

Il riconoscimento delle ombre

Come facemmo notare la volta scorsa, il problema del riconoscimento degli oggetti non può più essere relegato a quello di corpi convessi, ben illuminati, con bordi diritti e rettilinei e senza ombre. È giunto il momento di allargare la nostra possibilità di conoscenza e, soprattutto di capacità di riconoscimento di oggetti inteso come classificazione degli stessi; finora avevamo discusso solo di riconoscimento, invece di linee e spigoli in un oggetto. Ma da questi come si fa a passare alla individuazione della «natura» del corpo stesso?

La maggior critica rivolta alla «lettura» artificiale degli oggetti è rappresentata dal fatto che l'analisi eseguita attraverso una «esplosione combinatoriale» degli oggetti, così come eseguita dai programmi finora descritti, è distruttiva di ogni realmente complesso algoritmo di riconoscimento. Tutti avranno compreso che i diversi algoritmi proposti da Guzman, Grape, Shirai ecc. sono, nella loro limitazione, ancora estremamente estenuanti nelle operazioni di confronto della parte d'oggetto riconosciuta con modelli precostituiti (i lavori più critici, che mettono alle corde in maniera anche feroce i difetti dei metodi finora descritti sono quelli di Sir James Lighthill e di HL Dreyfus (ed altri), raccolti in «Artificial Intelligence: A Paper Symposium» Londra 1973. Gli stessi problemi furono stigmatizzati ancora, successivamente, da P.J. Hayes in «Some Comments on Sir James Lighthill Report on Artificial Intelligence» AISB Study Group European Newsletter, n. 14 (luglio 1983) e, ancora in Dreyfus («What computer can't do; a critique of artificial reason» New York, 1972). Ammesso di risolvere, quindi, il problema del riconoscimento dei contorni, resta sempre il grave ostacolo del riconoscimento dell'oggetto totale per confronto con una «libreria» di oggetti precostituiti, cosa che nessun computer può ragionevolmente eseguire se non su modelli molto semplici. Con oggetti appena più complicati (con ciò intendo ancora corpi geometrici semplici, ma con facce non più ortogonali tra loro) il numero delle possibili combinazioni delle diverse misure delle diverse facce sale in maniera

astronomica, del tutto superiore alla possibilità di manipolazione di una macchina, oltre tutto destinata a lavorare in tempo reale. Svanisce nel nulla, se ancora esisteva, quindi, la possibilità di poter disporre in un futuro prossimo, di una macchina come HAL di 2001, capace, addirittura, di leggere parole su un corpo indefinito come le labbra di Bowman.

Come è possibile, allora, ottenere i migliori risultati, almeno nel riconoscimento di forme non proprio elementari, ma ancora definibili geometricamente, senza rinunciare ad una accettabile velocità del sistema. La risposta potrebbe sembrare paradossale, ma è realistica: ogni nuova conoscenza può essere utilizzata, oltre che per aggiungere informazioni, anche per sfrondare le pregresse conoscenze già acquisite e semplificare il problema a tutto vantaggio della semplicità del complesso.

Facciamo un esempio: nessuno dei programmi finora esaminati, tranne forse quello di Grape, molto elastico rispetto ai precedenti, tollerava la presenza di ombre. Sembra una contraddizione in termini, ma se i parametri della illuminazione non sono attentamente prefissati e noti alla macchina, è possibile giungere ad errori paradossali nella lettura della immagine stessa. Ritenendo, in questa ottica, le ombre una spiacevole caratteristica della immagine, Richard Orban scrisse un ben articolato programma (R. Orban, «Removing shadows in a Scene» Cambridge, Mass. 1972) che rimuoveva le ombre dall'immagine in base ad un algoritmo simile a quello utilizzato da Guzman. La cosa non è affatto generalizzabile e spesso neppure utile, in quanto sovente le ombre stesse possono essere estremamente utili in ottica di prospettiva, come ad esempio nella figura A dove la mancanza d'ombra non consente di individuare se e dove esista la distanza tra gli oggetti. Inoltre, e non sembra cosa da poco, la stessa ombra permette di orientare l'immagine ricevuta in termini di alto-basso, partendo dal presupposto che l'illuminazione viene eseguita da sopra. Fatti i debiti confronti ed analizzati i pro ed i contro, ci si è resi conto che l'abolizione delle ombre era, alla fine, più un danno che un vantaggio; il 1974 segnò la vittoria di tale concezio-

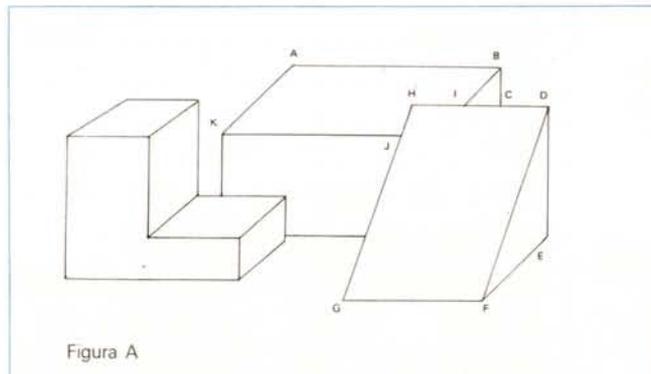
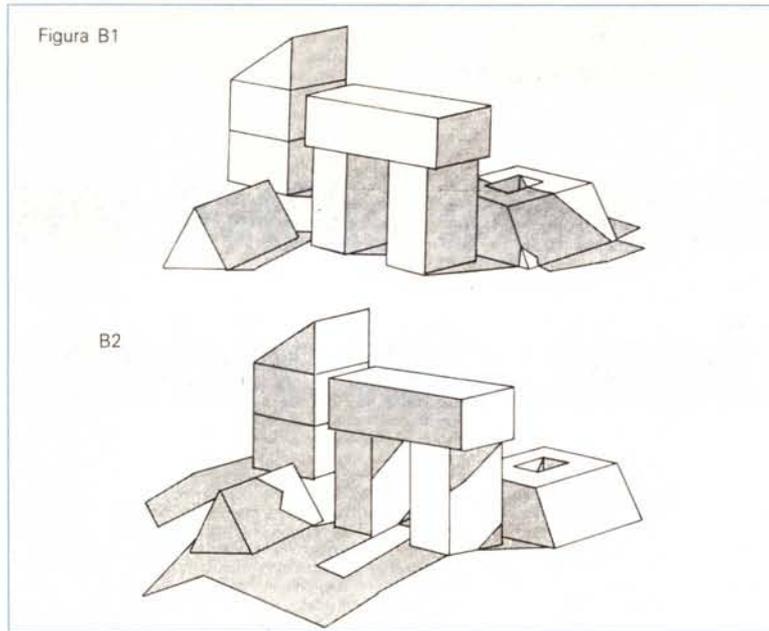


Figura A



ne, oseremmo dire filosofica, e le ombre furono a buona ragione inserite nei parametri d'analisi per la determinazione della natura degli oggetti, piuttosto che essere considerate come un'appendice inutile, un «rumore di fondo non necessario».

Questo principio fu appieno abbracciato da D. Waltz («Understanding Line Drawing of Scene with Shadow»), che mise a punto un potente programma capace di leggere scene anche discretamente complesse, ed ombreggiate; il programma era così efficiente da riuscire a riconoscere come la stessa scena le figure di B1 e B2.

In breve, il programma di Waltz interpretava le linee nei disegni di scene ombreggiate non solo nella loro dimensione principale, ma anche nelle regioni immediatamente circostanti. Se, come accade nel caso delle ombre, il pattern delle aree così indagate è riconducibile ad una serie di figure non «agganciabili» ad altre parti della figura stessa, esiste la possibilità che lo stesso poligono rappresenti un'ombra. Waltz utilizzava, nel suo programma, undici «schede» di base di riconoscimento di ombre. Il concetto fu subito ripreso da Martin Rattner (M.H. Rattner, «Extending Guzman's SEE Program», Cambridge, Mass. 1972), che nel suo programma, definito non a caso SEEMORE, superava ed ampliava i concetti del pur valido SEE, estendendone gli obiettivi, o come nel più puro slang informatico, il target domain. Il maggior problema affrontato da Rattner in questo senso, vale a dire l'abnorme proliferazione di combinazioni di immagini derivate dalla non prevedibilità delle forme analizzate fu enormemente semplificata dall'autore, che «costrinse» il programma, tramite un opportuno algoritmo, a riconoscere non

più di un migliaio di combinazioni diverse. Ed è questo forse il gran vantaggio del programma di Rattner; quello di riunire le indubbe capacità di un programma Waltz-like con la riduzione della necessità di riconoscimento di grandi numeri di figure.

Il gran merito di programmi come quelli di Rattner e Waltz sono rappresentati dall'aver reso, almeno parzialmente, intelligente l'algoritmo stesso. Inoltre, a corredo di una idea già ben nota, Waltz introdusse nella sua creatura un metodo di analisi della figura condizionato alla eventuale conoscenza delle modalità e della intensità di illuminazione della scena stessa. In base ai

risultati di Waltz e Rattner, K.J. Turner («Computer Perception of Curved Objects») perfezionò l'algoritmo, rendendolo capace, per la prima volta, di leggere ed interpretare figure curve, anche complesse, come quelle della figura C; in breve, Turner utilizzava un algoritmo di analisi di figure poliedriche per giungere al riconoscimento di intersezione di curve e piani. Ad esempio, nella figura D, il programma legge una forcina positiva e la trasforma in una curva, anche essa positiva, con un punto di giunzione al centro rappresentante, a tutti gli effetti, il vertice 3 + del parallelepipedo stesso.

Turner approfittò dell'occasione per ampliare, ancora di più, il concetto di leggibilità della immagine, introducendo la possibilità di discernere anche la direzione di illuminazione della scena. Si giunge, così, ad un programma ormai abbastanza efficiente, soprattutto grazie alla possibilità d'analisi di superfici curve; manca, ovviamente, ancora la possibilità di analizzare superfici non piane o con due raggi di curvatura, o più genericamente, irregolari. Siamo ancora lontani, ovviamente, da macchine intelligenti (basti pensare la semplice possibilità di analizzare un oggetto in moto per capire quanto ancora siamo lontani dalla soluzione), ma non disperiamo; la via da percorrere è ancora abbastanza lunga; d'altro canto siamo, nella nostra analisi, ancora nel 1973, un'era, in termini di storia informatica, lontano dalla data odierna. **MC**

