ con $j = (1, 2, ..., k)$. Ovvero ogni dimostrazione o esempio determina, similmente al paradigma di apprendimento supervisionato, una coppia *Input:Stato Output:Azione*.

2.3 Soluzioni al problema

Una volta analizzato e formalizzato il problema generale, in particolare dopo aver scelto attentamente quali sono gli aspetti principali della dimostrazione è necessario, considerando il problema nella sua totalità, scegliere quale processo di imitazione risulta più efficiente per la risoluzione del problema. Risulta efficace presentare queste tecniche riprendendo la suddivisione del problema nelle due domande fondamentali presentate da Nehavin e Dautenhahn.

2.3.1 Cosa imitare

Le possibili soluzioni al problema *Cosa Imitare?*, possono essere suddivise in due grandi gruppi:

- metodi che sfruttano la riproduzione di **micromovimenti**.
- metodi che si focalizzano sull'apprendimento di **comportamenti elementari**.

Questa scelta determina anche la futura codifica degli stati Z e delle azioni A e di conseguenza gli algoritmi di apprendimento che possono essere utilizzati successivamente.

Micromovimento

L'approccio a micromovimenti fa riferimento a uno dei paradigmi fondamentali dell'informatica: *Divide et Impera*.

La dimostrazione, in pratica, viene suddivisa in micromovimenti o azioni, le quali possono essere dimostrate singolarmente e imparate dal robot alunno. Risulta fondamentale con questo approccio un buona codifica del singolo micromovimento, per esempio la maggior parte dei lavori che utilizzano questo approccio, codificano le singole azioni nello spazio dei giunti del dimostratore, ovvero vengono memorizzate le traiettorie, la posizione e gli angoli formati dai giunti dell'utente che effettua la dimostrazione.

Questo metodo risulta assai conveniente quando i gradi di libertà dei giunti del robot alunno corrispondono agli stessi gradi di libertà dell'utente dimostratore, come ad esempio in un robot antropomorfo. Esempi di applicazione di questo metodo possono essere trovati prima nei lavori di Stefan Schaal [6] e poi in quelli di Ijspeer [47].

Questo approccio però introduce anche altre problematiche, che non sono da sottovalutare:

- la codifica del micromovimento è di solito effettuata in uno spazio a cardinalità troppo elevata, quindi per ridurre il carico computazionale dell'apprendimento, risulta spesso necessario l'applicazione di tecniche di riduzione della dimensionalità dello spazio, proiettando quindi la dimostrazione effettuata in uno spazio di moto fittizio a cardinalità ridotta, per maggiori informazioni su queste strategie si fa riferimento a [56];

- la riproduzione ripetuta dei micromovimenti, sopratutto da parte di un utente umano, è soggetta a cambiamenti involontari da dimostrazione a dimostrazione, causando perciò ambiguità e contrasti negli esempi di apprendimento. Una possibile soluzione a questo problema è l'utilizzo di tecniche di apprendimento statistico o di codifiche basate su equazioni differenziali [47].

Comportamenti elementari

Questo approccio consiste nell'apprendere comportamenti complessi formati da una serie di comportamenti elementari che possono essere

già presenti e codificati nel sistema, oppure similmente all'approccio per micromovimenti, possono essere segmentati dal flusso di dati principale durante la fase di dimostrazione, ovvero si suppone che la dimostrazione dell'utente venga mappata in un insieme di primitive motorie apprese in una fase preliminare, per poi essere riprodotte nella sequenza desiderata a seguito di un particolare stato esterno. Un altro utilizzo a questo approccio è l'apprendimento di comportamenti elementari tramite dimostrazione che poi vengono utilizzati per formare azioni complesse come, per esempio, una semplice sequenza esatta di queste azioni come nei lavori di Zhang and Weng [63] e Dominay [16].

Queste soluzioni risultano però poco interessanti poiché si basano sull'apprendimento di comportamenti elementari che vengono ripetuti in sequenza alla ricezione di particolari comandi, quali ad esempio quelli vocali utilizzati nel lavoro di Dominay.

Un approccio molto recente è invece offerto da Muelling (2013) [38], che, tramite algoritmi basati sull'apprendimento per rinforzo, combina le azioni elementari apprese in comportamenti più complessi. Questa soluzione risulta più solida delle precedenti poiché , tramite l'apprendimento per rinforzo, il robot può eseguire l'azione complessa anche in presenza di stati esterni simili ma non esplorati durante l'apprendimento.

2.3.2 Come imitare

Anche per quanto riguarda la scelta su come effettuare le dimostrazioni è impossibile determinare a priori un metodo che sia migliore degli altri. Esistono sostanzialmente due grandi categorie, ognuna con i suoi vantaggi o svantaggi in particolari problemi di apprendimento per dimostrazione. Queste categorie sono:

- muovere direttamente il corpo del robot
- osservare il comportamento del supervisore

Muovere il corpo del robot

Questo metodo consiste nel muovere direttamente il corpo del robot che deve essere in grado di memorizzare i movimenti effettuati durante

la dimostrazione per poterli ripetere successivamente, risulta per questo motivo uno degli approcci naturali se affiancato alla suddivisione della dimostrazione in micromovimenti. Il corpo del robot può essere mosso in due modalità principali:

- **Kinesthetic Teaching**: una delle tecniche maggiormente utilizzate e tutt'ora di grande interesse è costituita dal Kinesthetic Teaching, la dimostrazione viene effettuata controllando passivamente i giunti del robot muovendoli manualmente nelle configurazioni desiderate. L'apprendimento con questo metodo può essere continuo, ovvero vengono registrate tutte le traiettorie dei giunti dall'inizio alla fine della dimostrazione, oppure quest'ultima può essere eseguita come una registrazione discreta nel tempo di una successione di pose fondamentali che il robot deve imparare [1].

  Il kinesthetic Teaching però non è perfetto: robot con troppi gradi di libertà potrebbero essere troppo complessi da manovrare correttamente. Le dimostrazioni possono offrire informazioni solo sulle traiettorie dei movimenti, ma nessuna informazione viene data sulla forza da imprimere a essi, questo risulta un grande svantaggio nel caso in cui il robot debba imparare a maneggiare oggetti fragili. Inoltre questo non è assolutamente conveniente se applicato a robot molto grossi e pesanti, in questo caso infatti il supervisore dovrebbe affrontare non pochi problemi nel tentativo di muovere il corpo del robot.

- **Teleoperation**: La tecnica di teleoperazione è stata una delle prime utilizzate nella storia dell'apprendimento per dimostrazione, risulta molto simile alla Kinesthetic Teaching, tranne per il fatto che il robot viene comandato tramite un interfaccia che può essere ad esempio un joystick o un'apposita tastiera. Rispetto alla Kinesthetic Teaching questa tecnica offre diversi vantaggi, tra cui: la possibilità di eseguire la dimostrazione a distanza, per cui la grandezza o la pesantezza del corpo del robot non è più un problema se questo viene comandato da un'interfaccia con la quale è inoltre possibile trasferire altre informazioni durante la dimostrazione come ad esempio la forza da imprimere durante l'esecuzione dell'operazione. Di grande interesse

è il lavoro sviluppato da Peternel e Babic (2013)[41] e Babic et al.(2011) [7], i quali insegnarono a un robot umanoide alcune tecniche di equilibrio tramite la dimostrazione da parte di un supervisore umano. Il dimostratore indossava un interfaccia aptica collegata al dorso che inviava al robot informazioni sulla postura da tenere per mantenere l'equilibrio anche a seguito di una perturbazione ricevuta. Soffre però dello stesso problema della Kinesthetic Teaching, poiché un robot con molti gradi di libertà sarà difficilmente manovrabile tramite un'interfaccia.

Osservare il comportamento del dimostratore

In questo caso la dimostrazione non è per forza eseguita da un supervisore umano, ma si può osservare il comportamento di un altro robot. È molto importante risolvere anticipatamente il problema delle corrispondenze, ovvero effettuare una trasformazione tra lo spazio del supervisore e quello dell'alunno, che permetta un corretto trasferimento di informazioni.

In pratica le soluzioni possibili sono due:

- viene collocata sul corpo del dimostratore una serie di sensori che raccolgono delle precise informazioni sul compito da imitare, metodo che risulta particolarmente efficace per la qualità dell'informazione che viene trasferita durante la dimostrazione. Risulta, inoltre, un approccio naturale se utilizzato per l'apprendimento di un robot umanoide. L'unico grande svantaggio è che l'equipaggiamento tecnologico necessario è spesso estremamente costoso, in quanto vengono utilizzati sensori specialistici e strutture apposite;

- il robot può osservare il comportamento dimostrato direttamente grazie all'ausilio di una telecamera, in questo caso i dati raccolti sono spesso inaffidabili e potrebbero portare a un errato apprendimento da parte del robot. Rispetto all'utilizzo di sensori e apparecchiature costose risulta, però, un approccio molto più economico.

2.3.3 Apprendimento per dimostrazioni errate

L'approccio naturale e classico dell'apprendimento per dimostrazione consiste nel fornire al robot alunno esempi corretti del compito da imitare, ma ciò non è sempre possibile, infatti:

- il tempo necessario per la raccolta di un numero sufficiente di dimostrazioni corrette non è trascurabile e dipende direttamente dalla difficoltà del compito e dall'abilità del supervisore che ha effettuato la dimostrazione;

- il comportamento da imitare non è semplice e per alcune dimostrazioni occorre una conoscenza approfondita del compito d'apprendere e potrebbe richiedere costi aggiuntivi per la definizione del modello o per la consulenza di esperti del settore;

- alcuni comportamenti, seppure teoricamente semplici, potrebbero non essere replicati facilmente da un utente umano, come ad esempio nel caso di torsioni dei giunti impossibili all'uomo o il sollevare carichi troppo pesanti. Inoltre durante l'esecuzione della dimostrazione gli errori sono sempre possibili e con l'approccio classico non solo rallentano l'apprendimento ma potrebbero addirittura comprometterlo.

Però una dimostrazione sbagliata contiene comunque delle informazioni che possono essere utilizzate per velocizzare l'apprendimento del compito. L'apprendimento per dimostrazioni errate introdotto dalle ricerche di Billard e Grollman [19] [18]. consente un approccio leggermente diverso da quello classico, perché si tiene conto dell'inevitabile possibilità che un supervisore umano possa commettere degli errori in buona fede durante il processo di dimostrazione. Questi esempi errati, che partono comunque da una base corretta, possono essere utilizzati dal robot alunno per ampliare la propria esperienza:

- le dimostrazioni errate sono esempi di cosa non deve essere fatto, quindi comportamenti da evitare;

- le dimostrazioni errate contengono comunque indicazioni su come deve essere interpretato correttamente il compito;

- un insieme di dimostrazioni errate serve a delimitare uno spazio di esplorazione in cui ricercare il corretto modello del comportamento da imitare.

I lavori prima citati di Billard e Grollman utilizzano un appropriato algoritmo di apprendimento automatico quale il *DMM* o il *GMM* che permettono di sfruttare, durante il processo di apprendimento, anche le dimostrazioni errate.

2.4 Limiti dell'apprendimento per dimostrazione

I progetti di Billard e Grollman tentano, con un approccio diverso da quello classico, di ridurre alcuni limiti dell'apprendimento per dimostrazione, che consiste proprio nell'esecuzione del compito di interesse.

Infatti l'apprendimento è strettamente collegato alla qualità delle informazioni fornite durante la dimostrazione e quindi lo sviluppo di una strategia efficace dipende direttamente dalla quantità e dalla qualità delle informazioni raccolte tramite la serie di dimostrazioni fornite, le quali a loro volta dipendono dall'abilità dell'insegnante che esegue la dimostrazione.

Alcuni problemi non possono essere affrontati efficacemente dall'apprendimento per dimostrazione, come ad esempio i comportamenti difficilmente ripetibili o quelli fisicamente non realizzabili da supervisori umani.

Le dimostrazioni fornite per lo stesso compito possono essere diverse da esempio a esempio e quindi generare ambiguità durante l'apprendimento, oppure può esistere più di un comportamento corretto rispetto a un unico stato e anche questo può generare ambiguità o rallentamento nel processo di apprendimento. Inoltre l'apprendimento per dimostrazione soffre come tutte le tecniche di apprendimento automatico del problema della generalizzazione, non è possibile: effettuare una dimostrazione per ogni stato e quindi, in presenza di stati che non facevano parte dell'insieme degli esempi proposti, il robot potrebbe non riconoscere nessuna azione corretta come risposta allo stato. Risulta, pertanto, evidente che allo stato attuale dell'arte il solo

apprendimento per dimostrazione non è sempre sufficiente per ricavare i comportamenti desiderati, risulta necessario nella maggior parte dei casi la suddivisione del problema in due parti fondamentali:

1. L'apprendimento preliminare di un comportamento da replicare tramite tecniche di apprendimento per dimostrazione;

2. Una volta terminato il punto precedente, generalizzare o migliorare l'esecuzione del compito grazie ad algoritmi e tecniche di apprendimento automatico, in particolare è obbiettivo di questa tesi presentare le ricerche di Kormashev il quale utilizza, con ottimi risultati, tecniche di apprendimento per rinforzo per migliorare e generalizzare i comportamenti complessi appresi precedentemente tramite apprendimento per dimostrazione [26].

Capitolo 3

Apprendimento Robotico

3.1 Panoramica sullo stato dell'arte

L'apprendimento robotico rappresenta l'anello di congiunzione fra la robotica e l'intelligenza artificiale; è il campo che studia e sviluppa nuove tecniche per fornire ad un robot delle nuove abilità.

In particolare, dotare i robot di capacità di interazione simili all'uomo è uno degli obbiettivi principali della robotica.L'apprendimento robotico rappresenta, tutt'ora, uno dei metodi più interessanti ed efficaci per raggiungere questo obbiettivo. L'apprendimento robotico affronta questo problema fornendo, per quanto possibile, al robot una capacità di apprendimento simile all'uomo. L'obbiettivo finale di questo approccio consiste nel dotare i robot della capacità di: apprendere, migliorare, adattare e riprodurre abilità con vincoli che cambiano dinamicamente. I paradigmi presentati, nei primi due capitoli, insieme alla *programmazione diretta* rimangono gli approcci e i metodi tutt'ora più utilizzati nell'apprendimento robotico. In particolare è interessante notare come sia impossibile definire, a priori, una tecnica migliore delle altre. Ogni paradigma presenta caratteristiche uniche rendendolo più o meno efficacie nella risoluzione di un particolare problema.

Di seguito si vogliono riassumere e schematizzare le caratteristiche principali di ogni particolare approccio [28] :

Programmazione diretta

Programmazione classica di un robot, viene creato un algoritmo che dice al robot come deve comportarsi e muoversi.

- *Difficoltà per il supervisore*: Anche se non esiste un vero e proprio supervisore, poiché, non è un vero e proprio metodo di apprendimento. Si identifica il supervisore con la figura del programmatore.

- *Complessità computazionale*: Semplice, muovere un robot con la programmazione diretta è semplice e preciso, però richiede una formalizzazione dell'ambiente complessa e spesso impossibile.

- *Vantaggi*: Si ha il completo controllo dei movimenti del robot grazie alla programmazione di basso livello; permette quindi una grande precisione.

- *Svantaggi*: La programmazione diretta di un robot richiede tempo per formulare e testare l'algoritmo. Le soluzione apprese, inoltre, risultano non scalabili e non riusabili.

- *Possibili utilizzi*: Benché la programmazione diretta sia la tecnica meno conveniente, trova ancora utilizzo nella programmazione di robot industriali dove è richiesta una grande precisione e l'ambiente stazionario e strutturato riduce al minimo gli svantaggi di questo paradigma.

Apprendimento per dimostrazione

L'apprendimento per dimostrazione consiste nel presentare al robot il comportamento da eseguire attraverso le tecniche di teleoperazione, kinesthetic teaching e observational learning. Per informazioni più dettagliate, sulle tecniche e i rispettivi vantaggi e svantaggi, si faccia riferimento al capitolo precedente.

- *Difficoltà per il supervisore*: Variabile, dipende esclusivamente dall'abilità da riprodurre.

- *Complessità computazionale*: Bassa, il processo di imitazione consiste nella maggior parte dei casi della registrazione del movimento e della sua esecuzione.
- *Vantaggi*: Per l'apprendimento della capacità non è necessaria la programmazione diretta di essa. L'apprendimento è affrontato in modo semplice e naturale, particolarmente adatto per i robot umanoidi.
- *Svantaggi*: La qualità dell'apprendimento dipende direttamente dalla qualità delle dimostrazioni e dall'abilità del dimostratore di presentarle.
- *Possibili utilizzi*: L'apprendimento per dimostrazione risulta particolarmente efficace nell'apprendimento di azioni complesse che non richiedono grande adattabilità con l'ambiente esterno.

Apprendimento per rinforzo

Viene definita una funzione di ricompensa che fornisce al robot un voto in base all'azione intrapresa, l'obbiettivo del robot consiste nel massimizzare le ricompense ottenute.

- *Difficoltà per il supervisore*: Minima, in fase di apprendimento il supervisore si limita a fornire una ricompensa per l'azione eseguita.
- *Complessità computazionale*: Elevata, l'algoritmo di apprendimento deve stimare e approssimare: una serie di politiche di comportamento e una funzione per poter valutare la politica ottima.
- *Vantaggi*: Non è necessario mostrare o conoscere nel dettaglio l'abilità che deve essere appresa; il robot stesso esplora l'ambiente nella ricerca di una soluzione. Per questo motivo le abilità apprese con questa tecnica risultano le più adattabili a situazioni nuove e sconosciute.
- *Svantaggi*: Processo di apprendimento piuttosto lungo, assenza di controllo diretto sulle azioni del robot infine la soluzione è

basata su funzioni e parametri stimati, quindi soffre di un certo grado di incertezza.

- *Possibili utilizzi*: L'apprendimento per rinforzo fornendo capacità molto adattabili è usato in molto applicazioni robotiche, sopratutto nei problemi di interazione con un ambiente dinamico.

Goal-directed learning

Il goal-directed learning [28] è un approccio ancora teorico, viene specificato al robot solo l'obbiettivo da raggiungere senza valutare i processi intermedi come, invece, accade nell'apprendimento per rinforzo.

- *Difficoltà per il supervisore*: Minima, il supervisore deve conoscere l'obbiettivo da raggiungere e formalizzarlo in modo efficiente.

- *Complessità computazionale*: Massima, l'apprendimento non viene valutato da azioni intermedie, lasciando quasi la totale libertà di azione al robot che conoscendo solo l'obbiettivo da raggiungere deve estrapolare la politica più efficiente.

- *Vantaggi*: Teoricamente è la tecnica che fornisce i maggiori vantaggi in fatto di adattabilità e semplicità nella specifica del problema.

- *Svantaggi*: Non si ha alcun controllo sui movimenti e sul processo di apprendimento del robot; l'obbiettivo da raggiungere deve essere conosciuto e ben formalizzato.

- *Possibili utilizzi*: Come approccio teorico non è ancora stato utilizzato in contesti pratici, anche se le similitudini con l'apprendimento per rinforzo fanno immaginare applicazioni in contesti simili.

Dal riassunto sopra esposto è chiaro come non sia possibile definire una tecnica migliore delle altre, in particolare è interessante notare come i punti deboli di una siano i punti forti dell'altra. Si può supporre dunque, che per ottenere i risultati migliori, le varie tecniche

di apprendimento robotico devono coesistere nel raggiungimento della soluzione.

In particolare è interessante notare come l'apprendimento per rinforzo e l'apprendimento per dimostrazione possano cooperare per ottenere una soluzione migliore. Per esempio:

- l'apprendimento per rinforzo può essere utilizzato per adattare una capacità verso nuove situazioni;
- è possibile partire da una dimostrazione abbastanza buona e gradualmente migliorarla.

3.2 Apprendimento per rinforzo nella robotica

Nel capitolo 1 di questo elaborato è stata presentata la teoria che sta dietro all'apprendimento per rinforzo e alla fine del capitolo sono stati presentati i limiti di questa tecnica.

In particolare la *curse of dimensionality* ovvero l'impossibilità di gestire efficacemente il problema all'aumento delle sue dimensioni. I sistemi robotici sono, naturalmente, sistemi di una grande dimensionalità e complessità: hanno diversi gradi di libertà, presentano un interazione continua con l'ambiente e sono largamente soggetti al rumore. Per questo motivo, i metodi classici detti anche *temporal-difference* non hanno trovato grandi applicazioni nella robotica, dove soffrirebbero troppo della enorme dimensione dei problemi.

I primi successi dell'applicazione di algoritmi di rinforzo alla robotica è avvenuta con le prime tecniche di approssimazione di funzione, ma il vero rilancio è avvenuto con la nascita dei primi metodi di *policy-search*. Nei quali, invece che operare su un enorme spazio di stato-azione, viene definito uno spazio più piccolo, dove ogni singola politica può essere rappresentata da una particolare policy parametrizzata.

La dimensionalità dello spazio è così drasticamente ridotta, permettendo le prime efficaci applicazioni di questo paradigma nella robotica, per esempi si faccia riferimento agli algortimi di punta *eNAC* [42] e *REINFORCE* [62]. Sfortunatamente i metodi basati sul policy-

search soffrono di una grande dipendenza del rateo di apprendimento (*learning rate*) e dalle varianze nell'esplorazione.

Un approccio alternativo è invece offerto dagli algoritmi di *Expection-Maximization* o *EM*, in particolare l'algoritmo PoWER (*policy learning by weighting exploration with returns*) [25]. Il quale ha dimostrato performance superiori se utilizzato per migliorare l'efficacia di abilità precedentemente apprese, ad esempio con la dimostrazione.

3.2.1 Caratteristiche fondamentali della *policy*

La rappresentazione o codifica della policy è un passo fondamentale per la risoluzione dei problemi di apprendimento per rinforzo.

In particolare, nei metodi che sfruttano il policy-search, la parametrizzazione dello spazio delle politiche influenzerà direttamente il processo di apprendimento, determinando: la velocità di convergenza e la varianza delle politiche generabili.

Risulta quindi che la codifica di una buona politica è fondamentale, nella robotica le seguenti caratteristiche sono fondamentali, definite da Kormushev sempre in [28]:

- **smoothness**: l'abilità appresa deve poter essere appresa in modo scorrevole con traiettorie continue, ovvero senza accelerazioni o decelerazioni repentine, sia per preservare il robot da guasti sia per ridurre i consumi di energia;

- **safety**: la policy appresa deve essere sicura sia per il robot, in termini di sforzi sui giunti, sia per l'ambiente in cui agisce;

- **gradual exploration**: la politica deve evolversi e cambiare gradualmente seguendo una esplorazione dell'ambiente incrementale;

- **scalability**: la politica deve essere scalabile ovvero adattarsi anche a problemi di dimensioni e complessità maggiori;

- **compactness**: seppure deve essere in grado di affrontare problemi con più di 50 gradi di libertà, la politica deve avere una codifica compatta che ovviamente usi molti meno parametri;

- **adaptability**: la politica si deve adattare alla complessità o alle caratteristiche del compito;
- **multi resolution**: parti differenti della parametrizzazione della policy devono permettere soluzioni o precisione differente;
- **unbiasedness**: la politica deve funzionare anche senza una particolare conoscenza preliminare del compito, senza quindi restringere l'esplorazione della soluzione;
- **prior/bias**: d'altro canto deve poter anche gestire l'utilizzo di conoscenze preliminari per guidare il processo di apprendimento;
- **regularization**: la politica deve permettere un meccanismo di regolarizzazione per guidare l'esplorazione verso il tipo di politica desiderata;
- **time indipendece**: la politica non dovrebbe dipendere dal tempo o dalla posizione dell'agente nell'ambiente, al fine di poter affrontare anche situazioni impreviste;
- **embodiment-agnostic**: la politica deve essere indipendente dalle caratteristiche tecniche del robot, come ad esempio le traiettorie dei giunti;
- **invariance**: la politica deve essere indipendente dal modello del compito da apprendere;
- **correlations**: la parametrizzazione della policy dovrebbe incapsulare anche le variabili di controllo, come ad esempio i segnali degli attuatori;
- **globality**: la rappresentazione della politica deve aiutare l'algoritmo di rinforzo ad evitare i minimi locali, cooperando alla convergenza verso la soluzione ottima;
- **periodicity**: deve permettere una semplice rappresentazione di movimenti ciclici o periodici, quest'ultima è una caratteristica ricorrente in molti applicazioni robotiche;

- **analyzability**: la parametrizzazione deve essere chiara e permettere un'analisi chiara della politica;
- **multi-dimensionality**: deve permettere un uso efficiente dei feedback multidimensionali senza la necessità di convertirli in valori scalari;
- **convergence**: deve aiutare il processo di apprendimento per una convergenza rapida ad una soluzione ottima.

Codificare una politica in cui coesistono tutte le precedenti caratteristiche è per ora impossibile; le attuali soluzioni prendono in considerazione, infatti, solo un ristretto sottogruppo del totale. Nella maggior parte dei casi la codifica della politica e quindi le caratteristiche implementate dipendono quasi esclusivamente dall'abilità che deve essere appresa.

3.3 L'apprendimento di nuove primitive motorie

Nella robotica l'apprendimento di nuovi movimenti o primitive motorie è un problema fondamentale. Dotare un robot di sempre più complessi movimenti permette ,infatti, un loro utilizzo in contesti nuovi o aumenta la loro efficienza in vecchi contesti.

L'apprendimento di questi movimenti è un tipico esempio di cooperazione di tecniche di apprendimento diverse.

In particolare: l'apprendimento per dimostrazione permette di replicare velocemente ed efficacemente un movimento; l'apprendimento per rinforzo permette, invece, di migliorarlo e adattarlo a nuove situazioni.

3.3.1 Modello generale delle primitive motorie

Le primitive motorie possono essere rappresentate come: due sistemi dinamici collegati in una sola direzione in modo tale che uno guidi l'altro ma non viceversa [20].

Quindi un sistema dinamico per la rappresentazione di primitive motorie è suddiviso in: un sistema canonico h che comanda un sistema g_k per ogni grado di libertà k. In pratica il compito del sistema h consiste nell'accoppiare e sincronizzare fra loro i diversi gradi di libertà, i quali rappresentano i movimenti consentiti al robot.

Per combinare diverse primitive motorie serve un sistema h indipendente per ogni primitiva, oppure un sistema congiunto se queste primitive sono fra loro dipendenti.

Come risultato abbiamo quindi un sistema di equazioni differenziali:

$$\dot{z} = h(z)$$
$$\dot{x} = g(\mathbf{x}, z, \mathbf{w})$$

dove :

- $\mathbf{x}$ è la variabile da determinare e rappresenta lo stato del giunto: posizione, velocità, accelerazione;
- z determina lo stato del sistema h
- $\mathbf{w}$ è il parametro interno per codificare l'output del sistema canonico h.

Le informazioni codificate in $\mathbf{x}$ vengono utilizzate da un apposito controllore per inviare i comandi agli attuatori del movimento.

L'implementazione dei sistemi h e g dipende da come vogliono essere rappresentate le primitive [20], [46]:

Discrete Movement Primitives

La rappresentazione discreta delle primitive consiste nel rappresentare il movimento come il raggiungimento di una particolare posa finale a partire da una data configurazione iniziale.

Per cui il sistema generale prima esposto può essere così formulato:

$$\dot{z} = h(z) = -\tau\alpha_h z$$

che rappresenta la fase del movimento, formata dalla costante di tempo τ, mentre α è un parametro scelto affinché il sistema sia stabile.

La posizione del k-esimo grado di libertà data da: $q_k = x_{2k}$, dove x_{2k} è il 2k-esimo componente di x. Di conseguenza la velocità si ricava come: $\dot{q}_k = \tau x_{2k+1} = \dot{x}_{2k}$, mentre l'accelerazione $\ddot{q}_k = \tau \dot{x}_{2k+1}$.

Il sistema di trasformazione g può essere quindi formulato:

$$\dot{x}_{2k+1} = \tau\alpha_g(\beta_g(t_k - x_{2k}) - x_{2k+1}) + \tau((t_k - x_{2k}^0) + a_k)f_k$$

$$\dot{x}_2 k = \tau x_{2k+1}$$

dove τ è la stessa costante di tempo del sistema canonico, α_g e β_g sono i parametri da impostare per la stabilità del sistema e a_k è un parametro di amplificazione; mentre x_{2k}^0 è la posizione iniziale della componente del giunto e t_k rappresenta la sua posizione finale.

Particolarmente importante risulta la funzione di trasformazione f_k che permette di trasformare gli input di controllo inviati dal sistema canonico h:

$$f_k(z) = \sum_{i=1}^{N} \psi_i(z) w_i z$$

con w il parametro di aggiustamento presente nella formulazione generale, $\psi(z)$ è la funzione Gaussiana:

$$\psi_i = \frac{\exp(-h_i(z - c_i)^2))}{\sum_{j=1}^{N} \exp(-h_j(z - c_j)^2)}$$

La funzione $\psi(z)$ determina i valori dei pesi che localizzano nello spazio la fase del movimento.

Rhythmic Movement Primitives

La rappresentazione ritmica o ciclica delle primitive permettere di rappresentare un movimento ciclico o periodico. In generale, quindi, raggiunto lo stato finale del movimento deve essere possibile rieseguirlo anche partendo dalla nuova configurazione.

Il sistema canonico quindi è formulato:

$$\dot{z} = h(z) = \tau\omega$$

con ω che rappresenta la variazione di fase e può essere utilizzata per l'accoppiamento di diversi gradi di libertà.

Mentre funzione il sistema di trasformazione ottiene la seguente forma:

$$\dot{x}_{2k+1} = \tau\alpha_g(\beta_g(x_m - x_2 k) - x_{2k+1}) + a_k f_k$$

$$\dot{x}_2 k = \tau x_{2k+1}$$

dove i parametri sono gli stessi della formulazione fatta per i movimenti discreti. Il nuovo parametro x_m rappresenta il punto iniziale dell'oscillazione.

Similmente al caso precedente la funzione di trasformazione f_k è così formulata:

$$f_k(z) = \sum_{i=1}^{N} \psi_i(z) w_i$$

con:

$$\psi_i = \frac{\exp(-h_i(1 - \cos(z - c_i)))}{\sum_{j=1}^{N} \exp(-h_j(1 - \cos(z - c_j))}$$

.

3.3.2 Metodi di apprendimento per le primitive motorie

Apprendimento per dimostrazione

Un preliminare apprendimento per dimostrazione permette di definire approssimatamente una *policy* del movimento desiderato, grazie ad una dimostrazione diretta da parte di un supervisore.

La dimostrazione serve a dare una prima approssimazione della funzione di trasformazione f_k. Questa stima prevede, quindi, un errore ϵ così definito:

$$\varepsilon_m^2 = \sum_{i=1}^{n} \psi_i^m (f_i^{ref} - z_i^T \mathbf{w}^m)^2$$

questa funzione deve essere minimizzata per tutti i vettori di parametri $\mathbf{w}^m$, con $m \in \{1, 2, ..., M\}$. Come espresso sopra: ψ_i^m è la funzione dei pesi corrispondente.

Infine, f_i^{ref} rappresenta la funzione di trasformazione desiderata e $i \in 1, 2, ..., n$ indica l'i-esima ripetizione della dimostrazione.

La funzione di errore quadratico prima esposta può essere riscritta come:

$$\varepsilon_m^2 = (\mathbf{f}^{ref} - \mathbf{Z}\mathbf{w}^m)\ \Psi\ (\mathbf{f}^{ref} - \mathbf{Z}\mathbf{w}^m)$$

Da cui si può quindi ricavare il valore $\mathbf{w}^m$:

$$\mathbf{w}^m = (\mathbf{Z}^T \Psi \mathbf{Z})^{-1} \Psi^T \mathbf{Z} \mathbf{f}^{ref}$$

dove:

- $\mathbf{f}^{ref}$ prende il valore di f_i^{ref} per ogni esempio i
- $\Psi = \text{diag}(\psi_i^m, ..., \psi_n^m)$
- $\mathbf{Z} = z_i^T$

Quanti più parametri $\mathbf{w}$ sono utilizzati e definiti, tanti più sono i dettagli che descrivono la primitiva motoria.

Per informazioni maggiori all'approccio generale si faccia riferimento sempre a [20].

Apprendimento per rinforzo

Lo scopo dell'apprendimento per rinforzo in questo genere di problema è il miglioramento della primitiva, partendo da una base accettabile stimata attraverso la fase di dimostrazione.

Viene in questa sezione presentato uno degli ultimi algoritmi sviluppati per l'apprendimento di primitive motorie, l'algoritmo PoWER [25].

L'azione $a = \mathbf{f}(\mathbf{z}) + \hat{\epsilon}$ è l'output della primitiva mentre la variabile ϵ rappresenta il fattore esplorazione durante l'apprendimento.

In particolare, l'obbiettivo dell'apprendimento per rinforzo consiste nell'imparare un politica deterministica:

$$\bar{a} = \theta^T \mu(s) = \mathbf{f}(\mathbf{z})$$

dove $\theta = [\mathbf{w}_i \in \Re^N]$

Si può definire la politica di esplorazione come:

$$\mathbf{a} = \theta^T \mu(s,t) + \epsilon(\mu(s,t))$$

l'esplorazione può essere espressa come:

$$\epsilon(\mu(s,t)) = \varepsilon_t^T$$

con $[\varepsilon_t]_{ij} \approx N(0, \rho_{ij}^2)$, con ρ_{ij}^2 meta-parametro di esplorazione che deve essere ottimizzato durante il processo di apprendimento.

La regola di apprendimento per PoWER risulta quindi:

$$\theta' = \theta + \frac{E\{\sum_{t=1}^{T} \varepsilon_t Q^\pi(s_t, a_t, t)\}}{E\{\sum_{t=1}^{T} Q^\pi(s_t, a_t, t)\}}$$

dove

$$Q^\pi(s,a,t) = E\{\sum_{\tilde{t}=t}^{T} r(s_{\tilde{t}}, a_{\tilde{t}}, s_{\tilde{t}+1}, \tilde{t}) | s_t = s, a_t = a\}$$

che rappresenta la funzione *state-action value*.

I meta-parametri di esplorazione ρ_{ij}^2 determinano il comportamento esplorativo dell'agente. Valori elevati coincidono con un comportamento molto esplorativo che causa continui e veloci cambiamenti nella policy.

Il valore dei meta-parametri decresce durante il processo di apprendimento, permettendo così la convergenza verso la politica ottima.

La cooperazione del recente algoritmo PoWER insieme a tecniche di apprendimento per dimostrazione può essere visto nei lavori di [25], [27].

Conclusioni e possibili sviluppi

Nei primi due capitoli sono stati presentati, in modo teorico e generale, due particolari paradigmi di apprendimento: l'apprendimento per dimostrazione e l'apprendimento per rinforzo. Nel terzo capitolo, invece, l'attenzione si è concentrata sulle applicazioni e i problemi da affrontare quando si utilizzano, in campo robotico, le tecniche esposte precedentemente.

In particolare è interessante notare come i paradigmi presentati abbiano caratteristiche complementari: l'apprendimento per dimostrazione si rivela molto efficiente e veloce nell'apprendimento preliminare di movimenti che richiedono grande coordinazione fra i giunti del robot. Il comportamento appreso risulta però poco generalizzabile, perciò poco adattabile a nuove situazioni; inoltre, la qualità del comportamento appreso è estremamente dipendente dalla qualità della dimostrazione.

L'apprendimento per rinforzo, invece, si rivela il più adatto per l'apprendimento di specifiche interazioni con l'ambiente esterno. In particolare, il processo di apprendimento permette di generalizzare il comportamento e, di conseguenza,reagire a situazioni sconosciute, se queste presentano somiglianze con quelle già affrontate. L'apprendimento per rinforzo risulta però molto lento nel processo di apprendimento e non particolarmente adatto all'apprendimento di primitive motorie complesse.

Infine, la programmazione diretta risulta il metodo di apprendimento meno utilizzato; la grande precisione offerta da questo metodo, infatti, non compensa la lunga lista di limitazioni che non possono essere trascurate, fra le quali la scarsa o nulla generalizzazione del com-

pito che richiede una programmazione diretta da parte di un'utente umano.

Analizzando i pro e i contro dei paradigmi presentati, si nota come le qualità dell'uno siano i difetti dell'altro e viceversa. Risulta, quindi, evidente che una loro applicazione congiunta è assai performante per l'apprendimento di nuove capacità, in particolare di primitive motorie. L'apprendimento per dimostrazione viene utilizzato, in una fase preliminare, per l'apprendimento della primitiva, mentre, l'apprendimento per rinforzo è utilizzato per generalizzare e migliorare l'esecuzione della stessa.

Questo processo di apprendimento è molto simile al processo di apprendimento dell'essere umano: l'alunno osserva il comportamento del maestro e ne impara le basi (apprendimento per dimostrazione) e continua a provarlo finché non riesce a replicarlo efficacemente (apprendimento per rinforzo).

La creazione di robot sempre più simili all'uomo, sia nei metodi di apprendimento che nel comportamento, è, ancora oggi, la vetta da raggiungere prima della loro diffusione nella società.

I paradigmi presentati sono sulla buona strada per il raggiungimento di questo obbiettivo? È difficile dirlo: la robotica e, in particolare, l'apprendimento robotico sono dei campi di ricerca giovani e in continua evoluzione. Definire con chiarezza il loro futuro è un'impresa non facile; l'apprendimento per dimostrazione non offre solo una tecnica per l'apprendimento di nuovi movimenti, ma attualmente offre anche un'interfaccia di interazione uomo-macchina innovativa che sarà, molto probabilmente, alla base delle future interfacce uomo-robot.

Il futuro dell'apprendimento per rinforzo, invece, è strettamente legato alla rappresentazione della policy, in particolare le caratteristiche di: *correlations*, *adaptability*, *multi-resolution*, *globality*, *multi-dimensionality* e *convergence* sono difficilmente implementabili, ma una loro implementazione efficiente aprirebbe alla robotica nuove e numerose possibilità.

In particolare, implementare una policy trasferibile, espandibile e indipendente dalle caratteristiche tecniche del robot, aprirebbe le porte all'apprendimento comunitario da parte di un gruppo di robot che potrebbero,perciò, scambiarsi politiche e informazioni per apprendere nuove capacità o migliorare quelle già esistenti. Un progetto da poco

inaugurato presenta la stessa idea: *RoboEarth* [57], in cui i robot possono scambiarsi informazioni tramite la rete e migliorare così la loro interazione con l'ambiente.

Concludendo, non è ancora possibile definire limiti precisi alla robotica e all'apprendimento robotico; le tecniche, presentate in questo elaborato, risultano efficaci, ma non sono perfette: se non saranno, comunque,direttamente protagoniste nel futuro della robotica, saranno sicuramente alla base delle future applicazioni.

Bibliografia

[1] B. Akgun, M. Cakmak, K. Jiang, and A. Thomaz. Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4(4):343–355, 2012.

[2] A. Alissandrakis, C. Nehaniv, and K. Dautenhahn. Correspondence mapping induced state and action metrics for robotic imitation. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 37(2):299–307, April 2007.

[3] E. Alpaydin. *Introduction to Machine Learning*. The MIT Press, 2nd edition, 2010.

[4] C. W. Anderson. Strategy learning with multilayer connectionist representations. In *In Proceedings of the Fourth International Workshop on Machine Learning*, pages 103–114. Morgan Kaufmann, 1987.

[5] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robot. Auton. Syst.*, 57(5):469–483, May 2009.

[6] C. G. Atkeson and S. Schaal. Robot learning from demonstration. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 12–20, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[7] J. Babič, J. G. Hale, and E. Oztop. Human sensorimotor learning for humanoid robot skill synthesis. *Adaptive Behavior - Animals, Animats, Software Agents, Robots, Adaptive Systems*, 19(4):250–263, Aug. 2011.

[8] A. G. Barto, R. S. Sutton, and C. W. Anderson. Artificial neural networks. chapter Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problems, pages 81–93. IEEE Press, Piscataway, NJ, USA, 1990.

[9] R. Bellman. *Dynamic programming*. Princeton University Press, Princeton, NY, 1957.

[10] R. Bellman. *Dynamic Programming*. Dover Publications, 1957.

[11] A. Billiard, S. Callion, R. Dillman, and S. Schaal. *Robot programming by demostration*, chapter 59. MIT Press, 2007.

[12] M. Brady. *Robot Motion: Planning and Control*. Artificial Intelligence Series. MIT Press, 1982.

[13] W. A. Clark and B. G. Farley. Generalization of pattern recognition in a self-organizing system. In *Proceedings of the March 1-3, 1955, Western Joint Computer Conference*, AFIPS '55 (Western), pages 86–91, New York, NY, USA, 1955. ACM.

[14] R. Coulom. *Reinforcement Learning Using Neural Networks, with Applications to Motor Control*. PhD thesis, Institut National Polytechnique de Grenoble, 2002.

[15] R. Dillmann, M. Kaiser, and A. Ude. Acquisition of elementary robot skills from human demonstration. In *In International Symposium on Intelligent Robotics Systems*, pages 185–192, 1995.

[16] P. F. Dominey, A. Mallet, and E. Yoshida. Real-time spoken-language programming for cooperative interaction with a humanoid apprentice. *I. J. Humanoid Robotics*, 6(2):147–171, 2009.

[17] K. Fu. *Learning Control Systems: Review and Outlook*. School of Electrical Engineering, Purdue University, 1969.

[18] D. Grollman and A. Billard. Robot learning from failed demonstrations. *International Journal of Social Robotics*, 4(4):331–342, 2012.

[19] D. H. Grollman and A. Billard. Donut as i do: Learning from failed demonstrations. In *ICRA*, pages 3804–3809. IEEE, 2011.

[20] A. J. Ijspeert, J. Nakanishi, and S. Schaal. Learning attractor landscapes for learning motor primitives. In *in Advances in Neural Information Processing Systems*, pages 1523–1530. MIT Press, 2003.

[21] K. Ikeuchi, M. Kawade, and T. Suehiro. Toward assembly plan from observation - task recognition with planar, curved and mechanical contacts. In *IROS*, pages 2294–2301. IEEE, 1993.

[22] M. Kawato, F. Gandolfo, H. Gomi, and Y. Wada. Teaching by showing in kendama based on optimization principle. In M. Marinaro and P. Morasso, editors, *ICANN 94*, pages 601–606. Springer London, 1994.

[23] A. H. Klopf. Brain function and adaptive systems - a heterostatic theory. Technical Report AFCRL-72-0164, Air Force Cambridge Research Laboratories, L. G. Hanscom Field, Bedford, MA, 1972.

[24] A. H. Klopf. A comparison of natural and artificial intelligence. *SIGART Bull.*, (52):11–13, June 1975.

[25] J. Kober and J. Peters. Learning motor primitives for robotics. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 2112–2118, May 2009.

[26] P. Kormushev, S. Calinon, D. Caldwell, and B. Ugurlu. Challenges for the policy representation when applying reinforcement learning in robotics. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pages 1–8, June 2012.

[27] P. Kormushev, S. Calinon, and D. G. Caldwell. Robot motor skill coordination with EM-based Reinforcement Learning. pages 3232–3237, Oct. 2010.

[28] P. Kormushev, S. Calinon, and D. G. Caldwell. Reinforcement learning in robotics: Applications and real-world challenges. *Robotics*, 2(3):122–148, 2013.

[29] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Norwell, MA, USA, 1991.

[30] A. Markov and N. Nagorny. *The Theory of Algorithms*. Mathematics and its Applications. Springer, 2010.

[31] J. M. Mendel and R. W. McLaren. A prelude to neural networks. chapter Reinforcement-learning Control and Pattern Recognition Systems, pages 287–318. Prentice Hall Press, Upper Saddle River, NJ, USA, 1994.

[32] D. Michie. Trial and Error. In S. A. Barrett and A. McLaren, editors, *Penguin Science Survey 1961*, pages 129–145. Penguin Books Ltd., London, UK, June 1961.

[33] D. Michie. Experiments on the mechanization of game-learning Part I. Characterization of the model and its parameters. 6(3):232–236, Oct. 1963.

[34] M. Minsky. *Theory of Neural-analog Reinforcement Systems and Its Application to the Brain Model Problem*. Princeton University., 1954.

[35] M. Minsky. Steps toward artificial intelligence. In *Computers and Thought*, pages 406–450. McGraw-Hill, 1961.

[36] T. Mitchell. *Machine Learning (Mcgraw-Hill International Edit)*. McGraw-Hill Education (ISE Editions), 1st edition, Oct. 1997.

[37] H. Miyamoto, S. Schaal, F. Gandolfo, H. Gomi, Y. Koike, R. Osu, E. Nakano, Y. Wada, and M. Kawato. A kendama learning robot based on bi-directional theory. *Neural Networks*, 9(8):1281 – 1302, 1996. Four Major Hypotheses in Neuroscience.

[38] K. Muelling, J. Kober, O. Kroemer, and J. Peters. Learning to select and generalize striking movements in robot table tennis. (3):263–279, 2013.

[39] C. L. Nehaniv and K. Dautenhahn. Like me? – measures of correspondence and imitation, 2001.

[40] C. L. Nehaniv and K. Dautenhahn. Imitation in animals and artifacts. chapter The Correspondence Problem, pages 41–61. MIT Press, Cambridge, MA, USA, 2002.

[41] L. Peternel and J. Babic. Humanoid robot posture-control learning in real-time based on human sensorimotor learning ability. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 5329–5334, May 2013.

[42] J. Peters and S. Schaal. Natural actor critic. (7-9):1180–1190, 2008.

[43] L. Rejeb, Z. Guessoum, and R. M'Hallah. An adaptive approach for the exploration-exploitation dilemma for learning agents. In M. Pechoucek, P. Petta, and L. Z. Varga, editors, *CEEMAS*, volume 3690 of *Lecture Notes in Computer Science*, pages 316–325. Springer, 2005.

[44] S. Schaal. Is imitation learning the route to humanoid robots? *Trends Cogn Sci*, 3(6):233–242, June 1999.

[45] S. Schaal, A. Ijspeert, and A. Billard. Computational approaches to motor learning by imitation. *Philos Trans R Soc Lond B Biol Sci*, 358(1431):537–547, March 2003.

[46] S. Schaal, P. Mohajerian, and A. Ijspeert. Dynamics systems vs. optimal control a unifying view. In T. D. Paul Cisek and J. F. Kalaska, editors, *Computational Neuroscience: Theoretical Insights into Brain Function*, volume 165 of *Progress in Brain Research*, pages 425 – 445. Elsevier, 2007.

[47] S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert. Learning movement primitives. In *International Symposium on Robotics Research (ISRR2003*. Springer, 2004.

[48] A. Segre and G. DeJong. Explanation-based manipulator learning: Acquisition of planning ability through observation. In *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, volume 2, pages 555–560, Mar 1985.

[49] P. Simon. *Too Big to Ignore: The Business Case for Big Data*, page 89. Wiley, 2013.

[50] K. O. Stanley and R. Miikkulainen. Efficient reinforcement learning through evolving neural network topologies. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '02, pages 569–577, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[51] R. S. Sutton. *Temporal Credit Assignment in Reinforcement Learning*. PhD thesis, 1984. AAI8410337.

[52] R. S. Sutton and A. G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.

[53] G. Tesauro. Practical issues in temporal difference learning. *Machine Learning*, 8:257–277, 1992.

[54] G. Tesauro. Temporal difference learning and td-gammon. *Commun. ACM*, 38(3):58–68, Mar. 1995.

[55] E. L. Thorndike. *Animal intelligence : an experimental study of the associative processes in animals / by Edward L. Thorndike.* New York :Macmillan,. http://www.biodiversitylibrary.org/bibliography/25848.

[56] S. Vijayakumar and S. Schaal. Locally weighted projection regression: An o(n) algorithm for incremental real time learning in high dimensional space. In *in Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000*, pages 1079–1086.

[57] M. Waibel, M. Beetz, R. D'Andrea, R. Janssen, M. Tenorth, J. Civera, J. Elfring, D. Gálvez-López, K. Häussermann, J. Montiel, A. Perzylo, B. Schießle, O. Zweigle, and R. van de Molengraft. RoboEarth - A World Wide Web for Robots. *Robotics & Automation Magazine*, 18(2):69–82, 2011.

[58] M. Waltz and K. Fu. A heuristic approach to reinforcement learning control systems. *Automatic Control, IEEE Transactions on*, 10(4):390–398, Oct 1965.

[59] C. Watkins and P. Dayan. Technical note: Q-learning. *Machine Learning*, 8(3-4):279–292, 1992.

[60] C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, May 1989.

[61] B. Widrow, N. K. Gupta, and S. Maitra. Punish/reward: Learning with a critic in adaptive threshold systems. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-3(5):455–465, Sept 1973.

[62] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3-4):229–256, May 1992.

[63] Y. Zhang and J. Weng. Task transfer by a developmental robot. *Evolutionary Computation, IEEE Transactions on*, 11(2):226–248, April 2007.