Localizzazione di sorgenti di segnale ed istituzione di ponti di comunicazione con agenti mobili

Massimiliano Lungaro (566708 - IAM) , Enrico Maran (566758 - IAM) ,
Gian Antonio Susto (566706 - IAM)

***Abstract*—Il lavoro presentato in questo documento é incentrato sul problema di costituire un ponte di comunicazione con una sorgente di segnale, di collocazione ignota, attraverso l'utilizzo di un network di nodi mobili. Nel corso di questo progetto abbiamo supposto che gli agenti mobili disponessero esclusivamente di informazioni di posizione relative e non assolute: nessun Global Position System, ma solo distanze relative comunicate con altri nodi del network. Sono stati sviluppati approcci diversi al problema, che si distinguono fra loro essenzialmente per l'ordine con cui le fasi risolutive sono state realizzate: il primo algoritmo ideato e sviluppato, denominato *Connectivity-First* (CF) ricerca la sorgente di segnale (SS Source Signal) mantenendo ad ogni istante la connettivitá fra i nodi; in seguito verranno presentati diverse soluzioni nelle quali, invece, prima viene ricercata la SS e, successivamente, viene instaurato un ponte di comunicazione fra i vari robot: ci si riferirá a questa categoria di algoritmi con il nome *Search-First* (SF). L'approccio in cui la ricerca viene anteposta alla connettivitá si é dimostrato essere piú performante e robusto, comprovato da numerosi risultati simulativi: a supporto di quanto argomentato, sono riportate tutte le statistiche ottenute con questo tipo di implementazione. Infine, si é implementato in maniera sperimentale l'algoritmo $SF1$, che si é dimostrato il piú adatto alla strumentazione hardware a disposizione nel Laboratorio di Navigazione Autonoma (NavLab) del DEI.**

***Index Terms*—Autonomous Search, Autonomous Systems, Controllo Distribuito, Localizzazione, Ponte Radio, Swarm Robotics.**

I. Introduzione

LA localizzazione di uno o piú target in un ambiente sconosciuto tramite dispositivi mobili costituisce al giorno d'oggi un'interessante, e pressoché inesplorata, tematica per i suoi innumerevoli contesti applicativi: problematiche relative al caso di un singolo robot esploratore sono giá state ampiamente sviluppate, mentre la ricerca effettuata attraverso l'impiego di un sistema di ricerca multi-robot risulta relativamente un campo ancora inesplorato e pertanto necessita di ulteriori numerosi approfondimenti.

L'obbiettivo di questo lavoro consiste nello sviluppare ed instituire un ponte di comunicazione fra una sorgente di segnale, di ubicazione ignota, e una base di partenza attraverso un network robotico. Si suppone di aver a disposizione un gruppo di unitá mobili, dotate di un apparato sensoriale che consente loro di dedurre informazioni correlate alle distanze relative all'interno del sistema.

Un'ipotesi complicativa fondamentale é stata quella di supporre il network robotico privo di alcun *Global Position System (GPS)*: tale supposizione comporta un incremento del livello di difficoltá nelle soluzioni implementative, dato che gli agenti robotici non dispongono di posizioni assolute, ma esclusivamente relative. Si é reso quindi necessario adottare un controllo di tipo distribuito per disciplinare il movimento del network: per la gestione della ricerca non é consentito l'utilizzo di una unitá centrale di calcolo, data l'impossibilitá della base centrale di comunicare con ogni nodo ad ogni istante.

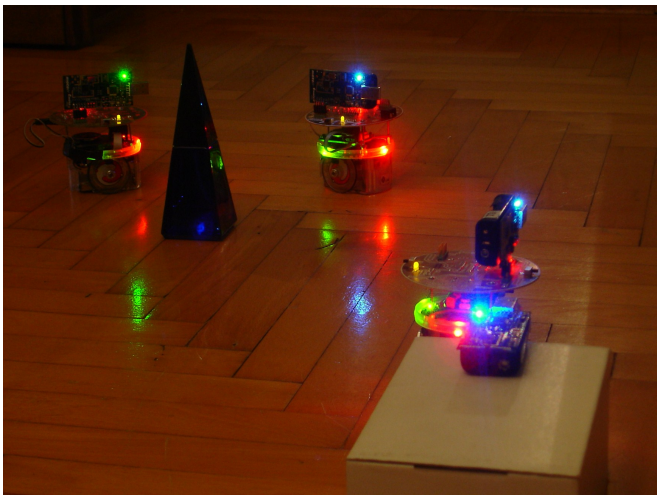

Fig. 1. Una delle prove effettuate con i robot *e-puck*

In secondo luogo abbiamo supposto di aver omogeneitá nel gruppo di agenti; ogni nodo é identico ad un altro per caratteristiche e mansioni: non v'é alcun tipo di gerarchia stabilita a priori fra le unitá. Tale approccio garantisce numerosi vantaggi in termini di scalabilitá del sistema: ogni algoritmo é stato implementato considerando un generico numero di agenti ricercatori e l'introduzione di ulteriori nodi nella swarm non comporta alcun cambiamento nella programmazione degli altri robot.

Discriminante principale nella filosofia d'esecuzione della ricerca, é stata la scelta di mantenere o meno la connettivitá fra i nodi ad ogni istante iterativo: dato che non in tutte le situazioni reali é possibile prescindere da tale specifica, sono stati realizzati algoritmi per coprire entrambe le casistiche. Vengono soprannominati di tipo CF, *Connectivity-First*, le soluzioni implementative dove mantenere la connettivitá é la prima specifica da soddisfare ad ogni iterazione, mentre, gli algoritmi dove il ponte di comunicazione viene stabilito solo una volta trovato il target, vengono detti di tipo SF, *Search-First*.

La prima delle soluzioni algoritmiche proposte, l'algoritmo

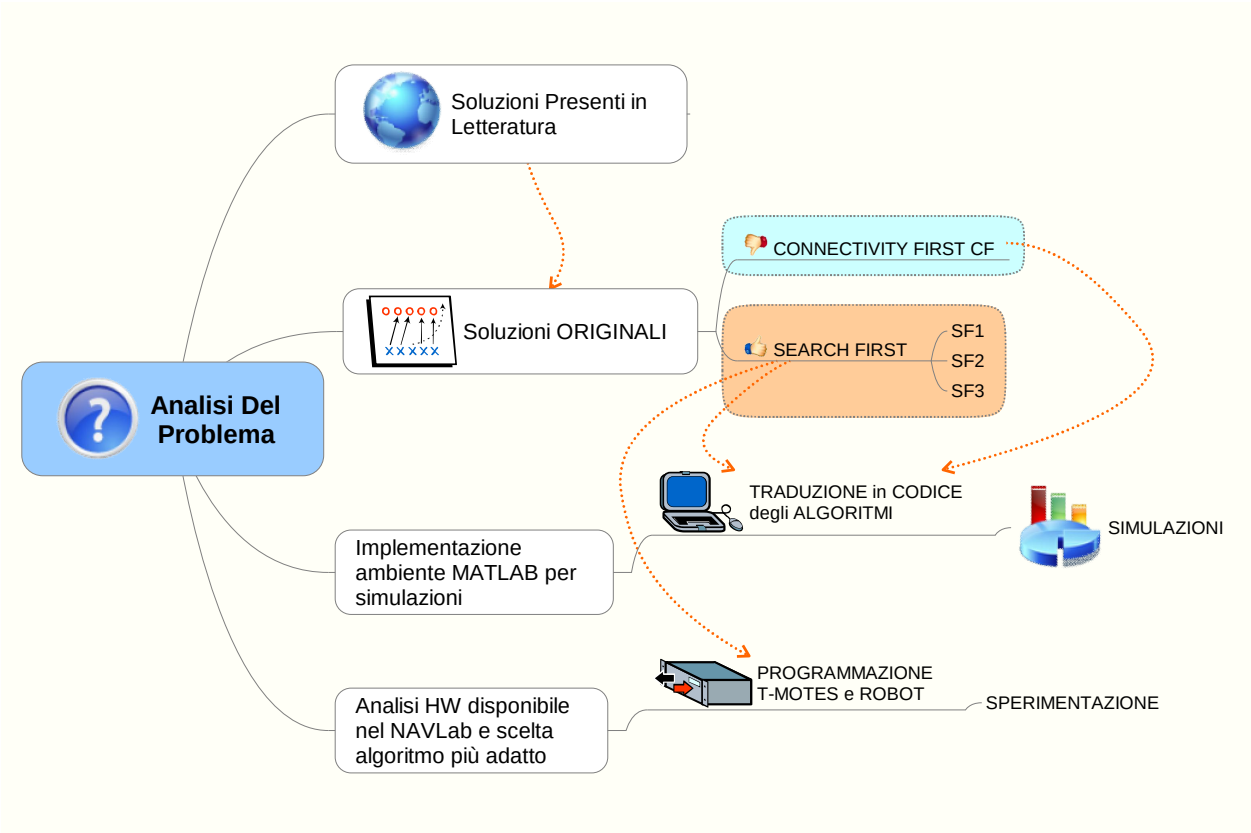

Fig. 2. I vari step del lavoro svolto: mappa concettuale

CF, é stata appunto sviluppata preservando ad ogni passo la comunicazione; tale scelta comporta una notevole complicazione del movimento che dev'essere sincronizzato fra piú agenti, introducendo pesanti complicazioni e rallentamenti per il processo di ricerca. Si é richiesta, ad esempio, la necessitá di mantenere una distanza relativa fra oggetti in moto, il che comporta un appesantimento del carico computazionale.

Dato che lo sviluppo di tali tematiche é pressoché recente, in letteratura non sono presenti lavori che trattino con cura tutti gli aspetti coinvolti; [8] é uno dei pochi articoli che considerano in maniera articolata le varie sfacettature del problema: l'approccio da noi proposto riprende la medesima struttura a colonne per la ricerca (come sará spiegato esaustivamente in seguito), ma differisce per l'ordine con cui lo spazio inesplorato viene esaminato; mentre nell'algoritmo presentato in [8] l'analisi procede per rette centrate nella base, in CF si procede per cerchi concentrici centrati anch'essi nella BS: in tal senso si intuisce come l' algoritmo qui proposto segua un criterio quasi deterministico, mentre nella soluzione trovata in letteratura la sequenzialitá con cui l'analisi viene portata avanti avviene in maniera praticamente randomizzata. É infine presentata un'implementazione alternativa di CF, adatta ad un hardware con capacitá di calcolo superiori: tale variante all'algoritmo consente di ovviare alle problematiche presenti in CF, introducendo peró ipotesi poco verosimili agli strumenti disponibili per l'implementazione pratica; é per queste motivazioni che tale approccio é presentato solo in maniera teorica e non simulativa.

Gli algoritmi $SF1$ ed $SF2$ successivamente esposti si discostano dalla procedura sistematica e computazionalmente impegnativa appena illustrata,sfruttando un metodo di ricerca semplice ed efficace del segnale, basato sull'esecuzione di movimenti in direzioni casuali. Tale metodica non prevede quindi il mantenimento del contatto radio tra gli agenti, ma semplicemente un ritorno ad intervalli regolari alla BS per il reperimento di informazioni sullo stato del ponte. Il piú elementare dei due approcci, l'$SF1$, si basa su una estesa ripetizione del segnale incognito: l'algoritmo prevede la mappatura da parte degli agenti dello spazio circostante alla BS, ed il loro arresto immediato se giunti a contatto con la sorgente o con qualche altro robot componente del ponte in costruzione.

La successiva evoluzione algoritmica, l'$SF2$, punta ad una progressiva diminuzione di taglia del problema da risolvere: ogniqualvolta un'unitá identifica un target (sia la SS che un altro nodo giá in collegamento con questa) non si limita esclusivamente a ripetere il segnale, ma torna sui suoi passi e riferisce alla BS l'informazione relativa alla distanza dell'obbiettivo;in questa maniera si riduce progressivamente il campo di ricerca assegnato poiché i restanti robot si inoltrano in esplorazioni di distanza piú breve. L'ultimo algoritmo, l'$SF3$ sfrutta la semplicitá computazionale del metodo di ricerca suddetto ed opera la realizzazione del ponte radio tramite movimento coordinato dei veicoli: l'agente che per primo ha successo nella ricerca di SS, non solo riporta informazione sulla distanza del target trovato, ma anche sulla direzione dello stesso.

Dopo un accurato studio dei data sheet dei Tmotes per

valutare, in linea di principio, le possibilitá implementative per il progetto, si é passati ad una visita in laboratorio per verificare anche i dettagli del caso specifico e le problematiche software e di connessione con i robots e-Puck su cui i Tmotes sono montati. Si é preferito agire in questo modo al fine di mantenere un certo grado di conformitá e verosimiglianza fra le soluzioni proposte in Matlab e la realtá sperimentale.

Una volta osservate le numerose difficoltá implementative, si é deciso di ricreare in laboratorio solamente l'approccio algoritmico $SF1$. Per quanto riguarda le due estensioni ulteriori $SF2$ ed $SF3$ si é consci che il passaggio mentale per l' implementazione di queste é relativamente semplice, una volta apprese le metodologie di programmazione ed interfacciamento per i dispositivi in adottati. Per ottenere una cooeranza adeguata fra le strutture di laboratorio e gli algoritmi presentati si é reso necessario un riscalamento delle potenzialitá della strumentazione hardware a disposizione alle dimensioni compatibili con la struttura del laboratorio, hanno costituito aspetti nocivi per le prestazioni del sistema.
Ad ogni modo, grazie soprattutto alla semplicitá che caratterizza la soluzione $SF1$, si riscontra comunque un interessante comportamento del sistema reale in termini di velocitá d'esecuzione e di coordinazione del movimento delle unitá che compongono il sistema.
In definitiva quindi, nonostante gli ostacoli rappresentati dalla variabilitá di intensitá della segnale radio e dall'interfacciamento Tmote-robot, i risultati sperimentali ricavati si possono considerare soddisfacenti.

II. Stato dell'arte Esteso

In questa sezione si intende presentare lo stato dell'arte in merito alla tematica sviluppata per il progetto. Per questo intento, si offre dapprima una visione generale della situazione, per poi addentrarsi piú dettagliatamente nell'analisi di alcuni articoli, ritenuti particolarmente interessanti e significativi.

Come giá accennato in fase introduttiva, le dimensioni del panorama letterario tuttora esistente circa la ricerca multi-robot sono piuttosto succinte. Malgrado ció si possono riscontrare numerose idee algoritmiche, che riguardano peró un approccio di soluzione per mezzo di un unico robot esploratore.

I vantaggi che si hanno nell'approcciarsi al problema utilizzando un network di agenti rispetto alla sua alternativa a nodo singolo (filosofia dominante fino a qualche anno addietro) sono comunque numerosi. In primo luogo si rileva una significativa riduzione del tempo di ricerca del target dovuta al lavoro in parallelo di molte unitá. Altre importanti caratteristiche sono la scalabilitá del sistema e la ridondanza d'informazione, alla quale corrisponde una diminuzione della possibilitá di scelte erronee per il procedimento di ricerca, a cui a sua volta consegue una minor probabilitá di fallimento. Inoltre, soprattutto a livello pratico, risulta molto utile il fatto che la rottura di uno o piú dispositivi non comporti il fallimento dell'intera operazione e che quest'eventualitá non sia compromettente in termini economici. Infatti un singolo robot esploratore richiede necessariamente un'architettura hardware molto piú performante e costosa (in particolar modo l'apparato sensoriale) e quindi il verificarsi di un guasto puó divenire problematico. Da sottolineare che quest'ultimo aspetto risulta molto interessante quando l'impiego di siffatti sistemi é rivolto ad applicazioni di disinnescamento di ordigni inesplosi o brillazione degli stessi.

É bene notare peró che, a discapito di tutti questi vantaggi, si richiede una capacitá computazionale non banale non appena si voglia simulare il comportamento di questi sistemi con un certo grado di realismo. Infatti per verificare le prestazioni offerte si usa applicare gli algoritmi appositi a modelli semplificati, in modo tale da garantire la fattibilitá della simulazione anche a fronte di numerositá elevate per la *swarm* di ricerca. Ovviamente questo tipo di semplificazioni consentono valutazioni necessariamente approssimative delle performance e pertanto é buona norma mantenere un atteggiamento critico nei confronti dei dati pervenuti a calcolatore.

In aggiunta poi bisogna evidenziare che con il sistema multi-robot si ha il vantaggio di poter instaurare un ponte radio fra base e target di ricerca. Se si volesse ottenere uno scambio di informazione simile con un robot singolo si necessiterebbe un movimento del robot. Invece, agendo per via distribuita, si crea un canale fisico che consente una comunicazione real time fra base e target.

Fatte tutte queste considerazioni, si puó cominciare a studiare alcune soluzioni algoritmiche al problema di ricerca giá presenti in letteratura per valutarne l'efficacia ed ottenere spunti di confronto con l'approccio proposto in questo progetto.

Come punto di partenza si é prestata attenzione ad una recente (2007) linea di pensiero che prevede l'aggiustamento di un algoritmo abbastanza noto nel settore, quale il *Particle Swarm Optimization*(PSO), in modo tale da ottenerne una variante ad hoc per risolvere il problema di multi-robot search in un ambiente limitato e senza ostacoli [9]. In termini generici il PSO si basa su principi socio-psicologici e fornisce informazioni sul comportamento di una collettivitá con il fine di raggiungere una sorta di ottimizzazione sociale. Secondo tale algoritmo all'istante iniziale ogni particella *i* si trova nella posizione x_i, che in generale sará data da un vettore $[x_{i,j}]$, con $j = 1, \ldots, n$ visto che si suppone di operare in un ambiente di ricerca *n*-dimensionale, con una velocitá $[v_{i,j}]$. Si cerca di ottimizzare passo passo il posizionamento delle unitá nell'area muovendole verso i punti in cui si sono registrati i risultati migliori. Infatti ogni particella tiene in memoria la posizione in cui ha sentito la minor distanza dal target di ricerca $x^*_{i,j}$ e anche quella che ha dato la miglior performance fra tutte le particelle ad essa "'vicine"' $x^*_{i',j}$. Ad ogni passo iterativo le equazioni descrittive del sistema sono:

$$\begin{aligned} v_{i,j} &= w \cdot v_{i,j} + pw \cdot rand() \left(x^*_{i,j} - x_{i,j}\right) + \\ &\qquad + nw \cdot rand() \left(x^*_{i',j} - x_{i,j}\right) \\ x_{i,j} &= x_{i,j} + v_{i,j} \end{aligned}$$

dove w é un coefficiente d'inerzia necessario per garantire la convergenza finale della swarm, pw (particle weight) e nw (neighborhood weight) sono coefficienti di peso rappresentativi dell'attrazione esercitata dalle migliori posizioni trovate in precedenza, e $rand()$ sta a simboleggiare il campionamento di una variabile aleatoria uniforme nell'intervallo $[0, 1]$.

Dalle equazioni appena scritte si puó intravedere un parallelismo con le strutture sociali del mondo biologico: l'idea é che, per raggiungere un obiettivo collettivo, si necessiti di iniziativa personale fortemente correlata alla collaborazione fra tutti i componenti della squadra, che si manifesta attraverso lo scambio di informazioni circa i risultati ottenuti da ciascuno. Nonostante questa analogia con la realtá organica, si deve tener conto che la traduzione di una filosofia come questa in un ambito pratico assume necessariamente delle limitazioni in termini di comunicazione, di memoria e di capacitá di movimento, le quali vanno specificate e trattate caso per caso. Ricerche recenti hanno infatti dimostrato come non soltanto le prestazioni ma anche la verosimiglianza fra i risultati reali e quelli ottenuti tramite modellizzazione(cioé quelli previsti) vari in dipendenza di parametri come il range di comunicazione dei singoli robot, il loro numero nella swarm e la loro disponibilitá ad immagazzinare dati in memoria.

Un'altra considerazione notevole che a primo impatto puó apparire banale é che la determinazione delle performance di un algoritmo é fortemente influenzata dalla metrica stessa con cui le si studiano. Nel problema preso in esame infatti si puó appuntare come l'oggetto fisico che costituisce il target di ricerca abbia in generale dimensioni non puntiformi e che quindi l'emettitore dal quale viene emanato il segnale da captare sia solitamente allocato in qualche punto all'interno del corpo considerato e non sulla sua periferia. Di conseguenza é ragionevole utilizzare come metrica di valutazione non solo la distanza fra il target e la posizione di rilevazione del segnale con massima intensitá(cioé la distanza fra il robot piú vicino al target ed esso stesso), ma anche la media delle posizioni migliori ottenute da ciascun robot fino all'istante finale. Si osserva che questa metrica varia in funzione degli stessi parametri dell'altra, ma in maniera differente. Resta quindi al buon senso del progettista la scelta dell'opportuno criterio di valutazione in base alle caratteristiche dei dispositivi che si hanno a disposizione.

Altro aspetto discriminante nelle prestazioni dei sistemi di ricerca é la presenza o meno di GPS: in letteratura si é indagata la risposta di sistemi siffatti sotto diverse condizioni d'utilizzo. Come é intuitivo pensare, nell'ipotesi di conoscenza esatta del posizionamento dei robot nell'ambiente di lavoro(mediante GPS) si ottengono le prestazioni migliori. Difatti in questa situazione si ha che i nodi che compongono il vicinato di ogni robot sono tutti quelli della swarm e ció implica un corposo contributo in termini di volume di informazioni scambiate. Come giá accennato nell'introduzione, l'intento di questo articolo é quello di analizzare sistemi privi di Global Position Systems: ci si vuole porre cioé in una situazione in cui ogni unitá possa conoscere solo la sua posiziona relativa(non globale) corrente e quella precedente. Inoltre si suppone che la localizzazione dei nodi vicini avvenga attraverso un ricevitore radio con un range di comunicazione limitato. Si ha cioé che soltanto alcuni nodi sono visibili dal singolo agente, mentre gli altri risultano fuori dal campo di comunicazione: ne consegue che ad ogni passo di iterazione dell'algoritmo viene a definirsi un nuovo vicinato per ogni robot.

Bisogna mettere in evidenza che per l'adattamento del PSO al caso di veicoli robotici reali occorrono alcuni accorgimenti:

- Le iterazioni del PSO avanzano in maniera discreta mentre i robot reali operano nel tempo continuo. Pertanto si divide ogni step dell'algoritmo in due momenti differenti: il primo in cui avviene lo scambio di informazioni fra i robot ed il calcolo della nuova posizione da raggiungere, il secondo dedicato al raggiungimento fisico di tale posizione. Da notare che un'ipotesi fondamentale per fare questo é che i robot siano sincronizzati.
- I robot non sono in grado di mutare istantaneamente la direzione del loro vettore velocitá, ma richiedono delle procedure di rotazione. Una soluzione per bypassare questo problema consiste nel prolungare il secondo dei due momenti dello step di iterazione appena menzionati. Si osserva peró che questo approccio di soluzione comporta un rallentamento del processo di ricerca.
- I robot possono collidere con i limiti esterni dell'ambiente di ricerca o con il target. Inoltre, non essendo realmente puntiformi, possono collidere anche fra loro. Un'idea per ovviare a tale situazione é di dividere la fase di moto in piú intervalli alla fine dei quali controllare il verificarsi di una "collisione" mediante l'impiego di un sensore di prossimitá. In caso affermativo il vettore velocitá del robot coinvolto manterrá la stessa ampiezza ma subirá un redirezionamento opportuno in base al tipo di collisione.

Per verificare il comportamento del sistema si é considerata un'arena quadrata con otto metri di lato e si sono piazzati in modo random i robot *e-puck* all'istante iniziale. Per quanto concerne l'impostazione dei parametri per la simulazione, l'unica cosa che vale la pena osservare é che gli autori hanno scelto un valore per il coefficiente d'inerzia piú grande del dovuto. Questo per favorire l'iniziativa personale d'esplorazione, ma a scapito di una minor interazione con gli altri robot. I risultati che si ottengono, come media di un migliaio di simulazioni da cento iterazioni dell'algoritmo ciascuna, possono essere schematizzati come segue. Come é ragionevole pensare, all'aumentare del numero di robot a disposizione la ricerca diviene piú efficace. Si puó anche notare come esista una soglia in cui é rilevabile una grossa discrepanza fra il caso semplificato e quello realistico. Ció si puó motivare pensando alla strategia modellizzata per evitare le collisioni, che, in combinazione con altri effetti, favorisce il raggruppamento dei robot in qualche punto dell'area danneggiando l'operazione di ricerca. É peraltro semplice convincersi che ad un incremento del range di comunicazione corrisponde un miglioramento delle prestazioni. In merito a questo, peró, é bene fare un'osservazione. Nel caso di dispositivi dotati di grandi capacitá di memoria, si nota che per range di comunicazione notevoli viene incentivata la formazione dei raggruppamenti appena menzionati. Infatti puó succedere che una rilevazione rumorosa porti a registrare erroneamente una posizione come molto vantaggiosa quando in realtá non lo é. Se il raggruppamento che ne consegue va a formarsi presso la periferia dell'ambiente di ricerca, dove il rapporto segnale-rumore del target assume valori piú bassi, é possibile che i robot non riescano piú a captare il segnale del target e che quindi la missione richieda tempi piú lunghi o addirittura fallisca. Nel caso in cui ci si é posti, peró, la disponibilitá

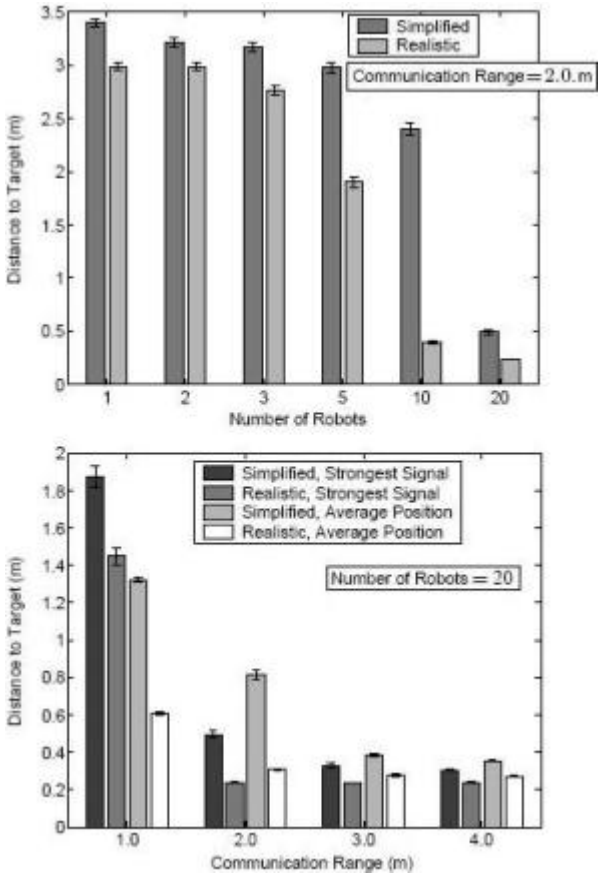

Fig. 3. Risultati_PSO.

mnemonica delle unitá é molto limitata e, in questo caso, ció si dimostra favorevole. Si ha cioé che una rilevazione erronea di segnale forte dovuta ad agenti esogeni(o altro) viene dimenticata al piú in due passi di iterazione, impedendo cosí il formarsi di raggruppamenti nocivi alle prestazioni del sistema.

Si é quindi analizzato un algoritmo di natura puramente randomizzata, frutto di un prolungamento specifico di una soluzione algoritmica piú generica. In letteratura sono presenti piú estensioni del PSO per risolvere la questione presentata, ma si é preferito studiare questa per motivi di similitudine con le condizioni sperimentali di laboratorio. Fondamentale é evidenziare che, con questo tipo di approccio, non ci si preoccupa di mantenere la connettivitá fra le unitá. Operando in questa maniera si ha il grosso vantaggio di accorciare i tempi d'esecuzione, ma d'altra parte si possono denotare alcuni aspetti negativi, che possono avere piú o meno rilievo in base alla condizione pratica d'impiego. Si pensi ad esempio al caso in cui l'ambiente di ricerca sia limitato da un burrone o contenga qualche crepaccio. In tal caso, con una struttura connessa, allo sparire repentino di un certo numero di unitá adiacenti(il caso di un singolo agente puó essere dovuto semplicemente a rottura) si puó lanciare un messaggio dall'allarme per avvertire il resto del gruppo del pericolo. In assenza di connettivitá invece é possibile che vadano persi tutti i robot, comportando di conseguenza un notevole danno economico. Si é fatta questa osservazione per trasmettere al lettore l'idea che, per quanto una soluzione possa essere performante in termini di velocitá ed elegante dal punto di vista computazionale, non sempre essa costituisce la soluzione migliore. Infatti non esiste una soluzione universale ottima per tutti i problemi reali, ma si richiedono accorgimenti ad hoc in base alle condizioni di lavoro ed ai vincoli specifici del caso.

Nonostante il panorama letterario circa la ricerca multi-robot sia ancora abbastanza povero, si é trovato un articolo in cui il problema descritto e il modo di affrontarlo si sono rivelati molto simili a quelli presi in esame per lo sviluppo di questo progetto. In *Chain based Path Formation in Swarm of Robots* ([8]) del 2006 infatti gli autori si pongono l'obiettivo di formare una sorta di ponte fra una base ed un target. Qui, peró, inizialmente i robot vengono piazzati in modo casuale su tutta l'area da esplorare e quindi devono preoccuparsi di trovare anche la base oltre che il target. Il controllo che si propone é di tipo distribuito e puramente omogeneo fra i robot e fa uso soltanto di informazioni e comunicazioni locali.

Il ponte che si vuole formare fra le due locazioni ha una struttura a catena rettilinea in cui ciascun robot identifica la propria posizione in maniera relativa a quello che lo precede nella catena. In questo modo si riesce ad economizzare la struttura hardware dei robot riducendo il numero di segnali da emettere a tre, invece che richiederne tanti diversi fra loro quanti sono i robot che compongono la catena. Cosí facendo allora, invece di utilizzare strutture sensoriali aggiuntive (come i Tmotes nel caso della strumentazione disponibile in laboratorio Navlab), per gestire la creazione delle colonne basta montare su ciascun robot tre led differentemente colorati e utilizzare la telecamera dei robot per riconoscere il colore. Oltre a questo aspetto notevole si é ritenuto interessante il fatto che l'evolversi dell'algoritmo sia determinato dal "comportamento" delle unitá. All'inizio della procedura ogni robot cerca la base. Una volta trovata cerca di unirsi ad una catena giá preesistente. Se peró non trova immediatamente nessuna catena, in base ad una probabilitá preimpostata $P_{e\to c}$(probabilitá esploratore $\mapsto$ catena), decide lui stesso di formare una nuova catena o attendere il nuovo step di iterazione. Inoltre si contempla anche la possibilitá che, prima di trovare la base, un robot trovi una catena. Al manifestarsi di quest'eventualitá il robot si unisce alla catena, risalendola e posizionandosi all'ultimo posto. Una peculiaritá di questo algoritmo é che, l'ultimo robot della catena, a seconda di una probabilitá $P_{c\to e}$, puó mutare il suo comportamento da membro di una catena ad esploratore. In altri termini l'ultimo figlio puó ritenere che la direzione in cui punta la catena sia poco fruttuosa per la ricerca del target e quindi decidere di staccarsi dalla catena per tornare a cercare la base ed eventualmente dare vita ad una nuova catena. Le catene in questo caso sono infatti orientate direzionalmente, cioé sono caratterizzate da una precisa posizione angolare. La perlustrazione dell'ambiente che ne risulta é quindi caratterizzata da raggi di dimensione variabile. Si puó anche osservare come questa variabilitá sia determinata dalla scelta del parametro $P_{c\to e}$. Infatti per valori bassi di questo i robot tendono a creare poche catene e lunghe e viceversa, per valori prossimi all'unitá, si avranno molte catene ma corte. Ne si evince che, se si avesse una stima a priori della distanza del target dalla

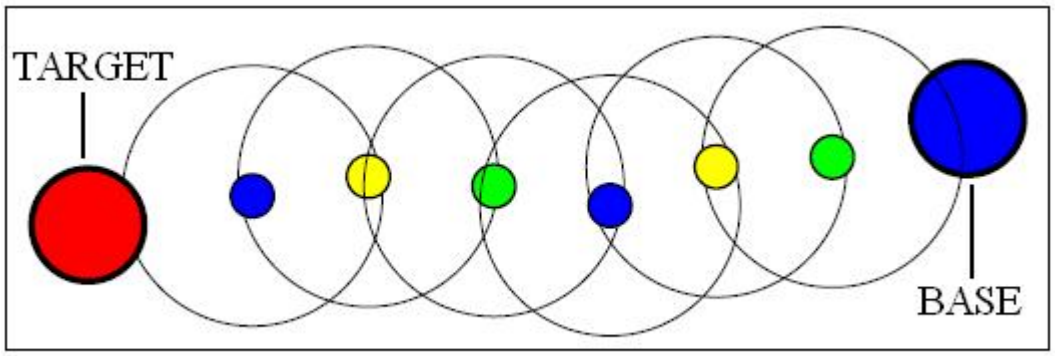

Fig. 4. Struttura a catena.

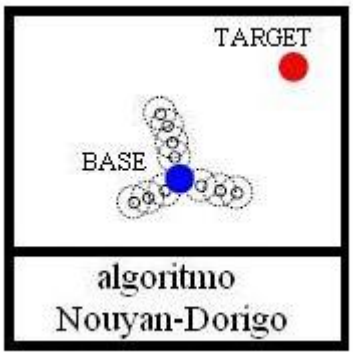

Fig. 5. Procedura di ricerca.

base, si potrebbe aggiustare accuratamente questo parametro in modo da ottimizzare il tempo di ricerca.

La semplicitá di questa soluzione algoritmica dona velocitá alle manovre di gestione dei robot, pur non garantendo la costante connettivitá fra le unitá e nemmeno il successo della ricerca entro un certo time out anche se il target si trova in posizione vicina alla base.

Per tale algoritmo si sono osservati un paio di accorgimenti interessanti. Il primo consiste nel mutare la direzione del vettore velocitá di un robot in caso di collisione: la nuova direzione sará semplicemente quella opposta al sensore di prossimitá che viene attivato durante la collisione. Il secondo

Fig. 6. Angolazione minima consentita tra gli agenti della catena.

accorgimento riguarda invece il carattere rettilineo delle catene da formare. Per cercare di mantenere le unitá di una catena il piú possibile in linea tra loro infatti gli autori hanno studiato una routine di allineamento che entra in azione se l'angolo fra i baricentri dei robot é inferiore a $120°$. In questo modo si riesce ad ottenere la massima lunghezza della colonna a paritá di robot che la compongono, e quindi una buona esplorazione.

Fatti tutti questi cenni preliminari, si vogliono considerare i risultati offerti dall'adozione di questo algoritmo mediante prove in simulazione. Come setup per ogni esperimento si é considerata un'arena quadrata di cinque metri di lato e quattro variabili da combinare: numero di robot a disposizione, distanza fra base e target, $P_{e\to c}$, $P_{c\to e}$. Per ogni combinazione si sono ripetuti cento esperimenti. Come misura di prestazione si é scelto il tempo di completamento della missione, ovvero il tempo entro cui la swarm riesce a creare un ponte e mantenere la connessione su di esso per almeno cento secondi. Il limite impostato per determinare il fallimento della missione é stato fissato a diecimila secondi e la distanza per riconoscere il target con la telecamera di un robot é di 80 cm. Per la quasi totalitá dei casi esaminati, cioé nei casi in cui $n_{robot} \in \{5, 10, 15, 20\}$ e $distanza \in \{0.6, 1.2, 1.8, 2.4, 3.0\}$, il limite appena definito viene oltrepassato solo nel 10% degli esperimenti, ottenendo quindi in generale un buon esito.

Ad ogni modo, per quanto riguarda la scelta ottima per i parametri $P_{e\to c}$ e $P_{c\to e}$ dalle simulazioni si ricava che questi dipendono dalla distanza base-target. Tale risultato conferma quanto si poteva logicamente attendere: per distanze brevi conviene mappare molte direzioni con piccole catene, richiedendo cioé alti valori per i due parametri e viceversa per distanze grandi l'unico modo di raggiungere il target é di formare colonne lunghe, impostando quindi un valore prossimo a zero per $P_{c\to e}$.

Per riassumere schematicamente i risultati degli esperimenti si puó fare riferimento ai grafici seguenti, che illustrano rispettivamente il minor tempo di completamento registrato ed il minor tempo di completamento normalizzato: quest'ultimo viene definito come il prodotto fra il tempo di completamento minimo e il numero dei robot utilizzati, e rappresenta cioé il tempo speso da ciascun robot fino al termine dell'operazione. Nell'immagine *Nouyan-Dorigo A* si puó notare come il tempo di completamento cresca in modo piú che lineare con l'aumento della distanza base-target, ma

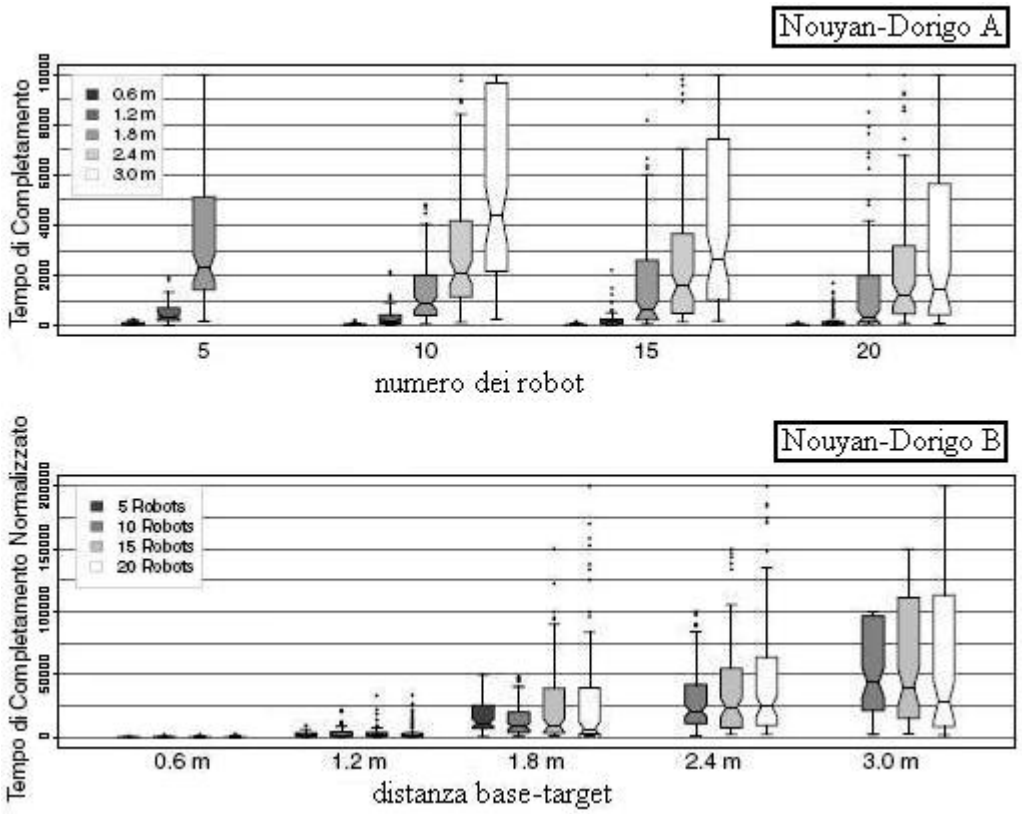

Fig. 7. Sintesi dei risultati dell'algoritmo Nouyan-Dorigo .

ció si spiega facilmente pensando che al crescere lineare della distanza corrisponde una crescita quadratica dell'area da esplorare. Il grafico *Nouyan-Dorigo B*, infine, offre una misura dell'efficienza del sistema, soprattutto in termini di scalabilitá. Sarebbe infatti opportuno che, a paritá di distanza, il tempo speso da ciascun robot fosse lo stesso, cosa che utilizzando questo algoritmo si riesce ad ottenere approssimativamente bene.

Un'idea alternativa che si puó considerare per questo algoritmo é di iniziare la formazione delle colonne dal target piuttosto che dalla base. In questo modo si puó supporre di avere una stima a priori della posizione della base e quindi della direzione da dare alle colonne. Sfruttando opportunamente questa informazione quindi si potrebbe elaborare una procedura piú performante come tempistica d'esecuzione.

Oltre ai due appena presentati, in letteratura si possono trovare altri approcci circa l'esplorazione di ambienti sconosciuti, ma con condizioni sperimentali e robot differenti da quelli scelti per questo progetto. Ad ogni modo, fra questi ce ne sono alcuni che hanno destato particolare intaresse per la specifica problematica affrontata o per semplice curiositá. Per esempio in [10] l'idea di base considerata é di suddividere l'ambiente di ricerca in strutture poligonali che vengono esplorate sequenzialmente dalla swarm di robot. Per l'esplorazione di ogni forma si prevede che ci sia un robot che procedere in avanscoperta e che successivamente, in caso di buon esito di questa ricognizione preliminare, avanzi tutto il resto della squadra. Il pregio di questa metodologia é la consistente riduzione dell'errore dell'odometro durante la procedura, ma il fatto che le unitá debbano stare perennemente vicine fra loro rallenta il processo di ricerca.

Un'altra linea di pensiero si trova in [1], in cui gli autori si pongono il problema di esplorare un'area costruendone gradualmente una mappa, per poi arrivare a distribuire i robot su di essa in modo tale che i loro sensori la coprano il piú possibile. L'approccio algoritmico presentato tiene anche conto del costo di raggiungimento delle locazioni necessarie per ottenere la miglior distribuzione sensoriale.

Altre tecniche simili invece assegnano la posizione di ciascun robot basandosi solo sulla minimizzazione del tragitto necessario per raggiungerla. In questo caso puó verificarsi che due robot vadano ad occupare la stessa locazione, con conseguente perdita in termini di copertura sensoriale. Appare evidente che un'opportuna distribuzione dei robot sull'area consente la localizzazione di un target che non sia piú semplicemente immobile, ma che sia libero di spostarsi su di essa. Inoltre, puó risultare molto utile in contesti pratici che richiedono la monitorizzazione di ambienti. Per rendersene conto basta infatti pensare a situazioni in cui si vuole mappare una zona con sensori di fumo per rilevare un principio di incendio o utilizzare informazioni su piú punti dello spazio in modo da creare un gradiente di temperatura che consenta di stimarne il nucleo ([3]). D'altra parte peró si osserva che una soluzione di questo tipo, per mantenere sotto controllo un'area in modo efficace e completo, richiede un numero di robot molto maggiore di quello necessario per effettuare la stessa

operazione con un'unica colonna rotante di robot. Tuttavia, pur considerando questo sacrificio economico, l'approccio con distibuzione sensoriale risulta anche intuitivamente piú veloce nel fornire risultati.

Si é ritenuta assolutamente interessante dal punto di vista culturale una recente estensione del PSO per simulare la ricerca di cibo da parte di un gruppo di animali ([2]). Quest'idea innovativa, per la quale si sono dovuti apportare alcuni accorgimenti notevoli all'algoritmo originale, prevede di affidare a ciasuna unitá un'energia. Al decrescere di quest'energia si fa piú importante l'esigenza di trovare una fonte di cibo. Quando un'unitá raggiunge tale sorgente, questa si ferma per un periodo rappresentante il tempo necessario per l'alimentazione, per poi ricominciare a muoversi. Appare evidente che l'ambiente che si vuole descrivere in questo caso ha un'aspetto tutt'altro che statico, in quanto necessariamente il cibo viene "consumato" dagli animali. É appunto attraverso questa dinamicitá che gli autori intendono dare una descrizione approssimativa della realtá naturale.

Ci si auspica per il prossimo fututo di trovare lavori che, basandosi su questo approccio, tengano conto dell'azione simultanea di piú apparati sensoriali per la ricerca(es.visivo e olfattivo), della possibilitá di morte di un'unitá nel caso di raggiungimento di una soglia di energia troppo bassa, e dell'interazione fra specie animali diverse.

In letteratura si puó trovare anche un modello per la ricerca in ambiente sconosciuto che trae spunto dal comportamento sociale delle formiche. Queste infatti, per attirare altre formiche verso una direzione fruttuosa per la ricerca di cibo, rilasciano al loro passaggio sostanze chimiche note come "ferormoni". Per quanto riguarda la situazione esaminata in questo progetto, non é affatto ovvio poter disporre di unitá che consentano di tradurre questo ultimo concetto in ambito robotico. Tuttavia si puó pensare ad un parallelismo fra l'idea di tracciare un territorio con ferormoni e la struttura a catene caratteristica del nostro algoritmo. Infatti, le catene di dimensioni maggiori esercitano un potere attrattivo sulle colonne piú piccole(o sulle unitá singole), le quali tendono a fondersi con esse per raggiungere collettivamente un risultato migliore. In entrambi i casi quindi ci si rifá alla nozione di *feedback positivo*, a sottolineare la natura sociale che sta alla base del filone della *swarm robotics*.

III. Approccio CF: Connectivity-First

Il primo approccio alla risoluzione del problema in esame é stato di tipo *Connectivity-First*, cioé una ricerca della sorgente di segnale effettuata mantenendo, iterazione dopo iterazione, la comunicazione fra gli agenti. Il vantaggio principale di questa filosofia risolutiva risiede nel fatto che, una volta identificato il segnale del target, non é necessario curarsi di istituire il ponte radio fra sorgente di segnale (SS) e Base Station (BS), dato che la connettivita' é gia' stata garantita ad ogni passo algoritmico; ovviamente, cosí facendo, il carico computazionale viene spostato sulla fase di ricerca che, come vedremo in seguito, risultera' essere particolarmente onerosa.

L'algoritmo CF elaborato, nonostante l'elevata complessita', presenta diversi aspetti di interesse:

- analisi ordinata dello spazio, che viene indicativamente coperto da cerchi concentrici alla BS via via di raggio maggiore;
- massimo sfruttamento delle capacitá di copertura del network.

Con questo tipo di soluzione ogni nodo mantiene la connettivitá con la BS dipendendo da qualche altro agente: a tal proposito si fará in modo che ogni robot si allontani il piú possibile da quelli con cui comunica in modo tale da coprire la piú ampia porzione di spazio possibile. La difficoltá maggiore risiederá nel fatto che ogni movimento di un robot, che dipenderá da un altro per comunicare con la base, dovrá essere coordinato con quello del suo 'superiore'.

A. Ipotesi iniziali

Vanno subito puntualizzate alcune ipotesi semplificative adottate.

1) Agenti puntiformi: si suppongono i nodi privi di una dimensione fisica. Questa ipotesi é stato necessaria per trascurare dall'analisi tutte le complicazioni derivanti da possibili scontri fra gli agenti mobili.

2) Analisi bidimensionale: tutte le considerazioni fatte nella fase teorica e simulativa di questo articolo sono state fatte considerando i robot agenti in un piano, senza curarsi delle complicazioni dovute all'introduzione della terza dimensione. Una fattore di ulteriore complessitá potrebbe essere ad esempio il considerare le differenze portate da sensori e trasmittenti posti ad altezze diverse fra loro.

3) Nodi identici: essenziale nei sistemi autonomi é supporre di aver a disposizione agenti fra loro identici in ogni termine; solo in questo modo si puo' pensare ad algoritmi di ricerca paritari, dove ogni robot ha gli stessi compiti dei colleghi; l'unico fattore che distingue un nodo dagli altri é l'identificativo $ident$, valore numerico assegnato biunivocamente ad ogni robot, con la semplice funzione di 'etichetta'.

4) Segnali isotropi: si ipotizza che ogni oggetto considerato in grado di trasmettere informazioni (la BS, la SS e l'intero network robotico) consentano di avere segnali della medesima intensitá per ogni punto equidistante dalla sorgente; cosí facendo di fatto le curve di livello di intensita' dei vari segnali non sono altro che circonferenze aventi come centro la sorgente emissiva (Fig. (8)): e' sotto quest'assunzione che, data la forma delle curve di livello, di seguito si parlerá di *raggio* di visibilita' r_v di un segnale per indicare la zona entro cui il segnale viene percepito. Quella appena illustrata é un'ipotesi molto forte, difficilmente traducibile in un contesto pratico, tuttavia risulta necessaria per l'algoritmo proposto; quest'assunzione verra'in seguito rilassata negli algoritmi di tipo SF.

5) Distanza deducibile dall'intensita': si suppone che ogni nodo riesca a desumere con correttezza la distanza relativa da una sorgente, a partire dalla potenza del segnale ricevuto.

6) Assenza riflessione e rifrazione: tali fenomeni, molto relativi nel campo dei segnali, non sono stati considerati. Va considerato come, di fatto, quest'ulteriore ipotesi non semplifichi ulteriormente il problema visto quanto giá assunto ai punti precedenti.

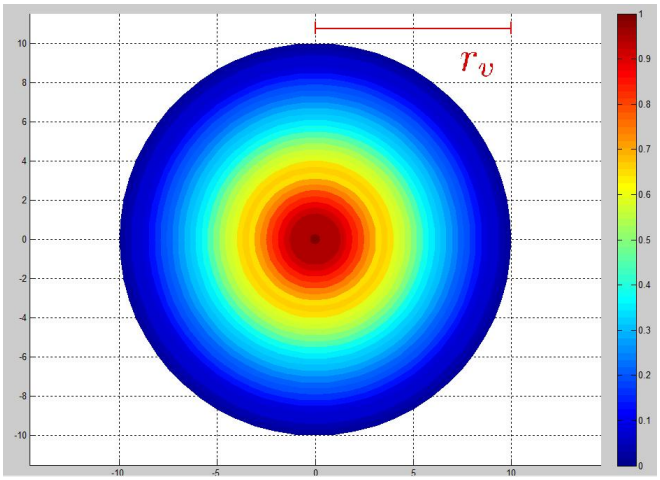

Fig. 8. Intensita' di un segnale 'uniforme'

7) Segnali della stessa ampiezza: BS, SS vengono considerati di fatto come due nodi identici a quelli mobili, solo che sono fissi; verranno nel seguito indicati come nodo 0 e nodo $N+1$. In quest'ottica, e riassumendo le ipotesi precedenti, tutti segnali emessi dall'insieme V di agenti, dalla BS e dalla SS hanno le stesse caratteristiche di ampiezza ed intensita': indicheremo nel seguito con r_v (raggio di visibilitá) il raggio di copertura massima di un generico segnale.

8) Movimento semplificato: si suppone che i robot riescano a girare su se' stessi senza bisogno di alcun angolo di sterzo; si assume che in un passo iterativo (di cui chiariremo in seguito il significato) ogni nodo riesca a spostarsi in maniera rettilineo al massimo di una quantita' pari a δ ed abbiano la possibilitá di ruotare di 180° su loro stessi.

9) Sincronia: in varie fasi dell'algoritmo si ipotizza che ogni robot riceva ed invii segnali al medesimo istante. Ovviamente tale ipotesi é una semplificazione che richiede in pratica una certa accuratezza per essere implementata;

10) Odometro: si suppone infine che ogni agente robotico possegga uno di questi strumenti e sia pertanto in grado di quantificare la distanza de esso percorso.

B. Ambiente simulativo

Prima di entrare nel merito dell'algoritmo CF vengono sinteticamente presentati gli elementi dell'ambiente creato per le simulazion. L'ambiente viene creato attraverso le funzioni[1]:

- *creazione della Base Station*: si colloca la stazione di partenza nell'origine di un fittizio sistema di riferimento;
- *creazione dei veicoli*: si crea un network V^2 composto da N nodi, con N arbitrario. I veicoli sono gli 'oggetti simulativi' piu' complessi, avendo essi una direzione, un campo di visibilita' e diversi altri attributi: al momento della creazione vengono allocati diversi campi, posizioni di memoria, il cui significato verra' chiarito nel proseguio della presentazione dell'algoritmo. Come accennato in precedenza, fra i vari campi c'é quello $ident$: ad ogni nodo viene assegnato un identificativo numerico da 1 ad N. Gli agenti al momento della creazione vengono collocati in posizioni casuali all'interno del campo di visibilitá della BS, con un'angolazione anch'essa del tutto aleatoria.
- *creazione della Source Signal*: si crea il target, sorgente d'informazione di collocazione ignota. Affinché il problema creato sia rispettivamente non banale ma comunque risolvibile, la SS viene collocata sufficientemente distante dalla BS da non essere identificata all'istante iniziale della simulazione, ma comunque entro una zona raggiungibile dal network robotico. Indicando con $(SS.x, SS.y)$ la posizione della SS rispetto al sistema di riferimento fittizio centrato nella BS, quanto detto sopra si traduce in:

$$2 * r_v \ \leq \ \sqrt{SS.x^2 + SS.y^2} \ \leq \ (N+1) * r_v,$$

$(N+1) * r_v$, di fatto, definisce la massima distanza entro la quale una sorgente ignota puo' essere scoperta; verra' dimostrato infatti come l'algoritmo proposto consenta di sfruttare al massimo le capacita' dell'insieme di nodi costituendo ponti di comunicazioni con nodi allineati, il che consente di coprire la massima distanza possibile.

Si suppone che i nodi aggiornino i segnali periodicamente, mentre il movimento degli stessi avvenga di fatto in maniera praticamente continua. Data l'assenza di una unitá computazionale che gestisca la ricerca nella sua complessitá, sono le unitá stesse a verificare le condizioni attuali del sistema ed a modificare il proprio stato in maniera tale da portar correttamente avanti la procedura di ricerca. I cambiamenti fra le vari fasi del movimento vengono disciplinati in base ai segnali scambiati con gli altri nodi, il che significa che sono le interazioni con gli altri agenti del sistema a determinare il movimento degli altri robot.

Proprio per questo motivo é necessario per ogni agente $V(x)$ che con regolaritá controlli i segnali ricevuti ed il segnale inviato vengano aggiornati: ogni robot deve eseguire pertanto periodicamente una procedura di *aggiornamento dei segnali*. A tal proposito in seguito si parlerá di passo iterativo intendendo il periodo di tempo intercorso fra un aggiornamento dei segnali ed il successivo.

Infine la routine principale di ogni nodo é quella alla quale é preposto il movimento: *muovi veicolo*. Il moto, per essere corretto, deve essere eseguito necessariamente in seguito alla procedurra di aggiornamento del segnale.

C. Fase iniziale

Come gia' accennato in precedenza gli agenti robotici cominciano la loro opera di ricerca in posizioni casuali all'interno del raggio di visibilitá della BS. Ovviamente lo scopo sará quello di coprire zone del piano non ancora esplorate pur mantenendo la comunicazione con la base: primo obbiettivo dei robot sará infatti quello di spostarsi verso una zona ad intensitá minima del segnale della BS; viene assegnato un valore particolare alla circonferenza C_{r_s}[3]

[1]Le funzioni descritte sono routine fittizie. I riferimenti alle reali funzioni MATLAB che implementano tali scopi sono riportati in Appendice

[2]In seguito si utilizzerá la seguente notazione: V per indicare un insieme di agenti, mentre $V(x)$ per indicare la singola unitá

[3]D'ora in poi indicheremo per semplicitá di notazione C_{r_x} le circonferenze di raggio r_x.

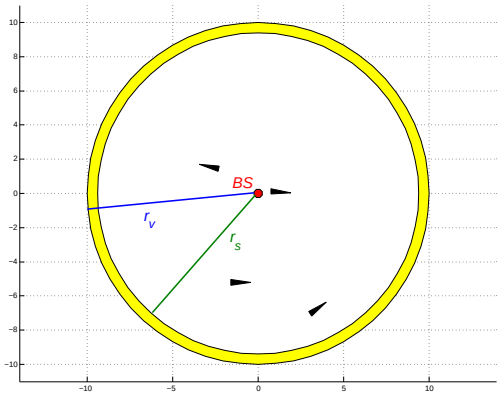

Fig. 9. Condizioni iniziali dell'algoritmo CF

centrata nella BS di raggio r_s, detto raggio di sicurezza, cosí definito:

$$r_s = r_v - \delta,$$

viste le ipotesi fatte in precedenza sulle limitazioni del movimento degli agenti, ogni nodo collocato all'interno di $C(r_s)$ non puo' in un passo uscire dalla circonferenza di raggio r_v, e perdere quindi la connettivitá. In (Fig. (9)) é riportato un esempio di condizione iniziale di un sistema con $N = 4$ nodi: la forma triangolare rappresenta l'angolazione iniziale dei vari robot. I nodi hanno come primo compito quello di allontanarsi dalla BS sino ad arrivare nei pressi della $C_{r_s}(BS)$; tale allontanamento avviene cercando di seguire una retta a gradiente decrescente d'intensitá di $S(BS)$[4]: tale scelta tecnica, che complica il movimento introducendo la complessitá del calcolo del gradiente, ha il vantaggio di diminuire la distanza per arrivare sulla circonferenza $C_{r_s}(BS)$ rispetto ad un'altra traiettoria e consente di intersecare la suddetta circonferenza in maniera abbastanza perpendicolare, il che si rivelerá un vantaggio nella successiva fase di movimento dell'algoritmo.

D. Calcolo del gradiente ed allontanamento dalla BS

Ovviamente i nodi non posseggono gli strumenti necessari per un calcolo perfetto del gradiente di $S(BS)$ ed, in generale, di un qualunque segnale: tuttavia sfruttando la possibilitá di leggere l'intensitá di $S(BS)$ in diversi punti nelle vicinanze di quello di partenza é possibile dedurre una buona stima della struttura del campo di comunicazione.

In (Fig. (10)) é schematicamente illustrato il processo di stima del gradiente. Il nodo avanza lungo la retta imposta dalla sua angolazione di una quantitá ϵ, del tutto arbitraria, e ruota di $90°$ in senso orario; in questo momento il nodo alloca in memoria lo spazio necessario per salvare un numero $ACCU$ di intensita' di $S(BS)$ calcolate in punti diversi del piano. L'agente robotico a questo punto percorre una circonferenza C e salva in memoria il valore ricevuto dell'intensitá di segnale in $ACCU$ punti equidistanziati della stessa C; nell'immagine a sinistra di (Fig. (10)) é riportato il movimento 'fotografato' ad intervalli regolari del nodo durante tale processo, nei momenti in cui viene effettuata la rilevazione del segnale: la lettura dell'intensitá di S_{BS} con tale procedimento viene eseguita sui vertici di un poligono di $ACCU$ lati di dimensione ϵ inscritto nella circonferenza C (in grigio la posizione in cui viene rilevata la minima intensitá di segnale). Va sottolineato come la scelta di ϵ sia ancora una volta del tutto arbitraria e come C non sia precisamente centrata in $(V.x, V.y)$ all'istante di inizio del calcolo del gradiente; dato che per soddisfare tale specifica, non necessaria ai fini dello scopo del processo, sarebbe stato necessario supporre gli agenti robotici in grado di effettuare calcoli trigonometrici, per non appesantire le ipotesi del modello utilizzato é stato scelta una soluzione un po' meno raffinata: ϵ é stato scelto sufficentemente piccolo affinché non si perda la connettivitá dalla BS effettuando il calcolo del gradiente.

$ACCU$ di fatto esprime il livello di accuratezza con cui la stima del gradiente viene calcolata: all'aumento di tale parametro aumenta la precisione con la quale viene stimata la direzione ad intensitá minima a discapito ovviamente della lentezza del processo. In (Fig. (10)) a destra viene riportata la direzione del gradiente negativo stimata dal nodo rispetto a quella esatta: δ esprime l'errore nel calcolo dello stesso.

Ovviamente la procedura illustrata in precedenza é del tutto valida anche nel caso in cui sia necessario calcolare una stima della direzione di gradiente crescente: al posto di ruotare verso la posizione dove é stata registrata la minima intensitá di segnale, il nodo in questione si direzionerá verso il punto dov'é stata rilevata un'intensitá di segnale maggiore.

Stimato il gradiente, il nodo si allinea con la direzione ottima trovata e procede in linea retta allontanandosi dalla BS: ad ogni passo iterativo l'agente avanza in maniera rettilinea della massima distanza possibile in un passo, cioé δ. Tale allontanamento procede fintantoché l'agente non si avvicina alla zona marginale del campo di $S(BS)$; si fa avanzare il nodo sino a che non raggiunge la prossimitá di $C_{r_s}(BS)$: a questo punto unitá robotica ruota di un angolo retto (in senso orario od antiorario con la medesima probabilitá); cosí facendo l'agente si colloca in una posizione pressoché tangente a $C_{r_s}(BS)$: l'errore rispetto alla tangente vera e propria é pari a γ. Passo successivo dell'algoritmo sará quello di far muovere gli agenti robotici lungo $C_{r_s}(BS)$: nel movimento ideale si vorrebbe che ad ogni passo l'agente robotico fosse tangente di $C_{r_s}(BS)$; si intuisce ora la necessitá del calcolo del gradiente come step iniziale dell'algoritmo: maggiore é la precisione del calcolo minor é lo scostamento γ dalla tangente di $C_{r_s}(BS)$.

E. Movimento circolare di un singolo nodo

Una volta raggiunta la $C_{r_s}(BS)$, i nodi percorrono la circonferenza per verificare la presenza della SS in posizioni del piano non ancora esplorate: in questa maniera se tutti i punti della $C_{r_s}(BS)$ venissero toccati da almeno un robot, se il target distasse dalla BS non piú di $r_s + r_v$, verrebbe localizzato.

Supponendo di conoscere il raggio di curvatura necessario ad effettuare una circonferenza, sarebbe sufficiente fornire tale

[4]In seguito verrá utilizzata la notazione $S(X)$ per indicare il segnale proveniente da nodo X, includendo anche i segnali della BS e della SS.

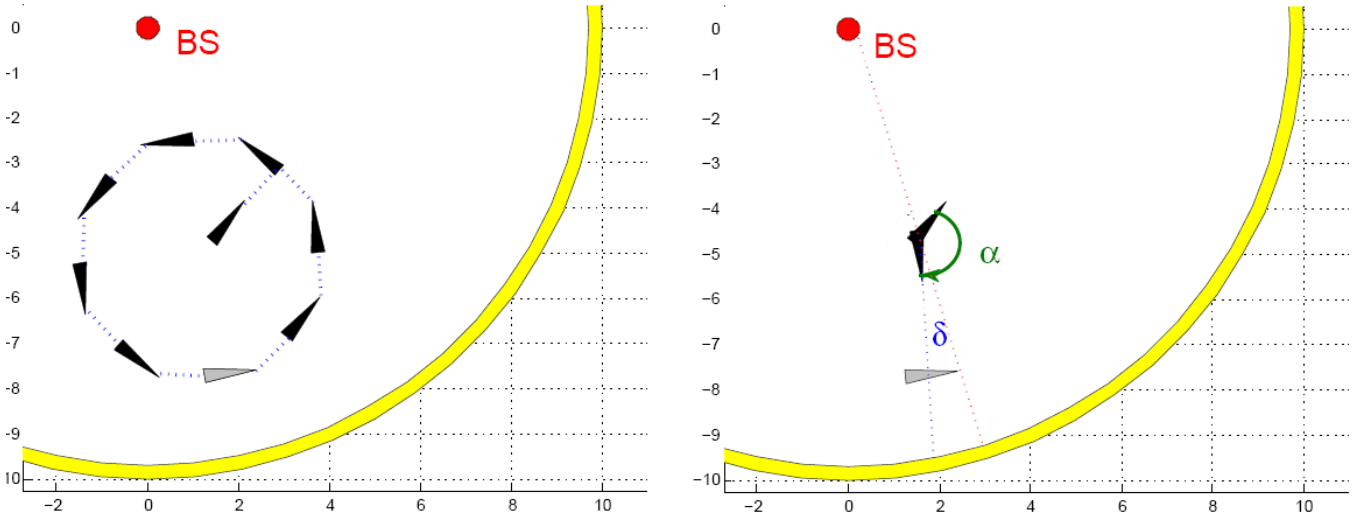

Fig. 10. Calcolo del gradiente

parametro all'unitá robotica per far sí che questi segua la traiettoria desiderata; per approcciarsi al problema in questa maniera, tuttavia, sarebbe necessario essere certi che l'agente cominci il movimento circolare da una posizione perfettamente tangente alla $C_{r_s}(BS)$, il che risulta essere molto difficile da realizzare per quanto argomentato nella sezione precedente. Per questi motivi si é preferito scegliere un'implementazione piú robusta fatta di correzioni successive in base all'intensitá di $S(BS)$ rilevata ad ogni passo iterativo; il generico agente $V(k)$, anziché cercare di seguire perfettamente la circonferenza, deve rimanere all'interno di una regione, detta *zona di correzione*, cosí definita[5]

$$r_s - \delta_{MIN} \leq d(V(k), BS) \leq r_s + \delta_{MIN}.$$

Nel caso l'unitá mobile rilevi che l'intensitá di $S(BS)$ é troppo elevata o troppo bassa da far sí che uno dei due limiti non venga rispettato, viene effettuata per una correzione di traiettoria: verso l'interno della circonferenza se $S(BS)$ é troppo lieve, verso l'esterno viceversa; ovviamente per effettuare la correzione nella direzione esatta é stato necessario supporre che il nodo fosse a conoscenza del verso con cui stava descrivendo la $C_{r_s}(BS)$. Il termine correttivo K applicato é di tipo proporzionale all'errore di posizione $\delta_{x,y}$,

$$K = \delta_{x,y} * G, \tag{1}$$

con G regolato ad hoc per mantenere appunto la giusta traiettoria.

E' stata contemplata anche l'ipotesi in cui un nodo continui a girare lungo la $C_{r_s}(BS)$ senza trovare il target, oppure senza incontrare altri nodi (situazione che vedremo dá luogo ad altre tipologie di movimento): a tal proposito é stata implementata una procedura per far sí che l'unitá robotica cambi verso di percorrenza della $C_{r_s}(BS)$ una volta che abbia giá toccato tutti i punti della circonferenza. E' possibile infatti che piú

[5]D'ora in poi, per indicare la distanza fra due oggetti x ed y (nodi, BS e SS), verrá utilizzata la notazione $d(V(x), V(y))$

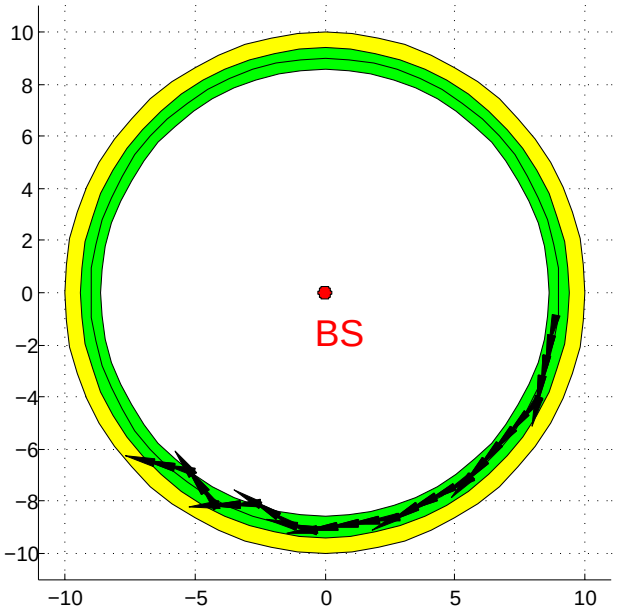

Fig. 11. Traiettoria di un agente durante il movimento circolare: in verde la zona di correzione

nodi stiano eseguendo il movimento circolare nello stesso verso ed alla medesima velocitá: in questa tipologia di situazione tali unitá non si incontreranno mai, non portando alcun vantaggio alla ricerca e creando una situazione di 'stallo'. Per ovviare a tali problemi si sfrutta il fatto che, senza considerare gli spostamenti dovuti alle correzioni, una volta che un robot compie un giro completo sulla circonferenza, esso anche ruota di 360° su sé stesso: viene utilizzata tale corrispondenza per dedurre se il nodo ha giá percorso interamente $C_{r_s}(BS)$. Ogni agente ha la capacitá di calcolare e mantenere memoria dell'angolo di rotazione eseguito durante ogni movimento: ogni unitá $V(x)$ tiene aggiornato un contatore $V(x).cnt$ che somma, passo dopo passo, l'angolo totale di rotazione. La

condizione

$$V(x).cnt = (360 + \beta)^\circ, \quad (2)$$

equivale al completamento di un giro da parte della $V(x)$; β é un angolo positivo, regolabile dall'utente per tener conto dell'angoli supplementari di rotazione introdotti dalle correzioni. Quando $V(x)$ verifica la (2) invia un segnale d'allarme alla BS: se $V(x)$ é il primo nodo che invia tale richiesta, la BS risponde con un comando di inversione del senso di percorrenza; allo stesso tempo viene allegato a $S(BS)$ un comando di azzeramento del contatore $V.cnt$ a tutti gli altri agenti. In questo modo $V(x)$ ha la possibilitá di incontrare i colleghi agenti che si stavano muovendo nel medesimo verso. Ogni altro nodo che verifica la condizione di 360° successivamente a $V(x)$ non ricevono dalla BS alcun comando d'inversione e proseguono nella direzione di percorrenza.

La procedura appena descritta ricopre inoltre il fondamentale compito di verificare l'eventuale fine senza successo dell'algoritmo: nel caso in cui un nodo abbia giá compiuto due giri attorno alla circonferenza dopo aver invertito senza incontrare nessun altro nodo, ció significa, come vedremo, che non é ulteriormente possibile estendere la ricerca. Verificata tale condizione il nodo comunica con la BS che il processo di ricerca si é concluso con esito negativo: in questo modo l'algoritmo ha una procedura di chiusura anche nel caso in cui il target non venga identificato.

F. Scontro ed accordo

L'idea per esplorare porzioni di piani distanti dalla BS piú di $r_s + r_v$ mantendo comunque la connettivitá, é quella di creare catene di nodi che si muovano in maniera sincronizzata (in maniera simile a quanto fatto in [8]): ogni catena avrá come primo nodo la BS, unico nodo fisso, mentre tutti gli altri si muoveranno in maniera coordinata; in tal modo, a prezzo di una pesante complicazione del movimento d'esplorazione, é peró possibile estendere la rilevazione di fonti di segnale entro una zona molto piú ampia pari a

$$r_{ris} = N * r_s + r_v;$$

r_{ris} dá una misura della capacitá risolutiva dell'algoritmo proposto: all'aumentare della superficie che si desidera esplorare é possibile scalare il problema impegnando un numero maggiore di agenti nella ricerca. Da questo aspetto si evince uno dei punti di forza di questo approccio: tale approccio sfrutta al massimo le possibilitá del network spingendo la ricerca alla massima distanza possibile con le risorse disponibili. Si parlerá di catene, quindi di strutture gerarchiche: per semplificare la descrizione delle seguenti fase dell'algoritmo verranno presi in prestito diversi termini di carattere 'geneologico', associando le gerarchie di posizione dei nodi sulla catena con analoghi rapporti gerarchici intercorrenti in una famiglia:

- i termini *padre* e *figlio* saranno utilizzati per riferirsi rispettivamente al nodo immediatamente precedente ed a quello immediatamente successivo in un catena (con precedente si intende piú vicino alla BS e successivo piú lontano);
- *capostipite* di una catena é il primo nodo di una catena, quello che comunica direttamente con la BS
- la *discendenza* di un nodo é l'insieme di tutti i nodi successivi nella catena;
- con *adozione* si indicherá il processo per cui un nodo o, piú in generale, una catena si concatenano con un'altra, per dar luogo a strutture di lunghezza maggiore;

L'algoritmo CF si presuppone di costituire via via catene sempre piú profonde, di fatto si partirá dalla costituzione di catene di 2 nodi procedendo fino ad avere, al limite, una singola catena contenente tutti gli N agenti; in tal modo si intende analizzare il piano incognito in maniera 'ordinata': porzioni di piano piú vicine alla BS sono quasi sempre analizzate prima di altre piú lontane. Si é scelto di far iniziare al sistema la formazione di catene nel momento in cui due nodi, $V(x)$ e $V(y)$, percorrendo la $C_{BS}(rs)$ con versi opposti, si scontrano: ogni nodo, infatti, proseguendo lungo la propria direzione esaminerebbe, almeno inizialmente, un porzione di spazio giá sondata dall'altro agente, il che sarebbe infruttuoso per la risoluzione della ricerca; data l'ipotesi di nodi puntiformi, si é considerata come condizione di scontro una vicinanza critica di $V(x)$ e $V(y)$. I robot che stanno compiendo il movimento circolare ad ogni passo iterativo eseguiranno una routine di verifica degli scontri; nelle funzioni caricate in ogni robot dev'essercene una (*verifica scontro*) preposta, come chiaramente espresso dal nome, ad identificare l'incrocio con un altro agente: in caso in cui la condizione di scontro venga verificata con piú di un nodo allo stesso istante, tale funzione discrimina, in base ad un criterio di maggior vicinanza, con quale dei vari robot effettuare il processo di 'adozione'.

Nel momento in cui il nodo $V(x)$ verifica la condizione di scontro trasmette, assieme agli altri dati, un segnale d'allarme che indica la volontá di costruire una catena con $V(y)$: se anche $V(y)$ trasmette il medesimo segnale viene di fatto stabilito un accordo fra i due nodi, dove si stabilisce quale dei due sará il padre e quale il figlio; a discriminare quale dei due succederá all'altro sará il campo *ident*: il nodo con l'identificativo piú basso diverrá il padre; tale scelta é del tutto arbitraria e serve solo per definire in maniera univoca come risolvere situazioni come questa, in cui, partendo dalle paritarie condizioni di partenza, ambo i nodi, o le catene, si collochino in una futura struttura dopo lo scontro. Tale attivitá di riorganizzazione delle strutture in gioco dopo una collisione viene disciplinata da una routine, riprendendo il gergo 'famigliare', detta *genealogia*, i cui criteri di scelta e peculiaritá varie verranno illustrati meglio successivamente quando verrá presentato il caso di fusione fra catene costituite da piú nodi. Indicheremo da questo momento $V(f)$ il nodo padre e $V(s)$ il figlio.

G. Adozione fra due singoli nodi

Una volta stabilito l'accordo fra due nodi $V(f)$ e $V(s)$, d'ora in poi questi effettueranno la ricerca in maniera sincronizzata: si verrá a costituire una catena con $V(f)$, nodo padre, che continuerá a muoversi lungo la $C_{r_s}(BS)$, mentre $V(s)$, nodo figlio, si muoverá lungo $C_{2*r_s}(BS)$ mantenendo ad ogni passo la connettivitá con il padre; l'esplorazione quindi si estenderá a porzioni di piano distanti $2*r_s+r_v$ dalla BS: si passerá dall'avere un insieme di singoli nodi che ruota attorno

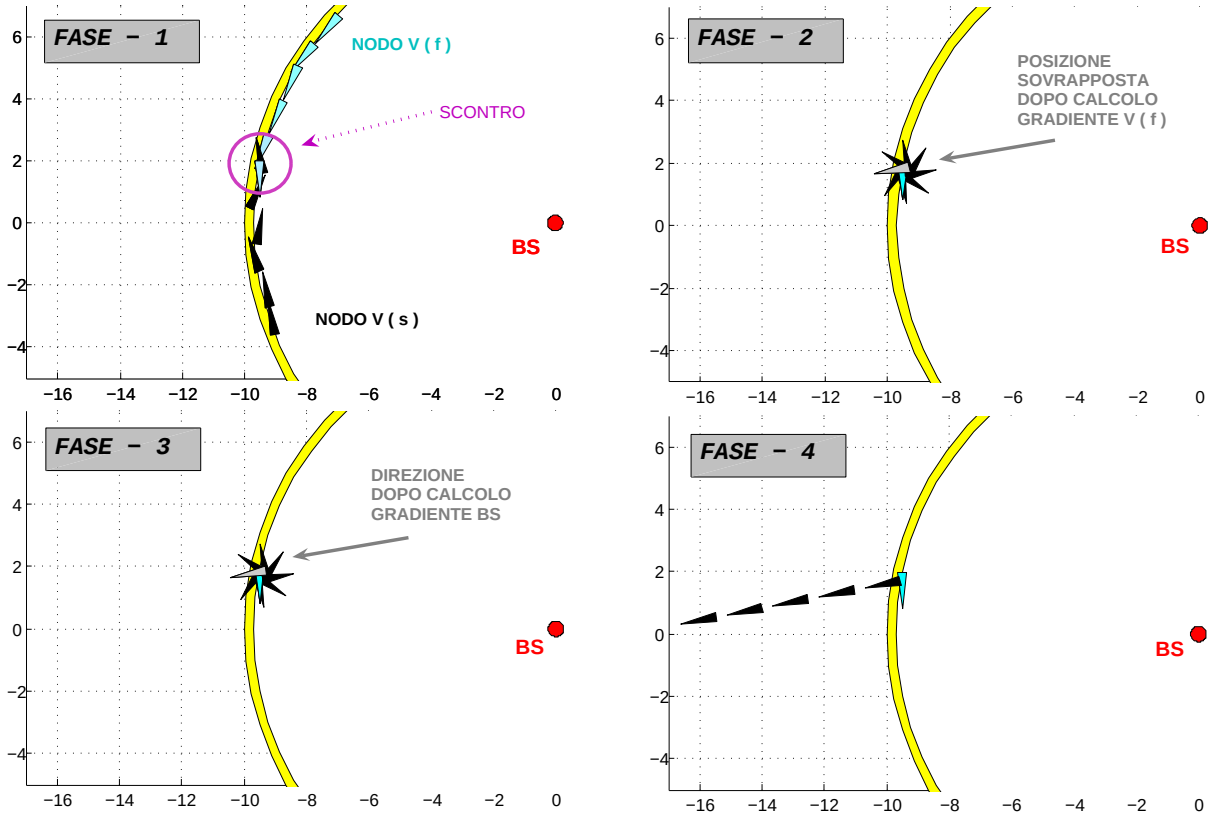

Fig. 12. Traiettoria di un agente durante il movimento circolare: in verde la zona di correzzione

alla BS ad avere anche delle catene di piú nodi nel processo di ricerca.

Affinché l'algoritmo sia il piú possibile performante é necessario che $V(f)$ e $V(s)$ siano perfettamente allineati con la BS: solo in questo modo infatti é possibile arrivare ad esaminare punti distanti sino a $2 * r_s + r_v$ dalla BS come desiderato. Tale procedura di allineamento viene fatta lasciando immobile $V(f)$ e posizionando il figlio nell'intersezione Z fra la retta passante da BS e $V(f)$ e la $C_{r_s}(V(f))$. Tale procedura risulta essere molto piú elaborata rispetto a quella iniziale di posizionamento sulla $C_{r_s}(BS)$: $V(s)$ non deve solo porsi in una zona a minima intensitá di $S(V(f))$, ma in un preciso punto, il che non puó essere risolto con un semplice calcolo del gradiente cosí come fatto in precedenza. Per ovviare a questa problematica viene implementata una procedura alternativa in quattro passi, schematicamente illustrata in figura (Fig. (12)):

1) $V(s)$ calcola il gradiente di $S(V(f))$ e deduce la direzione di massima intensitá del segnale;
2) si fa avvicinare il piú possibile $V(s)$ a $V(f)$ (giá vicini in conseguenza allo scontro) fino idealmente a sovrapporli;
3) anziché calcolare nuovamente il gradiente di $S(V(f))$, che direzionerebbe $V(s)$ verso un qualunque punto della $C_{r_s}(V(f))$, $V(s)$ esegue la procedura di calcolo dell'intensitá di $S(BS)$: i due nodi coinvolti nello scontro sono di certo ancora in comunicazione con la base a questo punto dell'algoritmo ed il calcolo di $S(BS)$ pertanto puó sicuramente essere eseguito. Il vantaggio di conoscere il gradiente di $S(BS)$ viene dal fatto che, nella nuova posizione raggiunta dopo il passo (2), $V(s)$ si trova infatti a dover andare esattamente nella direzione opposta a dove si trova BS: per avere quindi la corretta angolazione in cui muoversi $V(s)$ ruota verso la posizione in cui ha precedentemente calcolato la minima intensitá di $S(BS)$;
4) $V(s)$ avanza in maniera rettilinea fino a raggiungere $C_{r_s}(V(f))$ nel punto Z;

Da questa procedura partono le prime critiche nei confronti dell'approccio CF. L'allineamento di due oggetti mobili lungo la stessa retta richiede pesanti ipotesi sulle capacitá computazionali del network robotico a disposizione per avere precisione in ogni fase della procedura: complicazioni che non rientravano nel modello impostato nella nostra analisi. Innanzitutto al punto (2) sarebbe necessaria una sovrapposizione perfetta dei due nodi $V(f)$ e $V(s)$, che necessiterebbe del calcolo esatto del gradiente, non di una stima; va anche sottolineato come nel caso reale, dove non si puó ovviamente prescindere dal diametro fisico degli agenti robotici, la sovrapposizione degli agenti deve necessariamente sostituirsi con una procedura piú complicata: ad esempio, il nodo figlio dovrebbe essere costretto a girare attorno al padre per il calcolo del gradiente di $S(BS)$. La fase (3) del procedimento sopra descritto pertanto comincia da un'erronea posizione; inoltre, la seconda stima del gradiente, quello riferito ad $S(BS)$, introduce un secondo errore: ne consegue che il movimento rettilineo al punto (4) difficilmente porterá $V(s)$ ad essere allineato con $V(f)$ e la BS. Una volta posizionatosi su $C_{r_s}(V(f))$ il nodo $V(s)$ crede di essere allineato con gli

altri due nodi ed avendo perso la connettivitá con la BS non dispone di altri segnali con i quali confrontarsi.

Si suppone comunque di essere in grado di far posizionare $V(s)$, anche se non esattamente in Z, comunque nelle sue vicinanze: una volta verificato che $S(V(f))$ ha intensitá tale trovarsi ad una distanza r_s da esso, $V(s)$ associa al proprio segnale un bit di allarme, nel quale indica il raggiungimento della collocazione desiderata; una volta che $V(f)$ leggerá tale segnale commuterá il proprio stato dalla precedente condizione idle alla successiva fase di movimento sincronizzato con il figlio. Il verso di percorrenza di $C_{r_s}(V(f))$ sará pertanto lo stesso di $V(f)$ prima dello scontro.

H. Movimento circolare di una catena di nodi

Una volta instituita la catena comincia la rotazione della stessa attorno alla BS. E' necessario d'ora in poi, affinché $V(s)$ non porta la connettivitá con $V(f)$, che tale allineamento venga mantenuto ad ogni passo. $V(s)$ dipende totalmente da $V(f)$ per rimanere connesso al network, mentre di fatto il padre si comporta muovendosi lungo la $C_{r_s}(BS)$ esattamente come nella fase precedente in cui non aveva alcun figlio. L'unica cosa che viene modificata nel moto circolare di $V(f)$ é la velocitá: $V(s)$ deve descrivere una circonferenza due volte piú grande di quella descritta dal padre, tuttavia la massima distanza δ percorribile in un passo rimane invariata; é pertanto necessario che l'agente $V(f)$ rallenti il proprio passo in modo tale da consentire al figlio di muoversi senza andare fuori dal raggio comunicativo.

Nel caso di catene con numero di agenti maggiore, il ragionamento appena esposto viene semplicemente scalato: ogni agente $V(k)$ modera la velocitá d'azione in modo tale da percorre in un passo iterativo la distanza

$$\frac{\delta}{1+V(k).num_{SON}},$$

dove il parametro $V(k).num_{SON}$ indica la lunghezza della discendenza di $V(k)$. É necessario quindi che ogni nodo mantenga in memoria il parametro $V(k).num_{SON}$ per disciplinare il proprio movimento quando v'é necessitá di sincronia con altre unitá robotiche.

Il movimento che $V(s)$ deve compiere é del tutto analogo: esso deve sempre compiere un moto circolare lungo la $C_{r_s}(V(f))$, ma dato che anche il padre é in movimento quello che di fatto avviene é che segua la circonferenza immobile $C_{2*r_s}(V(f))$. Mentre la velocitá quindi é la stessa del moto circolare descritto nella sezione (III-E) é l'angolo di sterzo ad essere dimezzato. Tale differenza viene implementata facendo ridurre del 50% le correzzioni introdotte ogni qualvolta $S(V(f))$ ha un'intensitá diversa da quella attesa: in questo modo la circonferenza descritta viene raddoppiata dato che le variazioni di sterzata sono meno ampie.

In una catena con una generica profonditá quindi ogni nodo $V(k)$ non solo dovrá regolare la propria velocitá in base alla lunghezza della propria discendenza, ma dovrá inoltre dosare le correzzioni di angolatura in base alla propria profonditá $V(k).num_{FATHER}$. Riprendendo l'equazione (1), il guadagno del controllore proporzionale si modifica

$$\frac{G}{1+V(k).num_{FATHER}},$$

dato che ad ogni livello il rapporto fra l'ampiezza della circonferenza percorsa e $C_{r_s}(BS)$ aumenta proporzionalmente alla profonditá.

I. Fusione fra due catene di nodi

In questa fase dell'algoritmo é possibile che nodi o catene di due nodi incrocino la propria traiettoria su $C_{r_s}(BS)$ con altre; piú in generale nelle varie fasi dell'algoritmo sará possibile che catene composte da N_W nodi si incrocino con catene di N_L: tali situazioni di 'scontro' vengono disciplinate formando un'unica catena di $N_W + N_L$ agenti. Viene effettuata tale scelta implementativa con motivazioni analoghe all'adozione fra due signoli nodi: si evita che le catene di nodi esaminino porzioni di spazio giá sondate dall'altra catena e si estende in profonditá la ricerca. Va sottolineato che, proseguendo la ricerca in questa maniera, il piano non viene esaminato in maniera perfettamente ordinata: potrebbero ancora esserci zone di piano inesplorate raggiungibili dalle due catene senza bisogno di dover fondersi in una struttutura piú sviluppata, ma semplicemente proseguendo nella rotazione; tuttavia questa metodologia risulta essere piú dinamica, riducendo la ridondanza di passaggi per le stesse porzioni di piano con catene della stessa lunghezza, che non portano quindi alcuna informazione aggiuntiva alla ricerca.

Il processo di adozione successivo nel caso di catene di robot risulta inevitabilmente piú complicato rispetto al caso di singoli nodi. In figura (Fig. (13)) sono riportate le varie fasi del processo di adozione nel caso di scontro fra una catena con $N_W = 6$ nodi ed una con $N_L = 3$ agenti. Fra le varie scelte implementative possibili per tale procedura si é optato per una procedura abbastanza sofisticata, dove tutti gli $N_W + N_L$ nodi, tranne uno, devono muoversi in maniera sincronizzata per trovare la propria collocazione nella catena post-fusione. Fra le due catene ve n'é sempre una, detta *vincente*, che effettua un numero di movimenti inferiori rispetto all'altra, la *perdente*[6]; per una questione di economia e per ridurre le inevitabili imprecisioni introdotte da ogni movimento, il criterio per scegliere la catena vincente é la profonditá della stessa: si fa effettuare il maggior numero di spostamenti alla catena delle due con meno unitá. In caso di paritá di nodi la discriminante per scegliere quale delle due catene dev'essere la vincente é l'identificativo del capostipite di ogni catena: cosí come nel caso semplice di scontro fra due nodi, quello che fará meno movimenti sará l'albero con identificativo piú basso.

Tutto il processo di adozione viene di fatto gestito nella fase iniziale esclusivamente dai due nodi capostipite delle due catene: la verifica della condizione di scontro, la scelta dell'albero vincente, e di conseguenza anche la posizione di tutti i nodi nella futura catena unione, viene fatta autonomamente dai due nodi primi nodi, che in seguito comunicheranno

[6]Le notazioni N_W ed N_L indicano rispettivamente il numero nodi della catena vincente, *winner*, e della perdente, *loser*

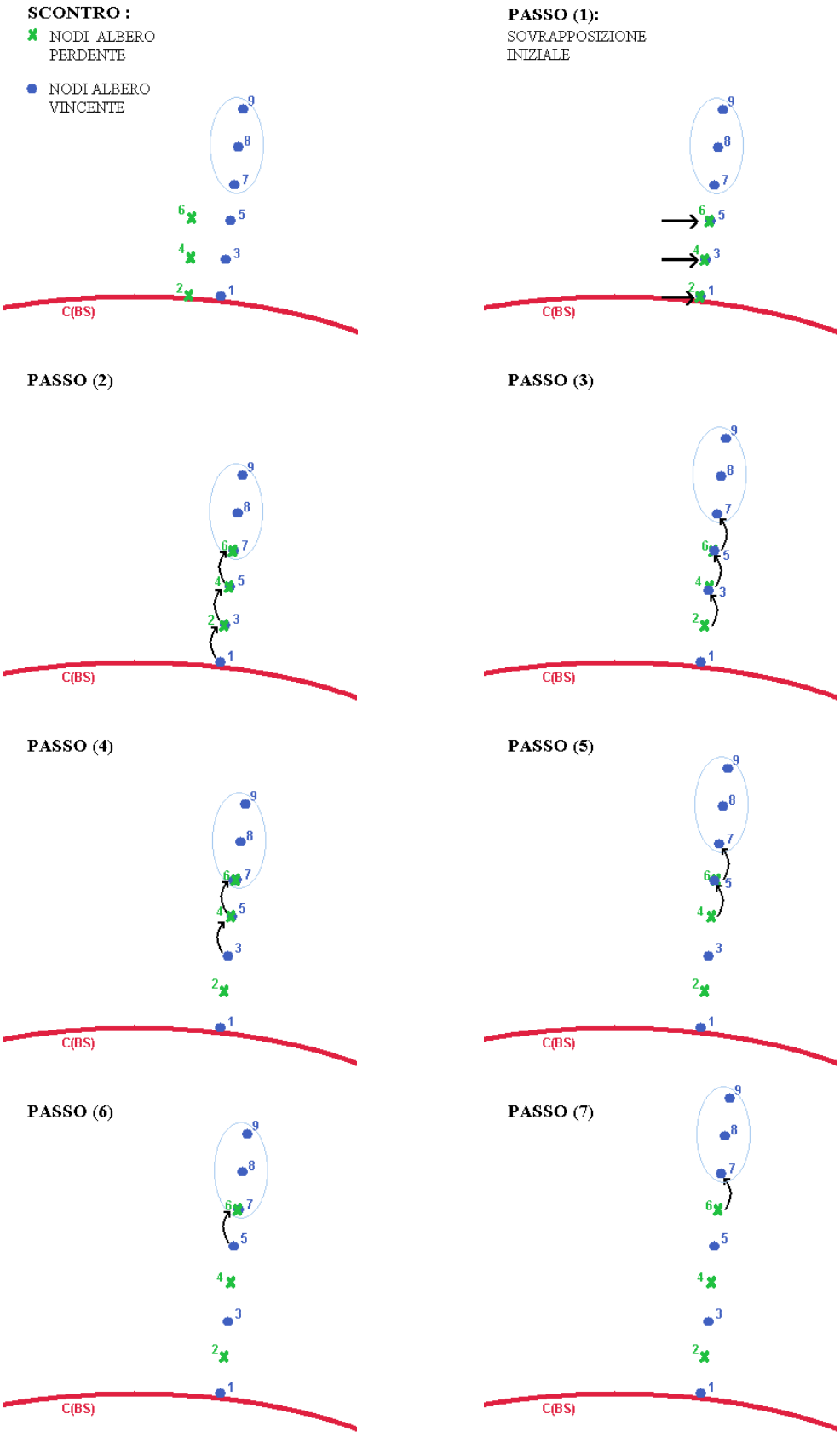

Fig. 13. Processo di fusione fra una catena di 6 nodi ed una di 3

ai figli come dovranno comportarsi nelle seguenti fasi dello scontro.

L'idea base del processo di fusione é la seguente: nel desiderio di minimizzare gli spostamenti complessivi dell'interno network, si penalizza la catena perdente, formata da meno nodi, spostandola interamente sull'asse dell'altra catena. Ogni nodo della *loser chain* alla profonditá[7] n_p viene collocato nella futura struttura immediatamente dopo all'unitá della *winner chain* avente la stessa profonditá n_p: in questo modo nella futura catena saranno presenti, a partire dal capostipite della catena vincente, un nodo della catena vincente sempre seguito da uno di quella perdente, fintantoché le unitá dell'albero di taglia inferiore non vengono esaurite. E' evidente come questa procedura richieda che tutte le unitá si debbano riposizionare, tuttavia complessivamente é una delle soluzioni che piú minimizza la distanza totale da far percorrere all'intero network per effettuare e che richiede il minor tempo d'esecuzione.

Vediamo piú in dettaglio com la procedura avviene. Una volta verificato lo scontro, i nodi partecipanti alla fusione vengono distinti in quattro categorie:

- il capostipite $V(w)$ della catena vincente: verificato lo scontro ed effettuate le comunicazioni necessarie con la propria discendenza, resta in una posizione inattiva fintantoché non viene completato il processo di merge delle due catene (di fatto é l'unico nodo delle due catene a mantenere la propria posizione durante tutta la fase);
- l'insieme dei nodi della catena perdente V_L: hanno come primo compito nella procedura di sovrapposizione quello di doversi sovrapporre agli agenti della catena vincente con la stessa profonditá; il generico nodo $V_L(x)$ e quello a cui dovrá sovrapporsi necessariamente devono trovarsi vicini, essendo alla medesima distanza dalla BS al momento dello scontro: é quindi possibile per $V_L(x)$ stimare il gradiente del futuro padre ed avvicinarvisici fino alla sovrapposizione;
- l'insieme dei nodi della catena perdente V_W dal secondo nodo per profonditá fino all'$(N_W - N_L + 1)$-simo: tale insieme di nodi resta inizialmente in uno stato *idle*, fino a che l'insieme V_L non ha completato la fase di sovrapposizione. Indicheremo fra questi l'$(N_W - N_L + 1)$-simo nodo con $V_W(RIF)$, dato che per la sua procedura di movimento sono necessarie alcune differenze rispetto agli altri V_W;
- il restante insieme dei nodi della catena perdente V_P, dall'$(N_W - N_L + 2)$-simo nodo per profonditá fino dall'ultimo: tali nodi mantengono la posizione relativa nell'albero rispetto a $V_W(RIF)$ anche dopo la fusione, non essendovi abbastanza nodi V_L per sovrapporsi fra loro. Sfruttando tale osservazione il moto di questi nodi puó essere implementato in maniera semplificata rispetto a quello di tutti gli altri agenti: é sufficente per queste unitá copiare il movimento relativo di $V_W(RIF)$ per mantenere la distanza relativa da esso. Ad ogni passo iterativo $V_W(RIF)$ comunicherá lo spostamento relativo effettuato in un passo: alla successiva iterazione il figlio riceverá tale input e copierá tale movimento e trasmetterá al seguente nodo la stessa informazione, affinché anch'esso cambi la propria locazione della medesima quantitá e cosí via. In questa maniera la distanza relativa fra i nodi viene mantenuta; si segnala come tale procedura richieda un certo tempo, affinché il segnale venga trasmesso da $V_W(RIF)$ sino all'ultimo nodo: quando $V_W(RIF)$ avrá concluso il proprio riposizionamento saranno necessari $N_L - 1$ passi affinché tutti i nodi completino il movimento. Per semplicitá d'ora in poi non considereremo piú nella procedura di fusione i movimenti eseguiti da questi agenti, considerando implicito che ad ogni passo questi stiano di fatto copiando il movimento fatto dal proprio padre. Anche se evidente, si sottolinea il fatto che tale insieme puó essere vuoto nel caso in cui

$$N_L \geq N_W - 1,$$

mentre gli altri tre insiemi sopra introdotti sono presenti in ogni procedura di fusione con almeno tre nodi in totale.

Una volta sovrapposti i nodi V_L ai nodi dell'albero vincente rispettivi per profonditá, tutti le unitá robotiche si trovano nel giusto asse dove formare la catena futura, ma la profonditá é inferiore a quella definitiva (tranne nel caso di $V(w)$ che si trova giá nella posizione finale). E' necessaria una procedura per far sí che i nodi avanzino per raggiungere la propria posizione mantenendo lo stesso asse; dato che l'asse é di fatto costituito dai robot stessi non é possibile spostare contemporaneamente tutti i nodi senza perdere il riferimento sulla linea da seguire: é necessario quindi far muovere solo alcuni degli agenti, mantendone fermi altri; a tal proposito la seconda fase della procedura di fusione viene detta fase *a turni*. Una volta sovrapposto, ogni nodo allega al segnale inviato un messaggio d'allerta che viene correttamente interpretato come segnale di inizio della fase successiva d'adozione: la seconda fase comincia per i nodi V_W una volta ricevuto tale allarme e per i nodi V_L al passo successivo all'invio (si suppone per semplicitá che ciascun nodo si sovrapponga al medesimo istante). Ciascun nodo inizializza un contatore CNT al valore 1; il valore di CNT é fondamentale, perché determina quale gruppo di nodi si deve muovere e quale deve restare fermo: con valori dispari di CNT sono i i nodi V_L a muoversi, mentre con valori pari tocca agli agenti V_W. In ambo i casi, le unitá che hanno il diritto di muoversi vanno a sovrapporsi a quelli che saranno i propri figli nella futura catena, passando ovviamente per la necessaria stima del gradiente; il generico nodo $V(x)$ a questo punto dell'algoritmo di trova a distanza $n_x * r_s$ da BS, mentre il figlio si trova a distanza $(n_x + 1) * r_s$: sovrapponendo padre a figlio si aumenta di r_s la distanza dalla BS, avvicinandosi alla posizione desiderata. Numerando i turni di questa fase in ordine crescente partendo da 2, per ogni nodo appartenente a V_L e V_W che si collocherá alla profonditá n_{final} é sufficente aspettare n_{final} turni raggiungere la circonferenza $C_{n_{final}}(BS)$: dopodiché sará sufficiente per tale agente restare idle ed attendere che tutti gli altri agenti completino il posizionamento.

Considerazioni leggermente diverse valgono per l'ultimo nodo della catena $V(last)$, agenti V_P esclusi: nel caso in cui $N_L = N_W$ tale agente appartiene all'insieme V_L, vicerversa

[7]Con profonditá s'intende il numerodi agenti prcedenti in una catena, BS inclusa: il capostipite di una catena avrá profonditá pari ad 1.

esso coincide con $V_W(RIF)$. Per $V(last)$ non é possibile effettuare il movimento di sovrapposizione con il figlio, dato che non ci sono figli, oppure nel caso $V(last) = V_W(RIF)$ il figlio é in movimento, pertanto non é possibile effettuare una stima corretta del gradiente. Per ovviare a questo problema, $V(last)$ stima il gradiente del nodo precedente al proprio padre (ci riferiamo, estendendo la terminologia utilizzata, al nonno), visibile in questa fase dell'algoritmo, e si allontana da esso. Una volta che $V(last)$ ha raggiunto la collocazione finale la procedura d'adozione finisce: $V(last)$ invia nel proprio segnale un allarme al padre per indicare la fine del posizionamento. Prima di riprendere il movimento circolare ogni nodo dovrá attendere che il nodo $V(f)$, l'ultimo nodo a ricevere l'allarme, sia anch'esso informato, in modo tale far cominciare ad ogni anello della catena la rotazione nello stesso istante; ogni nodo deve calcolare un determinato numero di turni d'attesa pari alla propria profonditá nella catena

$$n(x)_{WAIT} = V(x).num_{FATHER},$$

quantitá di turni in cui il nodo resta idle prima di ricominciare il movimento circolare. Nel caso $V(last) = V_W(RIF)$ va tenuto conto anche della presenza degli eventuali nodi V_P che devono tutti essere informati della fine della fine della procedura di fusione: l'attesa ora quindi dipende da[8]

$$n(RIF)_{WAIT} = max\{\Delta_{V_W(RIF)}, \Delta_{V_P(last)} - \Delta_{V_W(RIF)}\},$$

il massimo cioé fra la distanza fra $V_W(RIF)$ e il capostipite $V(f)$ e fra $V_W(RIF)$ e l'ultimo nodo della catena $V_P(last)$.

Una volta fatta la fusione la catena prosegue nella rotazione attorno alla BS con il medesimo verso dell'albero vincente prima del merge; tale scelta non é del tutto arbitraria come nel caso di adozione fra due nodi singoli, ma é vantaggiosa rispetto alla scelta opposta: cosí facendo si va ad esaminare la porzione di piano che si interseca con lo spazio in precedenza analizzato dall'albero perdente, se viceversa la catena fosse andata nella direzione contraria l'intersezione sarebbe avvenuta con lo spazio esaminato invece dalla catena vincente. Dato che l'albero perdente aveva una profonditá di certo non superiore all'altro la porzione di piano da esso esaminato sará piú piccola: la possibilitá quindi di trovare il target é di certo non inferiore cominciando l'analisi in questo verso.

J. Sintesi e critiche all'algoritmo CF

In figura (Fig. (14)) sono sintetizzate tutte le fasi dell'articolato algoritmo presentato. Nonostante la complessitá realizzativa e le supposizioni semplificative introdotte, tale algoritmo risulta in fase simulativa essere inadatto alle ipotesi di partenza. Si sono verificate infatti diverse situazioni nelle quali veniva persa la comunicazione a causa di errati movimenti dei nodi. Le procedure che hanno dato luogo a questo genere di situazioni sono state

- il moto circolare: in alcuni casi si é verificato che, in una catena di due nodi, padre e figlio effettuassero correzzioni divergenti, perdendo per un attimo la connessione fra loro; anche con un controllore piú sofisticato di un semplice P le cose non migliorano: é necessario per un corretto funzionamento che la zona di correzzione venisse limitata, oppure il periodo di aggiornamento del segnale ridotto. Questo ovviamente comporta un pesante rallentamento del sistema in generale facendo ripiegare su altre soluzioni implementative piú performanti.
- l'allineamento nella catena: il nodo figlio, a causa del calcolo del gradiente, che ribadiamo essere impreciso in maniera intrinseca, non si posiziona mai in maniera corretta sulla retta; mentre in alcuni casi tale errore viene tollerato, in altri la correzzione non riesce in partenza a rimediare all'imprecisione iniziale e giá alla prima iterazione di movimento

Nella seguente sezione verranno illustrate alcune possibili varianti alle problematiche sopra descritte.

L'algoritmo CF tuttavia, nel caso di problematiche di ricerca dove non é possibile prescindere dalla specifica di mantenere la connettivitá ad ogni istante oppure con hardware piú complesso a disposizione, é un ottimo approccio, soprattutto quando si attribuisce un'importanza maggiore al ritrovamento di target vicini, mentre si tollera un certo ritardo per identificare sorgenti lontane. Inoltre con hardware computazionalmente competitivo é piú probabile che non ci siano nodi in esubero e che si desideri coprire la massima distanza di con il minor numero di agenti possibili: tale specifica é ampiamente soddisfatta dall'algoritmo CF che sfrutta al massimo le capacitá di copertura del network robotico.

Riassumendo sinteticamente pregi e difetti dell'algoritmo CF:

- Vantaggi: esplorazione ordinata dello spazio, sfruttamento delle complete possibilitá del network robotico;
- Svantaggi: complessitá realizzativa e di programmazione, necessitá di ipotesi di campo isotropo.

K. Implementazione alternativa

Viste le osservazioni precedenti si é introdotta una seconda versione dell'algoritmo CF ipotizzando di avere a disposizione unitá robotiche in possesso di capacitá di calcolo superiori: consideriamo i robot in grado di approssimare funzioni complesse come quelle trigonometriche. Tale soluzione é stata solo introdotta, dato che in corso d'opera si é preferito dedicarsi alle soluzioni SF, performanti anche con poche ipotesi di partenza, quindi piú adatte ad essere implementate in pratica.

Sarebbe possibile implementare un metodo per raggiungere $C_{r_s}(BS)$ (ed in generale tutte le circonferenze a minima intensitá di segnale) perfettamente perpendicolari sfruttando la possibilitá di desumere con esattezza $dist_{BS}$ dall'intensitá del segnale ricevuto. Una volta scelta la direzione verso la quale allontanarsi il nodo salva in memoria la distanza $dist_{old}$ dedotta dall'intensitá di S_{BS}. Se la direzione di movimento fosse esattamente coincidente con quella di gradiente decrescente allora al passo successivo si dovrebbe rilevare che

$$dist_{new} = dist_{old} + \delta;$$

[8]Dove si sono indicate con $\Delta_{V(x)}$ le profonditá nella catena del nodo $V(x)$

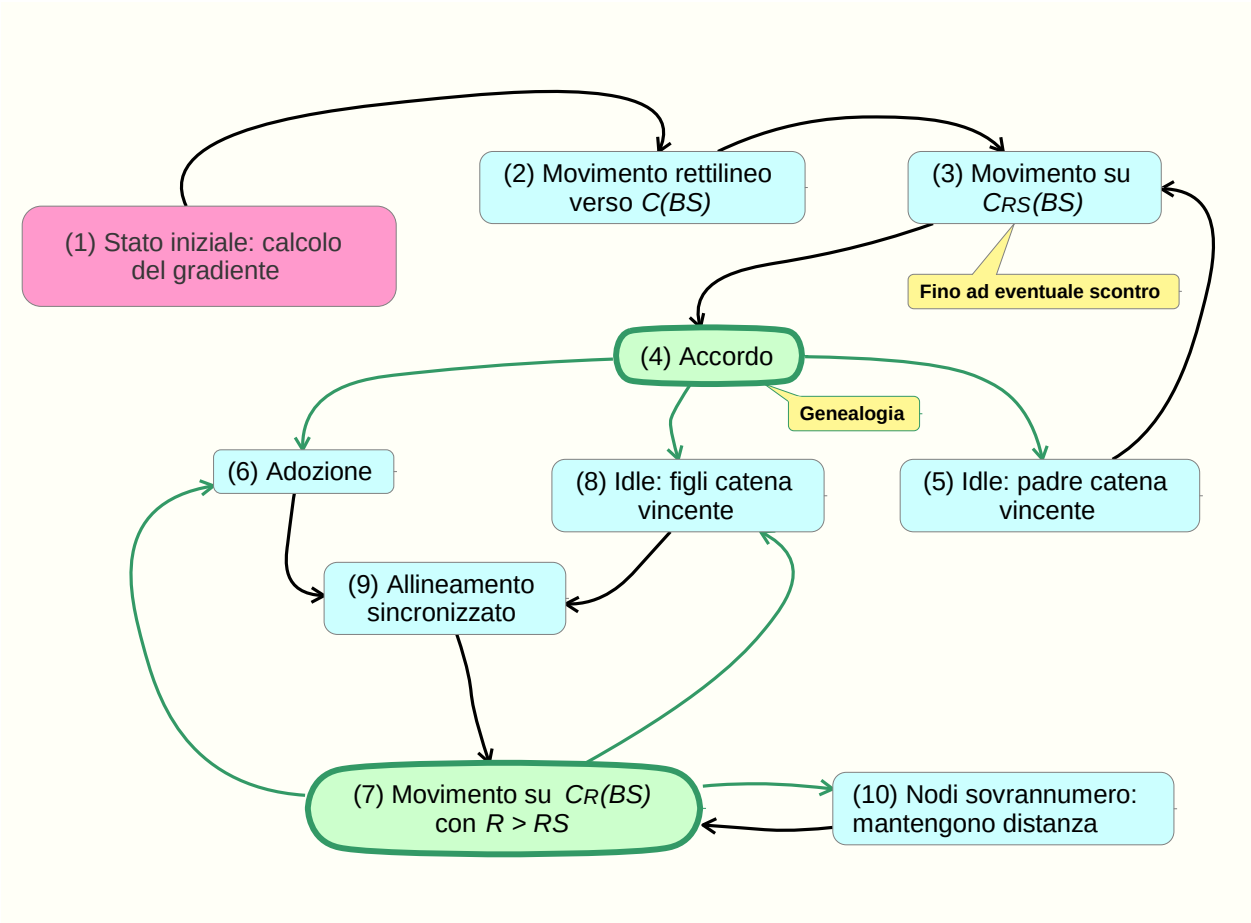

Fig. 14. Mappa degli stati dell'algoritmo CF

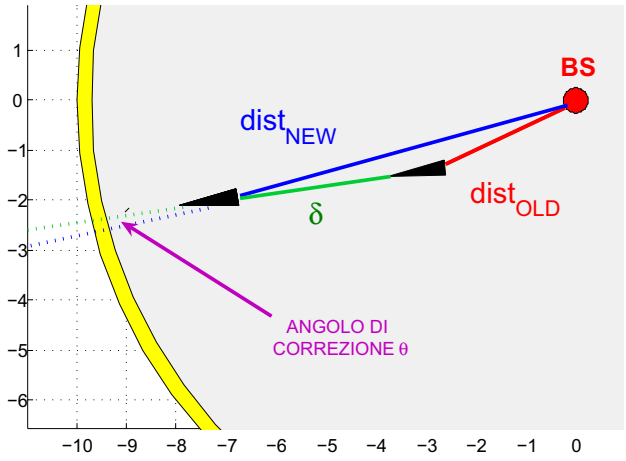

Fig. 15. Processo calcolo angolo correttivo θ

ovviamente, dato che il calcolo del gradiente fornisce solo una stima, l'equazione precedente non sara' mai verificata e ci sará sempre un'errore

$$e_{dist} = dist_{new} - dist_{old} + \delta.$$

Usando la ben nota formula dei coseni e' possibile dedurre l'angolo θ (vedi (Fig. (15))) necessario per correggere l'angolazione di $V(k)$ e farlo arrivare perpendicolare alla $C_{r_s}(BS)$: dal triangolo delimitato da $dist_{new}$, $dist_{old}$ e δ é infatti possibile dedurre θ, dati che tutti i lati rappresentano distanze note per l'unitá; si ottiene

$$\theta = \arccos\left(\frac{dist_{new}^2 + \delta^2 - dist_{old}^2}{2 * dist_{new}\delta}\right).$$

Resta da discriminare il verso in cui aggiungere la correzzione d'angolo: questo puó essere dedotto osservando che la retta perpendicolare a $C_{r_s}(BS)$ che ha origine in BS é sempre passante fra due delle posizioni assunte da $V(k)$ nel calcolo del gradiente; la piú vicina delle due viene scelta dalla routine di calcolo del gradiente ed é la direzione presa dal nodo nel movimento circolare. Mantenendo in memoria il secondo valore meno intenso registrato nella procedura di stima del gradiente di $S(BS)$ é possibile capire qual'é la parte corretta dove aggiungere l'angolo: se la seconda posizione si trova a sinistra di quella scelta per il movimento significa che ci siamo spostati troppo a destra, pertanto sará opportuno aggiungere θ in senso antiorario, viceversa in senso orario.

Il vantaggio di raggiungere con perfezione un punto compensa le imperfezioni introdotte dal calcolo del gradiente, procedura che si é rivelata, come argomentato in precedenza, la causa scatenante di tante problematiche; arrivando in maniera perfettamente perpendicolare su $C_{r_s}(BS)$ e ruotando di $180°$ il nodo si trova perfettamente tangente alla circonferenza centrata nell'origine. E' sufficente quindi impostare un angolo

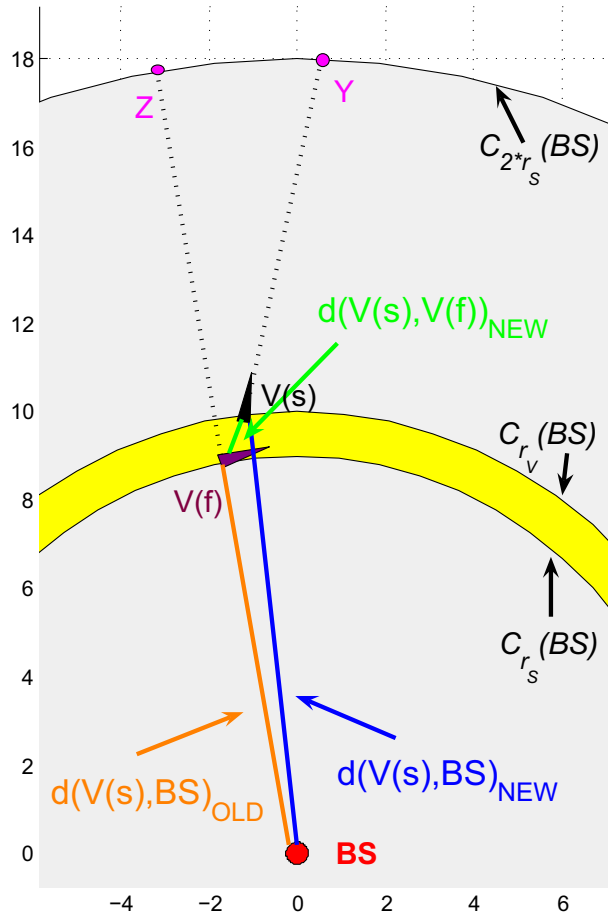

Fig. 16. Schema del processo di triangolazione

di sterzo costante per effettuare un corretto inseguimento sopra $C_{r_s}(BS)$, senza bisogno di alcuna correzione.

Per ovviare al secondo problema riscontrato dall'algoritmo CF, quello dell'allineamento non perfetto, si propone una procedura che, sfruttando la visibilitá della BS e la capacitá di effettuare calcoli trigonometrici consente il corretto posizionamento di un nodo figlio $V(s)$ sul punto Z. Tale processo viene detto *triangolazione*. Nel caso di adozione di due singoli nodi, é possibile sfruttare la contemporanea visibilitá della BS e del nodo $V(f)$ per dedurre l'esatta posizione di $V(s)$, senza bisogno di introdurre alcuna approssimazione. In generale, tale procedura é possibile anche nel caso di fusione di due catene: la posizione di un singolo agente che si sta allineando alla retta puó essere con precisione dedotta dalla distanza dal padre e dal nonno. Come rappresentato in (Fig. (16)), $V(s)$ si sposta su $C_{r_v}(BS)$ impegnando un certo numero di iterazioni: é stato scelto di esaminare il sistema in questo specifico caso, a cui é sempre possibile ricondursi regolando il moto rettilineo, anche se era equivalente considerare $V(s)$ arrivata in una qualunque circonferenza centrata in BS (l'unica cosa di rilievo é l'importanza di non perdere la connettivitá); in analogia con quanto fatto nel processo di correzione sopra esposto, del triangolo di lati noti

$$d(V(s), V(f))_{NEW}$$

$$d(V(s), BS)_{OLD} = r_s$$

$$d(V(s), BS)_{NEW} = r_v$$

dove con i pedici NEW e OLD si é inteso rispettivamente al momento dell'allontanamento da $V(f)$ e dopo, é deducibile l'angolo di rotazione per posizionare $V(f)$ in maniera perpendicolare a $C_{r_v}(BS)$. Una volta applicata la correzzione nel verso adatto, sempre con la procedura elencata che sfrutta la seconda posizione a minima intensitá di segnale, é possibile ruotare $V(s)$ nello stesso senso di altri $90°$: cosí facendo l'agente in esame si trova perfettamente tangente a $C_{r_v}(BS)$. Nota la posizione $(V(s).x, V(s).y)_{NEW}$ é possibile dedurre l'arco di circonferenza necessario per riportare $V(s)$ sulla retta da BS ad X (si tralasciano per non appesantire la lettura i calcoli numerici di tale procedura): una volta posizionato sulla retta, $V(s)$ ruota, in verso contrario a quanto fatto in precedenza, ritrovandosi perfettamente perpendicolare a $C_{2*r_s}(BS)$ nel punto Z.

Applicando qualche nozione di geometria é pertanto possibile risolvere in maniera precisa e corretta le procedure di movimento circolare e di allineamento in catena; si ribadisce tuttavia che questo approccio richiede capacitá di calcolo non previste nel modello computazionale supposto inizialmente: tale aspetto hann scoraggiato a proseguire l'analisi in questa direzione, stilisticamente elegante, ma difficilmente implementabile una volta passati alla fase pratica. Per questo motivo si é preferito concentrarsi sugli algoritmi SF, che non solo si sono dimostrati efficenti con le ipotesi di partenza, ma hanno consentito il rilassamento di alcune di esse.

IV. Approccio SF: Search-First

Nella sezione precedente sono state illustrate le difficoltá nel coordinare il movimento di piu' agenti mobili senza l'utilizzo di Global Position Systems. In questo capitolo verrá presentato un approccio del tutto diverso: la fase di ricerca, anziché avvenire mantenendo la connettivitá fra i nodi, verrá effettuata in maniera indipendente dagli agenti che, solo una volta trovata la sorgente di segnale, si cureranno di instaurare il ponte di comunicazione. Invertendo in questo modo le due fasi principali dell'algoritmo (ricerca da un lato e mantenimento della connettivitá fra i nodi dall'altro) vedremo come si semplifichi notevolmente la complessitá del problema. Ovviamente questo approccio é percorribile esclusivamente se le specifiche di un determinato problema non richiedano la connettivitá fra i nodi ad ogni singolo istante del processo.

Altro grosso vantaggio di questo approccio rispetto a quello Connectvity-First é dato dalla possibilitá di rinunciare all'ipotesi molto forte e, considerando gli strumenti reali, praticamente impossibile da realizzare, di avere canali di comunicazione dove l'intensitá del segnale sia la stessa per ogni punto del piano equidistante dalla sorgente.

Di seguito verranno illustrati diversi algoritmi con ordine crescente di complessitá e di efficienza, illustrando i vantaggi di ogni soluzione e sottolineando gli aspetti migliorabili: le complicazioni concettuali introdotte, passando da un algoritmo ad una versione piu' elaborata, sono state giustificate da simulazioni MATLAB che ne hanno effettivamente dimostrato i vantaggi.

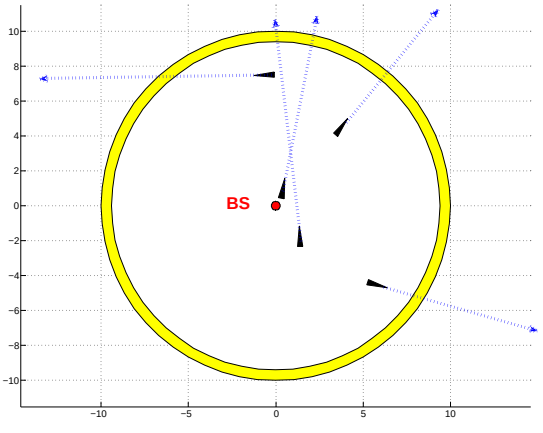

Fig. 17. Condizioni iniziali

A. Algoritmo SF1 - ripetizione del segnale

Questa prima versione dell'algoritmo SF si basa su una semplice idea: se un segnale proviene da diverse sorgenti é piú facile da localizzare rispetto al caso in cui provenga da un'unica sorgente. Partendo da questa banale osservazione, si é sviluppato un procedimento di ricerca dove i nodi, esplorando lo spazio in maniera casuale, una volta trovata la SS si fermino sul posto e non facciano altro che ripetere il segnale, diventando anch'essi a questo punto sorgenti ed obbiettivo per gli altri agenti. Cosí facendo, di volta in volta che un nodo rileva la presenza della Sorgente di Segnale o di un altro nodo che fa da ripetitore, il network si trova ad affrontare un problema di ricerca di decrescente difficoltá .

Illustriamo nel dettaglio le varie fasi dell'algoritmo.

1) Condizioni iniziali e prima fase di ricerca: cosí come nell'algoritmo CF presentato nella sezione precedente, i nodi vengono inizialmente posizionati casualmente all'interno dello spazio di visibilitá della Base Station, con angolazione, rispetto ad un sistema di riferimento non inerziale, anch'essa del tutto arbitraria. I robot cominciano ad esplorare lo spazio circostante muovendosi lungo la retta imposta dalle condizioni iniziali (Fig. (17)): si é supposto che gli agenti siano provvisti di un odometro, cioé siano capaci di calcolare la distanza percorsa. I nodi proseguiranno la ricerca sulla retta sintanto che non raggiungeranno un valore *limit* posto in relazione al raggio di visibilitá r_v ed al numero complessivo N dei robot stessi. Il valore limite di fatto esprime la massima distanza, a partire da $C_{r_v}(BS)$, entro la quale una sorgente incognita verrá individuata; questo limite risulterá essere inferiore ad Nr_s, che era il limite ottenuto con l'approccio CF, che prevedeva la possibilitá estrema di costituire una catena di nodi perfettamente allineata: sará chiaro in seguito come i ponti comunicativi ottenuti con metodo SF non risulteranno essere catene allineate, ma assumeranno le forme piú svariate, il che riduce il potenziale risolutivo del network in termini di copertura dello spazio circostante. D'altro canto il limitarsi a ricercare sorgenti su porzioni di piano piú ristrette di fatto consente di velocizzare la ricerca, dato che i nodi non devono allontanarsi troppo dalla Base Station. Nelle simulazioni, ad esempio, é stato scelto $limit = (2/5)Nr_s$, valori che dimostrano di consentire un buon trade-off fra velocitá e spazio di esplorazione, nonché un'elevata probabilitá di instituire un ponte comunicativo fra BS e SS.

La ricerca, ovviamente, puó avere o meno successo: durante l'esplorazione gli agenti possono o meno trovare la SS; in caso negativo i robot, una volta coperta la distanza $limit$ si girano di 180^o su sé stessi e percorrono a ritroso la retta d'esplorazione. In caso favorevole invece, giá dopo il primo allontanamento (vedremo come il processo, di fatto, si sviluppi fra allontanamenti e fasi di ritorno alla BS) uno o piú agenti possono aver identificato la presenza della SS; in questo caso, come giá accennato in precedenza, i nodi che hanno trovato il target si fermano e diventano dei veri e propri ripetitori del segnale. A questo punto ovviamente la ricerca diventerá piú agevole per gli altri nodi, dato l'aumento del numero di target utili per trovare il segnale incognito.

2) Fase di ritorno ed allontanamenti successivi: gli agenti che non hanno trovato il target ripercorrono a ritroso il cammino percorso fino a ritornare nel raggio visivo della BS. Una volta tornati in prossimitá della base i robot cominciano ad analizzare e mantenere in memoria passo dopo passo l'intensitá del segnale del 'nodo di partenza': ad ogni iterazione gli agenti verificano se il segnale é aumentato di intensitá, cioé se effettivamente ci si sta avvicinando alla BS. Quando l'intensitá del segnale comincia a calare, il robot comincia una nuova fase d'esplorazione: l'agente routa di $(180 - \alpha)^\circ$, dove α é regolabile dall'utente, e comincia con una seconda fase d'allontanamento seguendo la retta impostata dal nuovo angolo; in questo modo il nodo percorre una retta diversa da quella percorsa nella prima fase d'allontanamento spostando la ricerca su una nuova porzione del piano. Giá a questo punto é intuibile come il processo venga iterato nelle fasi successive di movimento nel caso in cui non venga mai trovato alcun target: il nodo quindi esplora rette che hanno come centro il punto dove l'intensitá di segnale proveniente dalla BS comincia a diminuire. Va sottolineato come sia possibile che anche durante una fase di ritorno, e non solo durante le fasi di allontanamento, i robot possano trovare un agente che sia in funzione di ripetitore.

3) Ultimazione del ponte radio: nel momento in cui un robot $V(k)$ stabilisce il contatto con un ripetitore e si trova entro il raggio visivo della Base Station, il collegamento tra BS e SS risulta ultimato: al successivo rientro in base di tutti i robot in esplorazione, ai quali la BS comunica un segnale di stop immediato, la procedura $SF1$ termina.

Alla luce di quanto illustrato é chiaro che la realizzazione del collegamento non prevede alcuna gerarchia tra i veicoli via via inseriti; infatti, data l'aleatorietá della disposizione iniziale del network, e la derivante casualitá delle direzioni intraprese da ogni robot, accade spesso che piú veicoli si trovino a poca distanza l'uno dall'altro nel percorrere le rette di esplorazione e, se la distanza che li separa é minore di r_v, ognuno dei robot in questione é in grado di percepire il segnale di almeno uno dei vicini: nel caso in cui uno di essi stabilisca il contatto con il segnale sorgente, tutti gli agenti, ad esso collegati, diventano immediatamente ripetitori.

Sono queste le situazioni, come poc'anzi accennato, in cui

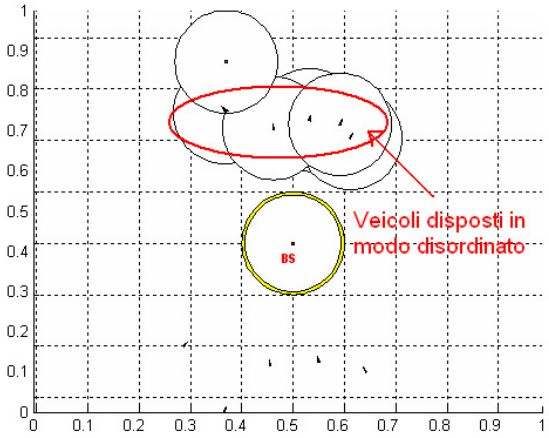

Fig. 18. Distribuzione non ordinata dei veicoli durante una procedura dell'algoritmo $SF1$.

piú robot si dispongono, nei pressi del segnale sorgente o di un qualsiasi ripetitore, in modo disordinato, 'espandendo' la connessione in piú direzioni. Un simile contesto, di cui si riporta un esempio in figura (18), limita notevolmente l'efficienza dell'$SF1$, poiché il numero di robot in eccesso, coinvolti nel collegamento, potrebbero essere utilizzati nell'attivitá di eplorazione e, non meno importante, nella realizzazione di ponti radio piú estesi, aumentando quindi l'insieme delle casistiche risolvibili con l'utilizzo di tale procedura.

Altro aspetto negativo dell'algoritmo $SF1$ é l'aumento del tempo impiegato nella realizzazione del collegamento, al crescere della distanza relativa al posizionamento del segnale sorgente: questo infatti provoca un aumento corrispondente dello spazio che ogni veicolo deve percorrere durante la fase di esplorazione. I dati esposti nella sezione (V) daranno una prova tangibile delle prestazioni appena descritte.

A vantaggio delle qualitá di questa procedura si ha una notevole semplicitá computazionale: i componenti del network hanno infatti un numero molto limitato di istruzioni da eseguire e di condizioni da verificare.

L'$SF1$ puó essere considerato un buon punto di partenza, la filosofia di base da cui sviluppare nuove soluzioni che mirino ad ottimizzare il posizionamento dei veicoli costituenti il collegamento, e che, nello stesso tempo, riducano i tempi di realizzazione del ponte, aumentando, di conseguenza, l'insieme delle casistiche in cui tali procedure giungono al termine in tempi accettabili.

Riassumendo:

- vantaggi:
  - semplicitá realizzativa, nessuna ipotesi sulla geometria del campo di comunicazione;
  - leggerezza computazionale, le istruzioni per ogni robot sono semplici e limitate;
- svantaggi:
  - prestazioni non ottimali, sovrabbondanza di veivoli connessi al ponte;
  - probabilitá di ultimare il ponte radio decrescente all'aumentare della distanza di SS.

B. Algoritmo SF2 - ricerche di taglia decrescente

La procedura implementata nell'algoritmo $SF2$ riprende l'idea di base illustrata nel precedente $SF1$: in un primo momento, la ricerca del segnale sorgente da parte dei robot mediante esplorazione lungo percorsi in direzioni random, e, successivamente, l'estensione del ponte radio tramite 'ripetitori'. La differenza sostanziale, rispetto al primo algoritmo, sta nella modalitá di connessione degli agenti al segnale sorgente (o, allo stesso modo, ad un qualsiasi ripetitore), la quale prevede un assegnamento di un preciso obbiettivo, che verrá indicato come *target*, agli agenti in esplorazione, in modo da imporre un'ordinamento gerarchico nella costruzione del collegamento tra SS e BS.

Si analizzano in dettaglio le varie fasi dell'algoritmo, evitando descrizioni approfondite riguardanti procedure giá illustrate in precedenza.

1) Condizioni iniziali e ricerca della sorgente: inizialmente la Base Station imposta come target il segnale sorgente aggiornando il campo $BS.signal.target = N + 1$; la BS inoltre aggiorna in memoria il limite superiore della distanza che ogni robot in perlustrazione dovrá percorrere in ($BS.signal.limit = \frac{2}{5}Nr_v$): entrambi i dati vengono 'letti' da ogni generico robot $V(k)$ che si trovi all'interno di $C_{r_s}(BS)$, il quale, a sua volta, imposta rispettivamente i campi $V(k).target = N + 1$ e $V(k).limit = \frac{2}{5}Nr_v$.

Ad inizio procedura, pertanto, tutti i veicoli hanno il compito di rintracciare il SS di collocazione ignota (le componenti del network, in principio, sono infatti posizionate all'interno della $C_{rs}(BS)$ e quindi in grado di percepire le informazioni messe a disposizione dalla BS) ed il network intraprende la fase di esplorazione in modalitá adottata nell'algoritmo $SF1$: i robot percorrono una retta (sino a raggiungere la distanza $V(k).limit$ impostata), in direzione stabilita dalle condizioni iniziali, esplorando una porzione di piano e, se nel cammino non trovano il segnale sorgente, reiterano il movimento in direzione opposta e si muovono successivamente su una retta con diversa angolatura.

2) Procedura di connessione al ponte radio: quando un qualsiasi robot $V(k)$ rintraccia la sorgente, inverte immediatamente il senso di marcia ed avanza sino a ristabilire il contatto con la Base Station: ad essa comunica l'obbiettivo raggiunto ('Sorgente di Segnale trovata') e la distanza calcolata, e tenuta in memoria, durante il cammino eseguito nella fase di rientro. Se $V(k)$ é il primo veicolo ad aver 'conquistato' tale obbiettivo, esso ritorna nella posizione in cui ha stabilito il contatto con la sorgente e diventa il primo componente del ponte radio.

Nel frattempo la BS aggiorna i campi summenzionati con i dati ricevuti dal robot: l'identificativo di $V(k)$ diventerá il nuovo target, mentre la distanza che intercorre tra la BS e il SS ($d(BS, SS)$) sostituirá il valore $\frac{2}{5}Nr_v$ precedentemente salvato in $BS.signal.limit$: si noti che $d(BS, SS)$ puó essere minore, o al massimo equivalente, a tale valore, poiché la sorgente puó essere rintracciata ad una distanza dalla BS pari

o minore al limite superiore $\frac{2}{5}Nr_v$ inizialmente specificato. I nuovi dati, da questo momento in poi, sono disponibili per ogni agente $V(p)$, in perlustrazione, che entri in $C_{r_s}(BS)$: $V(p)$ aggiorna il target da identificare e la distanza massima da percorrere nella nuova ricerca assegnata.

Da qui in poi, ogni robot che raggiunge per primo l'obbiettivo prefissato, eseguirá la stessa procedura appena descritta, posizionandosi successivamente 'alle spalle' dell'ultimo ripetitore collegato al ponte, e provocando perció un'espansione della catena di comunicazione in direzione della base.

3) Ultimazione dell'algoritmo: La procedura termina quando l'ultimo ripetitore é in grado di 'sentire' il segnale S_{BS}, e tutti i veicoli in esplorazione rientrano in base fermandosi nella posizione di partenza.

Utilizzando la procedura descritta, ogni robot in perlustrazione puó stabilire il contatto, escludendo la sorgente, solamente con l'ultimo veicolo divenuto ripetitore: durante l'evoluzione del ponte, che di fatto risulta regolarizzata da una precisa gerarchia, si assegna quindi ad ogni robot una posizione piú consona alla corretta espansione della catena comunicativa, evitando, come avveniva nel caso precedente ($SF1$), l'utilizzo di un numero eccessivo di veicoli nel collegamento.

Per di piú, la limitazione del campo di eplorazione ad ogni aggiornamento successivo del target, dovuta alla ricerca di obbiettivi situati in posizioni via via sempre piú prossime alla base, riduce progressivamente i tempi necessari per il rintracciamento di tali obbiettivi, velocizzando la procedura di realizzazione del ponte radio, rispetto all'implementazione precedente.

D'altro canto ogni singola aggiunta di agenti richiede piú tempo ($t_{connessione}$) rispetto al caso precedente ($t_{connessione} \approx 0$: in $SF1$ veicoli si bloccano appena entrano in contatto con un ripetitore), dato che i veicoli, una volta rintracciato l'obiettivo, devono prima ritornare alla BS per uno scambio di informazioni, e successivamente ripercorrere la stessa distanza per connettersi al ponte in costruzione. Questo fatto non é rilevante quando si opera con un segnale sorgente non troppo lontano, ma comporta un aumento dei tempi di risoluzione del problema al crescere della distanza della sorgente: si delinea infatti che durante $t_{connessione}$ il nuovo target non é attivo (non é ancora collegato al ponte) e di conseguenza, nel medesimo intervallo, i veicoli in esplorazione cercano un obbiettivo invisibile.

Ció nonostante, si puó considerare l'evoluzione $SF2$ migliore rispetto alla versione precedente, poiché il sistema di assegnamento dei target regolarizza la costruzione del ponte, limitando il numero di robot utilizzato nel collegamento e massimizzando in ogni istante il numero di veicoli in eplorazione: di conseguenza all'aumentare della distanza del SS, a paritá di agenti inizialmente disponibili, con l'approccio $SF2$ si riesce ad ultimare il ponte con piú probabilitá rispetto al caso in cui si utilizzi l'$SF1$. Tali affermazioni verranno comprovate nell'analisi dei risultati simulativi (sezione V).

Riassumendo

- Vantaggi:
  - semplicitá realizzativa, nessuna ipotesi sulla geometria del campo di comunicazione;
  - istituzione di un ordine gerarchico tra i veicoli che costituiscono il ponte;
  - maggiore probabilitá di istituire il ponte (al crescere della distanza del SS);
- Svantaggi:
  - aumento dei tempi di realizzazione del ponte al crescere della distanza del SS.

C. Algoritmo SF3 - Pianificazione del ponte radio

Nelle procedure viste sinora la costruzione del ponte, in generale, partiva dal segnale sorgente ed era completata dai veicoli in esplorazione che, in modo completamente casuale, stabilivano un contatto con i ripetitori presenti: tale metodologia, come giá puntualizzato, risulta inefficace, in quanto la probabilitá di ultimare il collegamento in tempi non troppo elevati diminuisce all'aumentare della distanza tra BS e SS.

Si é quindi pensato di implementare una procedura che realizzi il ponte in maniera del tutto deterministica, in modo tale da avere il controllo completo sul posizionamento di ogni singolo componente della catena comunicativa, nell'intento di massimizzare la lunghezza del collegamento utilizzando il minimo numero di robot.

L'algoritmo SF3 propone una risoluzione del problema in questione basata su due principali momenti:

- ricerca del segnale sorgente e nomina di $leader$ per il primo robot che comunichi alla BS l'obiettivo raggiunto;
- riunione alla base del gruppo di ricerca robotico e realizzazione del ponte radio.

Esso si discosta quindi dal metodo di ripetizione del segnale, concentrandosi su una pianificazione del ponte prettamente deterministica. Tale progettazione ha origine all'interno del raggio visivo della BS, e prevede la progressiva aggiunta di veicoli alla $catena$ [9] e la conseguente espansione del ponte verso la sorgente (a differenza degli algoritmi precedenti), durante la quale ogni robot adotta un sistema di controllo sulle distanze relative con gli agenti contigui che garantisca la connettivitá.

Il leader si identifica come il veicolo che per primo ha trovato la sorgente e funge da guida per tutti gli agenti in catena: esso é infatti l'unico a conoscere la giusta direzione per raggiungere la sorgente, dato che gli altri anelli, come verrá dimostrato in seguito, non sono in grado di allinearsi perfettamente tra di loro, e risulta quindi una componente fondamentale per l'istituzione del ponte.

Come si puó intuire dovrá essere ripristinata l'ipotesi di isotropia dei segnali (la numero 4 del paragrafo III), essenziale per il controllo del movimento di piú veicoli basato unicamente sulle informazioni delle distanze relative tra i veicoli stessi.

Si analizza ora nel dettaglio l'evoluzione dell'algoritmo SF3. Non verrá illustrata l'impostazione delle condizioni iniziali e la modalitá di ricerca del segnale sorgente, poiché

[9] Con il termine $catena$ si alluderá al ponte radio e con $anello$ ci si riferirá ad un suo qualsiasi componente robotico.

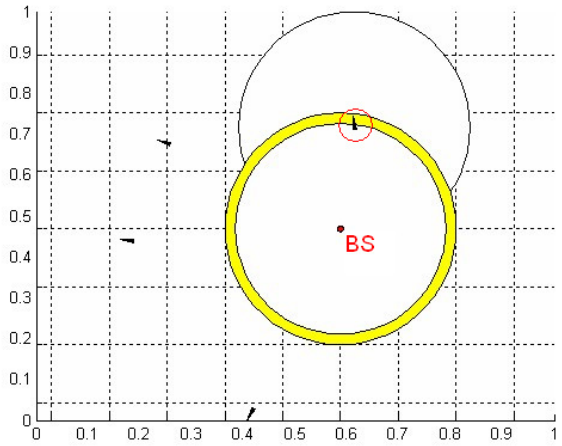

Fig. 19. Leader (evidenziato in rosso) in attesa di veicoli per estendere la catena.

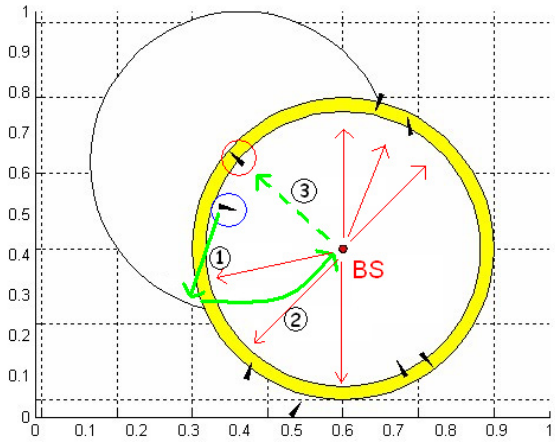

Fig. 20. Processo di aggiunta di un anello: percorso (frecce verdi) del figlio eletto (evidenziato in blu). Le frecce rosse indicano le direzioni di minima variazione dell'intensitá del segnale $S(BS)$.

analoghe a quelle ampiamente descritte negli algoritmi precedenti (SF1, SF2).

1) Elezione del leader e raccoglimento dei robots alla base: si supponga che sia giá inizita l'operazione di ricerca della sorgente. Qualora la Base Station riceva da un qualsiasi robot la conferma di rintracciamento dell'obbiettivo, nomina $leader$ il veicolo (che verrá chiamato $V(l)$), il quale si posiziona in prossimitá della circonferenza $C_{r_s}(BS)$ in direzione del segnale sorgente (Fig. (19)): $V(l)$ d'ora in poi eseguirá solamente movimenti rettilinei, poiché é l'unico veicolo che conosce la direzione verso il SS. Il leader é il primo anello del ponte che realizzerá il collegamento tra base e sorgente, e, in questo caso, rimane in attesa di entrare in contatto con un qualsiasi robot nelle vicinanze per dare inizio alla procedura di 'aggiunta di un anello' (descritta nel prossimo punto). $V(l)$, inoltre, salva in memoria la distanza percorsa dal segnale sorgente alla $C_{r_v}(BS)$ durante la fase di rientro, e la utilizza nel calcolo del numero di agenti n_{anelli} che servono per la realizzazione del ponte lungo tale cammino: il valore, opportunamente scalato, verrá impiegato per comunicare ad ogni anello entrante in catena il numero di veicoli mancanti per il completamento del ponte.

Se $dist$ rappresenta lo spazio percorso dal SS alla $C_{r_v}(BS)$, si ha che il numero di robots necessari é

$$n_{anelli} = \lceil dist/rs \rceil + 2.$$

Tale cifra é stata definita presupponendo che l'aggiunta di un generico anello, comporti l'estensione del ponte di una distanza pari, approssimativamente, ad r_s lungo la direzione della sorgente, ed é stata maggiorata di una unitá per ragioni di sicurezza: infatti, come verrá confermato nei prossimi punti, un ponte completo sará caratterizzato da anelli 'in linea' fra di loro, cioé visualizzabili in un intorno della retta fittizia, congiungente la BS ed il punto in cui il leader ha stabilito il contatto con la sorgente, e distanziati l'uno dall'altro di $r_s - \epsilon$, dove $0 < \epsilon << r_s$. L'imprecisione relativa al distacco tra gli anelli é dovuta alle imperfezioni nelle procedure di posizionamento, che verranno illustrate nel seguito.

Da queste osservazioni si puó dedurre che il limite massimo della distanza, a partire dalla BS, a cui possono spingersi i robot nella fase di esplorazione, coincidente con la distanza entro la quale deve trovarsi la SS perché il problema sia risolvibile, é maggiore rispetto al valore $limit(N)$ definito per i precedenti algoritmi: se la disposizione degli anelli di un generico ponte fosse perfetta (se fossero in linea e distanziati di un valore pari a rs) si avrebbe $limit(N) = (N-1)r_s$, e, di conseguenza, il SS potrebbe essere rintracciato nei casi in cui $C_{r_v}(SS)$ avesse punti in comune con $C_{r_v+limit(N)}(BS)$ (che rappresenta l'area totale esplorata da N agenti). Dato che il posizionamento di ogni anello ha un certo margine d'errore, si definisce $limit(N) = (N-2)r_s$, diminuendo di un valore pari ad r_s la distanza entro la quale ci si prefigge di riuscire nella realizzazione del ponte. Si precisa che tale limite é valido per N non troppo elevato (indicativamente $N < 30$), poiché all'aumentare del numero di robot componenti un dato ponte, aumenterebbe il peso dell'errore di distanziamento fra gli anelli e, di conseguenza, la massima lunghezza del collegamento potenzialmente realizzabile con gli agenti a disposizione.

Nel contempo i robots in esplorazione, compresi eventuali agenti entrati in contatto con il segnale sorgente successivamente al leader, convergono alla base. Il loro compito, una volta rientrati, é di riportarsi nella zona in cui ha origine la catena, restando all'interno di $C_{r_s}(BS)$, e rendersi disponibili per eventuali richieste di aggregazione. Da questo punto in poi l'algoritmo si distacca completamente dalle procedure illustrate in $SF1$ ed $SF2$, concentrandosi nella cooperazione tra veicoli per la realizzazione del ponte radio.

2) Aggiunta di un anello alla catena: durante questa fase si considera, come nel processo di 'adozione fra due singoli nodi' visto nel paragrafo D della sezione III, la nomina di nodi padre e figlio, in modo tale che ogni robot della catena abbia un padre, il nodo da cui é stato eletto, ed un figlio, il nodo che ha successivamente designato: il leader, per esempio, vede come figlio il veicolo scelto, ma non possiede un padre

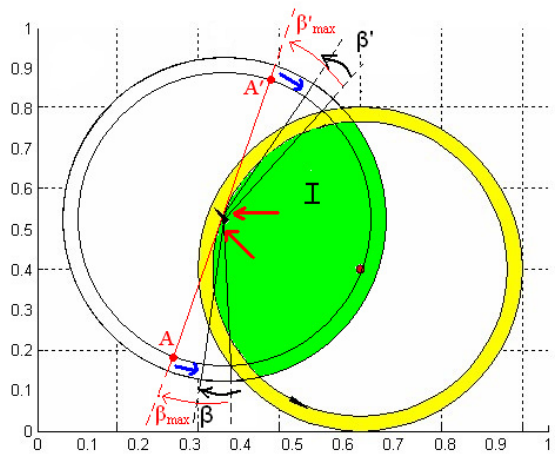

Fig. 21. Processo di aggiunta di un anello: percorso nel caso peggiore.

poiché non é stato eletto da nessuno.

In figura (20), come ausilio per la prossima descrizione, é riportato un esempio (frecce verdi) dei vari movimenti che un robot deve eseguire per completare la procedura di aggiunta alla catena.

Si supponga che la generica catena in questione sia caratterizzata da un solo anello (il leader $V(l)$), e che il medesimo sia in contatto con uno dei veicoli rimanenti $V(k)$: questo verrá scelto da $V(l)$ per essere annesso alla catena (se $V(l)$ fosse in contatto con piú agenti ne sceglierebbe uno in modo casuale, poiché non sussistono particolari motivi nel prediligere la scelta di uno rispetto ad un altro).

Il procedimento di aggiunta ha inizio solo se la distanza tra i due é minore di rs, ipotesi fondamentale per garantire la connettivitá nel corso delle prossime operazioni.

In primo luogo $V(k)$ calcola la direzione di variazione minima del segnale relativo al padre. Successivamente avanza lungo tale direzione fino a quando l'intensitá di $S_{V(l)}$ non si riduce al limite di sicurezza (movimento n^o1 in figura (20)), corrispondente al posizionamento del veicolo in prossimitá di $C_{r_s}(V(l))$. A movimento terminato, in base alla locazione del veicolo, si possono contraddistinguere due diversi casi:

1) $V(k)$ si trova ancora all'interno di $C_{r_s}(BS) \rightarrow$ il veicolo si muove a gradiente costante (movimento circolare lungo $C_{r_s}(V(l))$) rispetto al leader in un senso di percorrenza scelto casualmente, e valuta ad ogni passo come varia la distanza $d(V(k), BS)$: se $d(V(k), BS)$ aumenta significa che si sta allontanando dalla posizione da raggiungere e deve invertire il senso di marcia; se diminuisce il robot puó continuare il moto sino ad un incremento di $d(V(k), BS)$ (movimento n^o2 in figura (20)), condizione valida per concludere il movimento circolare.
2) $V(k)$ non é in comunicazione con la base (si trova per esempio nel punto A in figura (21)) $\rightarrow$ il veicolo effettua il movimento circolare rispetto al leader (in senso di rotazione casuale) lungo un arco di circonferenza sotteso da un angolo $\alpha \approx 50°$: se nel frattempo il robot entra in contatto con la base, si passa al caso sopra citato; altrimenti il veicolo inverte il senso di marcia ed avanza lungo $C_{rs}(V(l))$ sino a ristabilire il contatto con la Base Station.
Il numero di passi (n_{prova}) corrispondente al percorrimento del suddetto arco é stato definito in via sperimentale.

Per delucidare la procedura attuata nel caso 2, si considerino le posizioni $p(grad)$, in cui il figlio acquisito compie il calcolo del gradiente di inizio procedura, e $p(mov_1)$, in cui si trova $V(k)$ a movimento n^o1 terminato. L'area I in figura (21) rappresenta l'insieme di tutte le possibili posizioni in cui puó trovarsi un veicolo appena scelto come figlio e si ottiene, in generale, dall'intersezione delle aree $C_{r_s}(BS)$ e $C_{r_v}(V(l))$; l'angolo β (β' nel caso simmetrico) viene utilizzato per quantificare la distanza di $V(k)$ da $C_{r_v}(BS)$ misurata lungo l'arco a cui é sotteso. Si osserva che l'ubicazione $p(mov_1)$ dipende da $p(grad)$: si puó dunque affermare che β aumenta tanto piú $p(grad)$ é prossima a $C_{r_s}(BS)$ e, contemporaneamente, alla posizione del leader, raggiungendo il valore massimo β_{max} ($\beta_{max} < 50°$ come evidente dalla figura) quando $p(grad)$ é uno dei punti indicati dalle frecce rosse. Di conseguenza $p(mov_1)$ si troverá, nella peggiore delle eventualitá, in un intorno del punto A (A' nel caso simmetrico), dato che il calcolo del gradiente ha sempre un lieve margine di errore.

Dalle osservazioni fatte si puó concludere che se $p(mov_1)$ é fuori dal raggio visivo della BS, il percorso minimo per completare il movimento n^o2 é quello in direzione indicata dalle frecce blu in figura (21), e, se presa tale direzione, il rientro alla BS avviene percorrendo un arco di $C_{r_s}(V(l))$ sotteso da un angolo sicuramente minore di $50°$. Quindi se la posizione di $V(k)$ alla fine del movimento n^o1 rientra nel $caso2$ sopra citato, $V(k)$ 'riconosce' di aver scelto il percorso piú impegnativo, se dopo n_{prova} passi non é ancora in contatto con la BS.

Al termine dello spostamento n^o2, il robot calcola la direzione di massima variazione di $S_{V(l)}$ per l'esecuzione del movimento n^o3.

Il leader, a questo punto, esce dallo stato di attesa e le sue mansioni d'ora in poi saranno: 'copiare' i movimenti in linea retta del figlio, a condizione che la distanza che li separa risulti inferiore a $rs + \delta$, mantenendo sempre la stessa direzione (in caso contrario si rischierebbe di guidare la catena verso un punto morto) e, al ritrovamento della sorgente, inviare un segnale di allarme al figlio, che si propagherá ad ogni anello della catena notificando l'ultimazione del collegamento radio.

A causa delle notevoli imprecisioni nel calcolo del gradiente, $V(k)$ non é in grado di fissare una direzione tale da consentire l'avanzamento perfettamente parallelo tra i due agenti, condizione fondamentale per garantire la connettivitá. In principio si era pensato di risolvere la complicazione correggendo lo spostamento del robot, mediante opportune variazioni dell'angolo di sterzo, in modo da mantenere costante nel tempo la distanza $d(V(k), V(l))$: tale espediente si é peró rivelato irrealizzabile in quanto presuppone capacitá di calcolo superiori alle potenzialitá computazionali degli agenti robotici (per esempio l'utilizzo della funzione $arcos(y)$). É stata quindi adottata una soluzione che prescinde dal calcolo della direzione esatta: $V(k)$ controlla ad ogni passo la sua

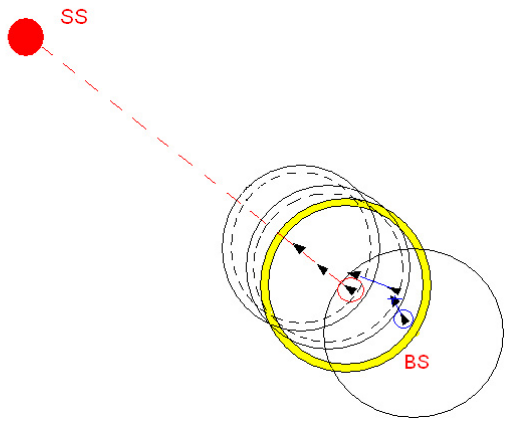

Fig. 22. Avanzamento del leader (in rosso) e fasi di avanzamento e correzione della direzione per il figlio (in blu).

distanza dal padre, e, nel caso questa superasse la soglia di un valore pari ad rs, reimposta la direzione con quella ottenuta in base al calcolo di massima variazione di $S_{V(l)}$. Tale operazione, quando l'errore nel calcolo del gradiente non é eccessivo (in tal caso viene nuovamente eseguito il medesimo calcolo), viene illustrata in un esempio in figura (22).
Al termine della procedura di aggiunta dell'anello alla catena, $V(k)$ si trova al limite della $C_{rs}(BS)$, é in contatto con il robot in grado di raggiungere la sorgente (il leader) e, nel caso in cui abbia ricevuto dal padre ordini di reclutamento ($n_{anelli} > 0$), puó designare un nuovo anello da inserire in catena; questo a sua volta eseguirá la procedura appena descritta e nel compiere il movimento n^o3 provocherá l'avanzamento del padre, passato nel frattempo allo stato di 'Avanzamento condizionato' (descritto nel prossimo punto), e quindi del leader.

3) Avanzamento condizionato della catena: si attiva per tutti gli anelli $V(k)$, tranne il leader, che hanno giá completato la procedura 2.

Siano $V(k-1)$ e $V(k+1)$, rispettivamente, il figlio ed il padre di $V(k)$, e $d(V(k), V(k-1))$, $d(V(k), V(k+1))$ le distanze di $V(k)$, rispettivamente, da $V(k-1)$ e da $V(k+1)$. L'obbiettivo primario, per il generico anello $V(k)$, é il mantenimento della connettivitá con il padre ed il figlio; in secondo luogo deve procededere in modo tale che $d(V(k), V(k-1)) \approx rs$ e $d(V(k), V(k+1)) \approx rs$ allineandosi il piú possibile con $V(k-1)$ e $V(k+1)$: in tali condizioni si massimizza la lunghezza del ponte in base al numero di anelli che lo realizzano.
$V(k)$ copia ogni movimento rettilineo eseguito dal figlio solamente se vale

$$(C1)\ d(V(k), V(k-1)) < rs + \delta;$$

simultaneamente valuta, ad ogni passo, la distanza che lo separa dal padre: se non é soddisfatta la

$$(C2)\ d(V(k), V(k+1)) < rs,$$

si blocca ed esegue il calcolo del gradiente scegliendo la direzione di massima variazione di $S_{V(k+1)}$. Si noti che la condizione $C2$, e le operazioni previste per il suo soddisfacimento, sono le stesse adottate per il movimento n^o3 della procedura 2 [10].

L'avanzamento condizionato della catena si puó quindi riassumere come segue: un generico veicolo effettua il movimento n^o3 della procedura 2, e questo viene 'copiato' dal padre e via via da tutti gli agenti della catena in successione sino al leader; ció determina un'estensione complessiva del ponte approssimativamente pari alla distanza rs. Quando un generico anello $V(k)$ si trova costretto a correggere la propria direzione (condizione $C2$ non soddisfatta), mediante calcolo del gradiente, la parte superiore della catena (dal veicolo $V(k+1)$ in poi) si blocca, non essendoci alcun movimento rettilineo da copiare; in tale situazione si avrebbe quindi l'avanzamento della sola parte inferiore (dal primo anello al veicolo $V(k-1)$), e una coseguente deformazione del ponte. Per evitare simili frangenti, ogni veicolo in procinto di eseguire il calcolo del gradiente trasmette un'allarme al figlio, che si propaga sino all'origine del movimento rettilineo (il primo anello della catena), bloccando tutti i veicoli ad esso precedenti.

É evidente come le continue correzioni eseguite dagli anelli rallentino sensibilmente il processo di estensione della catena; d'altro canto esse risultano indispensabili per il mantenimento in linea dei robot coinvolti.

Le prestazioni dell'$SF3$ risultano superiori a quelle ottenute nei casi precedenti: con tale approccio, infatti, si riescono a creare catene di veicoli comunicanti piú estese, coprendo quindi un insieme di casistiche, in cui la risoluzione del problema é assicurata, molto piú ampio. Inoltre i tempi di esecuzione, come verrá esposto nella prossima sezione, a paritá di robot disponibili, non crescono in maniera spropositata all'aumentare di $d(BS, SS)$ come succede per gli algoritmi $SF1$ ed $SF2$.

Tuttavia, si deve tener presente che le operazioni descritte per la creazione del ponte, richiedono una maggiore complessitá computazionale, e prevedono l'utilizzo di ipotesi (come l'isotropia dei segnali) molto difficili da ricreare in un eventuale ambiente di sperimentazione.

Riassumendo

- Vantaggi:
  - istituzione di un ponte con componenti allineate e possibilitá di scelta del numero di anelli costituenti la catena;
  - possibilitá di realizzazione di catene comunicative di lunghezza $(N-2)r_v$
- Svantaggi:
  - ripristino dell'ipotesi di segnali isotropi;
  - aumento della complessitá computazionale per ogni singolo robot;

[10] Valgono quindi le osservazioni ivi indicate.

V. SIMULAZIONI E CONFRONTI

I dati raccolti ed esposti in seguito sono relativi agli algoritmi della tipologia *Search-First*.

Le simulazioni vengono eseguite al variare dei seguenti parametri:

- N = numero di robot inizialmente disponibili,
- $d(BS, SS)$ = distanza del segnale sorgente dalla BS (nel seguito si indicherá $d(BS, SS) = limit$).

Come giá osservato, nei primi due algoritmi ($SF1$ ed $SF2$) si riesce ad istituire un ponte radio con alta probabilitá, fissando un limite per la distanza massima percorsa dai robot in esplorazione che verrá denominato $limit_{SF(1-2)}(N) = (2/5)Nr_s$; per quanto concerne la terza versione $SF3$ questo limite diventa ($limit_{SF3}(N) = (N-2)r_s (> limit_{SF(1-2)}(N))$).

A seconda del valore $limit_{max}(N)$, coincidente con la distanza massima in cui potrá trovarsi il SS nelle prove eseguite, le simulazioni si dividono in due categorie:

1) $limit_{max}(N) = limit_{SF(1-2)}(N) \rightarrow$ Si considerano le casistiche in cui le simulazioni che utilizzano le procedure $SF1$ ed $SF2$ (e quindi anche $SF3$) hanno quasi sicuramente buon esito: si privilegia il confronto, dal punto di vista della tempistica, tra le prestazioni ottenute con la procedura $SF1$ e con $SF2$, dato che tali algoritmi utilizzano metodi molto simili nella risoluzione del problema. Vengono comunque riportati i risultati relativi alle simulazioni con la procedura $SF3$ per dare un'idea della superioritá di quest'ultima rispetto alle prime due.
2) $limit_{max}(N) = limit_{SF3}(N) \rightarrow$ Si eseguono una serie di simulazioni impostando come limite superiore di $d(BS, SS)$, il valore massimo di tale distanza per cui l'$SF3$ assicura la risoluzione del problema: si mette in evidenza la maggiore efficacia dell'$SF3$, il quale, a paritá di N, é in grado di risolvere un insieme di casistiche molto piú ampio rispetto agli algoritmi $SF1$ ed $SF2$.

In conclusione si riportano i risultati (Tabella III) ottenuti da alcune simulazioni, relative alle procedure $SF1$ ed $SF2$, nel caso di intensitá dei segnali $S(V(k))$ non uniforme nello spazio: tale situazione é stata riprodotta impostando un limite massimo, per le curve di intensitá di tali segnali, variabile in un intorno circolare dei veicoli. Il motivo principale dell'esposizione di tali risultati é evidenziare che l'ipotesi di uniformitá dei segnali non é necessaria per il corretto svolgimento delle procedure $SF1$ ed $SF2$, dato che la creazione del collegamento tra BS e SS avviene esclusivamente tramite operazioni indipendenti tra un veicolo e l'altro.

Si é scelto di considerare, per tutti i casi, un insieme di valori di N e di fissare, per ognuno di essi, due grandezze del il parametro $limit$: una relativa al posizionamento della sorgente non molto lontano dalla BS, l'altra corrispondente alla collocazione del SS al limite $limit_{max}(N)$ previsto. Si eseguono 50 simulazioni per ognuno dei casi considerati ed utilizzando i tre diversi algoritmi; si precisa che ciascuna simulazione prevede l'impiego delle stesse condizioni di partenza per ogni algoritmo. I risultati vengono espressi sotto forma di valor medio del totale dei tempi di esecuzione di ogni singola prova. Si é fissato, inoltre, un limite superiore al tempo di esecuzione pari a $t_{max} = 5000$: una generica simulazione é valida quando termina entro t_{max}, in caso contrario la prova si ritiene 'fallita'. Le tabelle illustranti i risultati ottenuti in ogni singola simulazione vengono riportate in appendice B.

A. CASO 1: $limit_{max}(N) = (2/5)Nr_s$

Imponendo questo limite alla massima distanza che i robot possono percorrere in esplorazione, ci si riferisce ai casi in cui tutte e tre le implementazioni assicurano il completamento del ponte radio. Per ogni valore di N sono state considerate due grandezze di $limit$: $limit_{SF(1-2)}(6)$, che rappresenta il caso in cui il SS é nelle vicinanze della BS, e $limit_{SF(1-2)}(6)$.

Dai tempi riportati in tabella (I) é evidente come, a paritá di agenti disponibili, all'aumentare del valore $limit$ si abbia un notevole incremento dei tempi di realizzazione del ponte. Ció dimostra che l'approccio di creazione del collegamento mediante esplorazione continua in direzioni casuali, diventa una procedura troppo lenta all'aumentare dello spazio che i veicoli devono percorrere in entrambi i sensi di marcia. Si noti, inoltre, come le prestazioni relative all'$SF2$, nel quale la ricerca viene progressivamente semplificata grazie al sistema di assegnazione di target sempre piú prossimi alla base, siano leggermente migliori rispetto al primo approccio. In particolare, il fatto che alcune simulazioni, alla massima distanza considerata, falliscano con l'algoritmo $SF1$ e siano invece terminate con successo dall'$SF2$, testimonia i miglioramenti ottenuti dall'inserimento di una gerarchia nella realizzazione del ponte secondo la procedura di 'ricerche di taglia decrescente'.

B. CASO 2: $limit_{max}(N) = (N-1)r_s$

Si confrontano le potenzialitá delle tre procedure, da un punto di vista di efficienza ed affidabilitá: un'algoritmo che presenti tali caratteristiche ai piú alti livelli, dovrebbe essere in grado di realizzare un ponte di lunghezza massima pari a $(N-1)r_s$ (con N il numero di robot disponibili) in tempi non troppo elevati.

Come giá osservato in precedenza, nel $CASO1$ gli algoritmi $SF1$ $SF2$ ed $SF3$ competono nella velocitá di risoluzione del problema; ma utilizzando delle specifiche che sfruttino a pieno le potenzialitá dell'$SF3$, si dimostra che quest'ultima implementazione é molto piú performante delle precedenti, poiché in grado di realizzare potenzialmente connessioni su distanza piú larga rispetto alle prime due.

Si é evidenziata tale caratteristica confrontando le prestazioni dei tre algoritmi nei casi di posizionemento della sorgente in $limit_{SF(1-2)}(N)$ e $limit_{SF3}(N)$ (TabellaII). Con un totale di 6 robot a diposizione i primi due algoritmi riescono a risolvere il problema senza fallimenti: la distanza della SS nel caso $limit_{SF3}(8)$, infatti, non é eccessivamente superiore rispetto a $limit_{SF(1-2)}(8)$ e ció consente, anche in questa situazione, la realizzazione del ponte in un lasso di tempo contenuto. Non appena si aumenta il numero di veicoli disponibili, il valore $limit_{SF3}(N)$ eccede notevolmente rispetto a $limit_{SF(1-2)}(N)$: l'$SF1$ e l'$SF2$, in questo caso, dimostrano prestazioni deludenti, con, rispettivamente, un 56% ed un 40% di prove non portate a termine.

VI. Fase Sperimentale

In questa sezione si vuole presentare la sperimentazione reale degli algoritmi ideati e simulati tramite MATLAB.

Per tale fine occorre inanzitutto effettuare una breve trattazione circa l'apparato hardware utilizzato. Si distinguono principalmete due dispositivi:

- Tmote Sky
- e-Puck robot

Il primo dei due é sostanzialmente una scheda dotata di memoria Flash e microcontrollore, caratterizzata da un interessante sistema sensoriale(sensori di luce, umiditá, temperatura). Tale struttura é inoltre dotata di un trasmettitore e di un ricevitore radio, con annesso un rivelatore digitale di intensitá di segnale in ricezione. Per la comunicazione cablata invece i Tmote hanno a disposizione due porte seriali, chiamate UART1 e UART0, che vengono utilizzate rispettivamente per instaurare un protocollo comunicativo attraverso la porta USB e attraverso una porta generica supplementare(che verrá poi utilizzata per l' e-Puck sottostante). Solitamente la porta USB, identificabile anche a livello macroscopico, si presta bene per connessioni con personal computer per molti motivi: infatti oltre alle ovvie operazioni di installazione del codice nel Tmote, si puó pensare di mantere collegato un Tmote ad un PC per l'elaborazione dei dati provenienti dal mote in maniera piú articolata, visto che quest'ultimo dispone sicuramente di memoria e capacitá computazionali piú limitate. La scrittura del codice per i Tmotes si effettua seguendo i dettami del linguaggio nC, il quale costituisce una variante del piú noto C. Per programmare questi dispositivi ci si deve appoggiare al sistema operativo TinyOS, il quale agisce sotto una piattaforma Linux, consentendo il caricamento del codice nei Tmotes direttamente da riga di comando. Si sottolinea infine che, per questo tipo di programmazione, risulta fondamentale disporre delle librerie standard consultabili sul sito Internet *www.tinyos.net*. L'e-Puck invece é un robot di forma cilindrica in grado di

Fig. 23. Tmote Sky.

compiere movimenti onnidirezionali e con buona precisione, per mezzo di una coppia di ruote governate da altrettanti motori. Ogni e-puck é caratterizzato da un microprocessore e da una memoria e possiede in dotazione una telecamera che si presta bene per operazioni di ricerca obiettivi. É necessario peró osservare che le potenzialitá offerte da questa telecamera sono piuttosto limitate e cosí il suo impiego si restringe ad applicazioni di individuazione di un determinato colore. Altre potenzialitá dei robot sono rappresentate da dei sensori supplementari. In aggiunta a quelli di suono e velocitá sono da evidenziare anche una dozzina di sensori di prossimitá piazzati sulla circonferenza esterna del robot. Un'interessante applicazione per questi é data dall'utilizzo in concomitanza con un gruppo di Led, i quali si trovano disposti in corrispondenza di tali sensori, per evidenziare ed eventualmente gestire delle situazioni di collisione con altri oggetti fisici. Il codice viene scritto in C ed il suo caricamento negli e-Puck puó avvenire comodamente per via Bluetooth, come del resto anche la comunicazione fra due diversi robot. Per la compilazione del codice si é fatto uso del software MPLAB, il quale richiede il successivo impiego di un BootLoader per l'operazione di scrittura nella memoria dei Puck. L'obiettivo

Fig. 24. e-Puck robot.

che ci si é posti per lo sviluppo di questo progetto é di implementare degli algoritmi di ricerca per un sistema multi-robot. Nel voler applicare questi algoritmi in un contesto reale si rende necessario innanzitutto definire la struttura fisica di target, base station e nodi di ricerca. Si é scelto di utilizzare due Tmotes fissi(e quindi dotati di batterie per garantire la tensione di alimentazione richiesta) per rappresentare il target e la BS. Ogni agente ricercatore é invece costituito da un robot e-Puck sopra il quale si trova un Tmote, interfacciato con esso attraverso una basetta di connessione. L'alimentazione del mote in questo caso viene prelevata direttamente dalla batteria in dotazione all'e-Puck. Come si é giá sottolineato piú volte fin'ora, l'approccio di soluzione di tipo CF risulta pressoché inapplicabile in ambienti simulativi e quindi tantomeno ci si aspetta in situazioni pratiche. Per tanto si é volta l'attenzione alla concretizzazione degli algoritmi SF. In una fase iniziale si é pensato di implementare in laboratorio l'algoritmo SF2. Questo infatti, rispetto alla versione precedente ha il vantag-

Fig. 25. Nodo di ricerca.

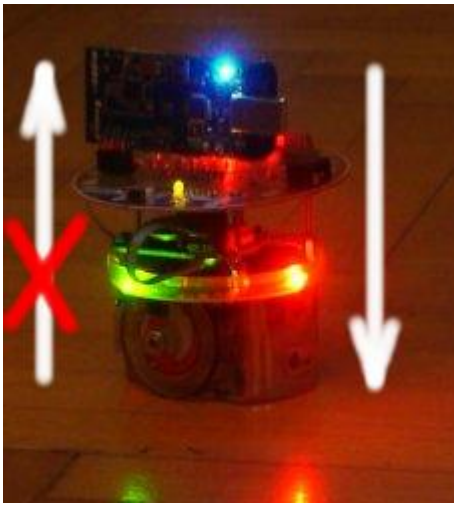

Fig. 26. Comunicazione unidirezionale.

gio di accorciare i tempi di ricerca in conseguenza a una diminuzione del raggio di ricerca dei robot. Tale pregio peró in laboratorio non risulterebbe affatto apprezzabile a causa delle dimensioni molto limitate dell'ambiente di ricerca. Per rendersi conto di ció basta immaginare una situazione in cui la ricerca avvenga in un'area da 150m di raggio. In circostanze siffatte se si accorciasse il raggio di ricerca di 50m, a paritá di velocitá dei dispositivi, il tempo d'esecuzione dell'algoritmo ne verrebbe ridotto in maniera importante. In termini implementativi le complicazioni che intervengono adottando questa miglioria sono essenzialmente due:

- la base station ha una funzione attiva durante la fase di ricerca
- si richiede una comunicazione bidirezionale fra Tmote ed e-Puck

Benché a livello algoritmico la differenza di codice non dovrebbe risultare molto pesante si é preferito, peró, adottare l'approccio descritto SF1. Infatti, usando SF2, si incorre in problematica pratica non indifferente, legata ad una questione di intensitá di segnale ricevuto che verrá a breve esaminata piú in dettaglio. In termini sintetici si trova che emergono dei problemi nella riscalatura della configurazione geometrica che i robot ancora in ricerca devono descrivere ruotando attorno alla base station. Vale comunque la pena osservare che, se si potessero effettuare dei calcoli di gradiente d'intensitá di segnale, questo problema non si manifesterebbe. Ad ogni modo, il compromesso offerto dal passaggio SF2 $\rightarrow$ SF1, consente al progettista di limitare la comunicazione fra mote e puck, inquanto l'informazione che si deve scambiare non comprende piú una misura odometrica del robot, per la precisione della quale anche in letteratura si sono trovate lunghe diatribe. Durante lo sviluppo dell'opera si sono manifestate situazioni che hanno dato origine ad un certo numero di considerazioni:

1) Problema sulla certezza di corrispondenza fra le affermazioni ti ho trovato - sono stato trovato.: Per stabile nella pratica quando un robot ha trovato il target si considera come trovato quando fra i due si stabilisce una connessione "stabile". Per stabile si é fissata la ricezione di 10 messaggi da parte del target. Questo fa capire che in realtá potrebbe verificarsi che il robot non sa di aver trovato il target mentre invece magari il target pensa di essere stato trovato. L'idea piú semplice per ovviare a questo problema sarebbe quella di usare una routine di acknowledgement da parte del robot ricercatore. Tale routine, che per completezza si trova come commentata nel codice C, peró non si é potuta utilizzare. Infatti, una delle complicazioni che sono scaturite dal riscalamento del problema in dimensioni di taglia inferiore rispetto a quelle previste dalla strumentazione hardware é che l'acknowledgement viene inviato in ogni caso. Si ha cioé che al target richiedente l'acknowledgement non intaressa che al robot ricercatore il messaggio giunga con un certo livello di potenza di segnale, ma intaressa che arrivi e basta. In questo modo risulterebbe che il messaggio, concepito come ricevuto per il target, in realtá é stato scartato dal robot. Analizzando in profonditá la soluzione adottata peró risulta che non si ha ancora la certezza di ricezione. Si puó comunque pensare di alzare la soglia di messaggi ricevuti dal target per considerare la connessione come stabile.

In ogni caso, per l'utente che sperimenta il codice, la verifica di ricezione da parte del robot ricercatore con un certa intensitá di segnale é relativamente facile. Infatti si é apportato il semplice accorgimento di far mutare lo stato di un led rosso ad ogni ricezione di questo tipo. L'effetto che ne si ricava a livello visivo é quello di un'intermittenza luminosa ad ogni situazione di comunicazione stabile. Al raggiungimento del numero limite di ricezioni preimpostato(in questo caso dieci) si osserva invece l'accensione dei un led di colore blu.

2) Problema nei calcoli del gradiente di intensitá di segnale.: In teoria l'idea della perlustrazione dell'ambiente ruotando attorno alla base station si basa sull'ipotesi di poter misurare l'intensitá del segnale inviato dalla base stessa. Non appena un'agente sente questa diminuire deve fermarsi per cambiare angolazione in modo da iniziare una esplorazione rettilinea in una nuova direzione. Infatti una diminuzione di potenza di segnale ricevuto idealmente significa che si é

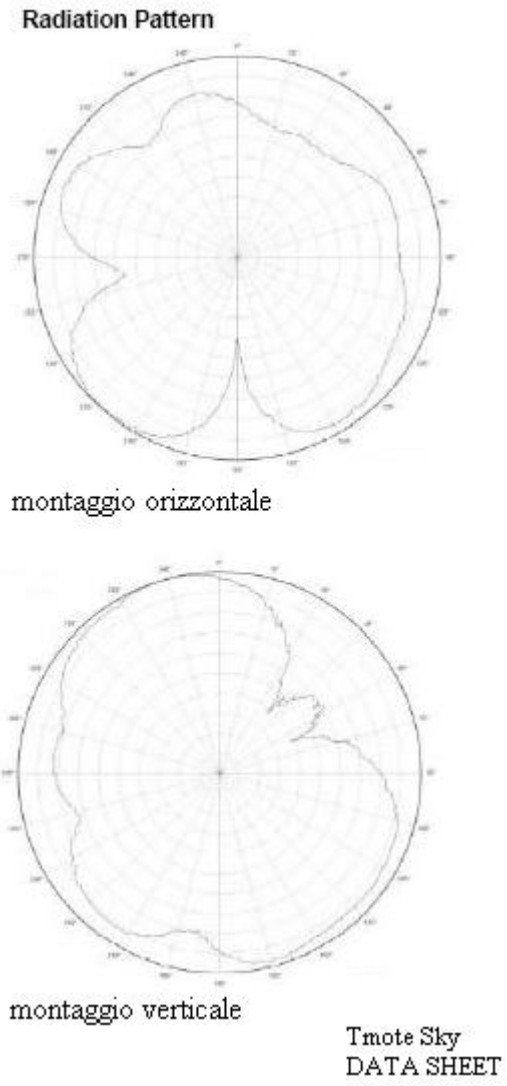

Fig. 27. Tmote Sky data-sheet: radiation pattern. Figura in alto montaggio orizzontale, figura in basso montaggio verticale.

passato il centro della BS. Questa corrispondenza peró, a causa delle complicazioni che emergono nella realtá, nella pratica non si puó affatto concludere a priori. Un dato che lascia intendere al lettore il motivo dell'ultima affermazione é che a paritś di posizione fra due Tmotes il segnale ricevuto puó oscillare di intensitá fino a 20 dBm. Questo fatto risulta essere il risultato di fattori di vario genere: si tratta infatti sia di caratteristiche proprie della struttura hardware che di disturbi esogeni. Invece che semplicemente attribuire il tutto ai disturbi di varia natura presenti nell'etere si puó cioé fare un'analisi piú attenta dei dispositivi considerati. Da un lato si ha un Tmote che trasmette segnale in maniera necessariamente incostante in intensitá e dall'altro si ha un altro Tmote che deve captare tale segnale. Dai data sheet si evince che, considerare come perfettamente circolare la sensibilitá di ricezione dei segnali radio dei Tmotes, é in realtá un approssimazione grossolana se si vuole agire su brevi distanze. Il costruttore illustra per la ricezione due differenti situazioni in base alla scelta di montaggio dell'antenna. Se in aggiunta a questo si pensa alla stocasticitá del montaggio, che viene effettuato da un operatore umano, e a quella propria del singolo componente rispetto al modello della sua serie, si intuisce come l'ipotesi di idealitá trasmissiva si dimostri sempre meno veritiera man mano che ci si addentra nel dettaglio. Ad ogni modo, per l'esperienza di laboratorio, si é cercato di utilizzare una tipologia di Tmotes dotati di antenna disposta perpendicolarmente rispetto alla struttura planare, in maniera da ottenere un comportamento il piú possibile vicino all'isotropia per la ricezione.

Si potrebbe ovviare a tale inconveniente attraverso un'operazione di media nel tempo. In effetti questa tecnica potrebbe funzionare se i robot in questione compissero dei tragitti lunghi almeno decine di metri, in modo da consentire un tempo sufficiente a donare realismo alla media temporale. Nel caso in questione peró le distanze risultano enormemente contratte. Questo comporta un'immediatezza delle operazioni che non lascia tempo a calcoli che richiedano certi intervalli temporali per essere eseguiti. Se si procedesse programmando tutti i nodi in questa maniera, si ricaverebbe in un rallentamento molto accentuato per il processo di ricerca complessivo. Per questo motivo in laboratorio si é scelto di non effettuare calcoli di gradiente di intensitá di segnale ricevuto. Affinché l'algoritmo possa funzionare ugualmente si deve allora escogitare quache accorgimento ulteriore. Dopo qualche riflessione, si é capito che, per ottenere l'effetto di rotazione attorno alla BS, basta pensare di poter predisporre i robot all'istante iniziale secondo una geometria prestudiata invece che casualmente. Per essere precisi si vuole arrivare a formare un poligono regolare circoscritto ad una circonferenza fittizia che pone al proprio centro la base station. Si osserva che una siffatta ipotesi non é in realtá cosí astratta e impensabile. Infatti non risulta difficile pensare che un'operatore umano possa svolgere questo compito. Inoltre si puó ottenere il risultato sperato magari implementando una routine apposita preposta per questo fine. In definitiva allora, il pegno che si paga allora per aver riscalato il problema, é l'aggiunta di un'ipotesi ed nella specifica applicazione un trade off di questo tipo risulta abbastanza soddisfacente.

3) Problema di cattura dei messaggi Tmote → robot.: Per come si é strutturata la comunicazione fra Tmote e robot si ha che il robot, mentre svolge la sua perlustrazione rettilinea, resta in ascolto di eventuali segnali che gli arrivano dalla porta seriale (quindi dal Tmote soprainstallato). L'interrogazione di questa porta viene eseguita con velocitá pari a quella del processore. Per quanto questa possa essere elevata, peró, risulta evidenziabile un gap temporale in cui il Tmote invia il segnale e il robot non ascolta. In questo caso c'é una discrepanza fra le azioni di mote e puck. Il primo trova il target ma il secondo agisce come se non lo sá e quindi continua ad eseguire la sua routine standard di movimento. Per ottenere certezza nella corrispondenza delle due operazioni l'accorgimento piú saggio da prendere sarebbe di impostare un acknowledgement inviato dal robot al Tmote in risposta alla ricezione del messaggio "trovato". Studiando opportunamente il protocollo di comunicazione attraverso la porta seriale, peró, si denota la presenza di un piccolo buffer, che risolve il problema giá direttamente a livello hardware. Si ha che tale memoria puó contenere un byte corrente piú altri tre di buffer. Eventuali

byte aggiunti oltre a questi quattro vengono cioé cestinati se non vengono letti i primi in coda, in modo da far shiftare la coda svuotando gli ultimi posti del buffer. Sará quindi questo buffer a mantenere il messaggio del Tmotes fino a quando non sia avvenuta l'effettiva lettura di tale messaggio da parte dell'e-Puck.

Questi dettagli comunque lasciano intuire come l'interfacciamento software fra i due dispositivi risulti particolarmente brigoso. Si necessita infatti di scendere di livello nella programmazione, curando aspetti quali la coerenza fra le velocitá di trasmissione e ricezione, struttura e dimensione dei messaggi, ecc..

Nella realtá sperimentale si é riscontrato poi un ulteriore problema. Infatti, per quanto tutte le sottolineature appena fatte siano veritiere, queste riguardano soltanto la parte software della situazione. Nella pratica invece bisogna anche considerare che un'interfaccia cablata é necessariamente anche fisica. Questo comporta un intervento umano che, in quanto tale, é aleatorio(di fatto si tratta di saldature eseguite manualmente). In alcune prove sperimentali si sono quindi verificati degli episodi di incoerenza fra i comportamenti di Tmote ed e-Puck, che hanno suggerito un accorgimento tecnico per essere individuati. Si é cioé impostata l'accensione di un Led verde sul mote in caso di trasmissione di un dato via seriale e quella di un corrispondente Led rosso sulla parte frontale del robot a segnalazione di avvenuta ricezione dalla porta medesima. Ricercando in rete delle possibili idee per il superamento di questa difficoltá implementativa, si é scoperto come questa costituisca per un gran numero di programmatori un grosso ostacolo. Per ovviare il problema alla radice, la ditta produttrice degli e-Puck ha introdotto una variante per l'architettura hardware dei robot, la quale prevede di dotarli giá di potenzialitá comunicative in radiofrequenza. Appare evidente come un accorgimento di questo tipo costituisca una facilitazione notevole in sede di programmazione, in quanto oltre all'interfaccia si evitano i costi di start up nella programmazione in TinyOS che caratterizza il software dei Tmotes. In Fig. (28) é schematicamente riportato quanto appena menzionato. Fatte tutte queste considerazioni si possono esaminare le

Fig. 28. Versione degli e-Puck con radio integrata.

diverse fasi di lavoro che si sono richieste per la realizzazione sperimentale dell'algoritmo SF1. Dopo un periodo iniziale di start up, necessario per prendere confidenza con i linguaggi di programmazione da utilizzare, si é prestata attenzione alla riscalatura del problema a dimensioni compatibili con quelle a disposizione in laboratorio. Per questo fine si sono andati a modificare la potenza trasmissiva via radio dei Tmotes e la loro sensibiltá in ricezione. Il risultato ottenuto dopo un certo numero di combinazioni per questi due parametri, é stato quello di consentire la comunicazione radio solo ad una distanza minore o uguale ad una quindicina di centrimetri. Sistemato questo aspetto fondamentale, ci si é preoccupati del

Fig. 29. Regolazione dell'intensitá del segnale radio.

moto dei robot. Come giá preannunciato, la routine mozionale degli e-Puck attorno alla base station, é tale da formare un poligono regolare. Questo poligono é circoscritto rispetto ad una circonferenza di centro la BS e raggio pari a quello di visibilitá della stessa. Inoltre si ha che il numero dei lati che lo compongono dipende da quello dei nodi che si utilizzano per la ricerca. Nel caso preso in esame si é fatto riferimento ad una struttura esagonale, con gli agenti disposti in tre dei sei vertici a diposizione. In figura si illustra, per chiarezza espositiva, il comportamento di un robot attorno alla base station. Le procedure di movimento che si evicono da questa situazione sono quindi tre: una perlustrazione rettilinea in avanti, un ritorno alla base, muovendo in retro marcia, ed una rotazione finale in senso orario di $\frac{2\pi}{n_{lati}}$ radianti (in questo caso $60°$). Da sottolineare che in ciascuna di queste procedure di movimento nascono delle inevitabili imperfezioni per la geometria del sistema. In primo luogo, per quanto siano precisi i movimenti rettilinei, si ha sempre a che fare con un pavimento non ideale, cioé contenente buche e privo di perfette condizioni d'attrito per l'aderenza delle ruote su di esso. Questo é stato il principale motivo della scelta di abbandonare la piattaforma a disposizione presso il laboratorio Navlab come ambiente di sperimentazione. Infatti, questa struttura fisica si compone di due piastre, fra le quali vi é un piccolo scanso. Passando per questo scanso necessariamente i robot deviano la loro traiettoria o rimangono addirittura bloccati. Si sarebbe anche potuta sistemare la situazione adottando una classe di robot differenti, dotati di capacitá motorie superiore, ma un approccio di questo tipo risulta pressoché impensabile per fini didattici.

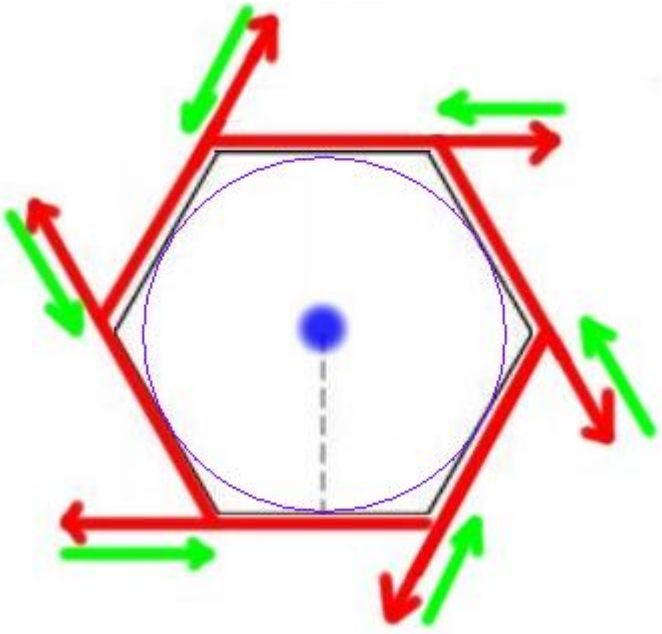

Fig. 30. Routine di movimento per l'esplorazione.

Inoltre si osserva che la procedura di rotazione termina con

Fig. 31. All-terrain e-Puck.

il bloccaggio delle ruote in modo abbastanza brusco. Questo fatto comporta un piccolo slittamento delle stesse con una conseguente variazione dell'angolo di sterzo rispetto a quello preimpostato. Tutte queste piccole imprecisioni fanno capire che, per quanto la geometria di partenza dei robot possa essere studiata accuratamente, asintoticamente, in caso di mancato ritrovamento del target, il sistema sará destinato ad andare alla deriva.

Successivamente si sono sviluppati aspetti software relativi alle operazioni di interazione fra i Tmotes, durante la loro comunicazione, ed infine l'interfacciamento mote-robot. Per queste due ultime sezioni del lavoro, si mette in luce come nel codice scritto in C si possono riscontrare delle strutture sovrabbondanti rispetto a quelle necessarie per la realizzazione dell'algoritmo SF1. Si é fatto ció anche in un ottica di possibili sviluppi futuri del lavoro, in modo da non richedere una totale ridefinizione del codice. Per esemplificare quanto detto basta considerare il tipo di pacchetti scambiati fra i motes:questi, oltre alle varie intestazioni, contengono un campo per l'ID del mittente e un campo da 16 bit inutilizzato. Adottando la soluzione SF2 questo avrebbe dovuto contenere la riduzione del raggio di ricerca, ottenuto tramite sottrazione fra il tragitto rettilineo totale da percorrere per la perlustrazione e una misura odometrica.

Altro aspetto notevole é che, al ritrovamento del target, il puck non vede arrivare sulla porta seriale un booleano(in effetti gli servirebbe solo quello), ma un numero intero. In questo modo si é pensato di stabilire una corrispondenza fra numero inviato e conseguente operazione da svolgere per il robot. Si intuisce allora come estensioni di questo lavoro possano risultare agevolate da questi accorgimenti tecnici.

Una volta curati tutti questi aspetti, gli esperimenti si sono rivelati abbastanza agevoli. A seguito di un certo numero di prove in vari ambienti si é riscontrato che un pavimento a parquet si presta molto bene come superficie d'appoggio per i dispositivi in questione. Il movimento degli agenti infatti si é dimostrato piuttosto ordinato e rapido, come richiesto dalla procedura. In definitiva quindi, nonostante gli ostacoli costituiti dalla variablitá del segnale e dalle imperfezioni fisiche, quali le non idealitá della componentistica e l'interfaccia, i risultati sperimentali si sono rivelati piuttosto soddisfacenti. Nelle immagini a seguito, che costituisce la situazione finale di un filmato allegato a questa relazione, si evidenzia il ponte creato dalle unitá ricercatrici fra il target di ricerca e la base station.

Fig. 32. Situazione finale: visuale $n^{\circ}1$.

VII. Conclusioni e Possibili Sviluppi Futuri

Nel limitato tempo a disposizione che si é potuto dedicare alla tematica si é cercato di affrontare il piú ampio numero di aspetti d'interesse per tale argomento. Tuttavia si é perfettamente consapevoli che certi aspetti potevano essere migliorati o sviluppati in maniera piú approfondita o seguendo diverse ipotesi di partenza. Pensando a questo articolo come ad

Fig. 33. Situazione finale: visuale $n^{\circ}2$.

un possibile inizio di lavori futuri piuttosto che un qualcosa che termina in sé stesso, si preferisce dare enfasi ai possibili sviluppi anziché sottolineare le conclusioni raggiunte giá ampiamente espresse in precedenza. Pertanto si suggeriscono alcuni possibili sviluppi futuri:

- l'implementazione effettiva dell'approccio alternativo di CF, presentato nella sezione (III-K);
- la realizzazione del controllo del movimento circolare, sia per gli algoritmi CF che $SF3$ attraverso un PID, piú complesso rispetto al semplice P proposto; ovviamente tale introduzione complica un ulteriore il modello computazionale ipotizzato a disposizioni dei singoli nodi;
- l'ottimizzazione della ponte di comunicazione una volta stabilita la connessione con la BS per gli algoritmi SF; in molti casi il canale comunitcativo é infatti composto da un numero di nodi superiori a quello necessario, oppure la posizione dei nodi ripetitori non é quella ottimale: sarebbe in certi casi possibile modificare la posizione dei nodi della catena per garantire un maggior rapporto SNR dei segnali scambiati. Tali operazioni di ottimizzazione dovrebbero rivelarsi piú agevoli nei casi degli algoritmi $SF2$ ed $SF3$, dove di fatto é giá stata stabilita un'ordine gerarchico sulle catene formate, mentre dovrebbe rivelarsi piú complicato nel caso dell'$SF1$, dove é presente una rete di ripetitori priva di alcuna gerarchia: in questi casi andrebbe anche implementata una procedura per l'identificazione di quali sono i nodi ridondanti che non servono a costituire un ponte comunicativo;
- data la totale generalitá degli algoritmi SF proposti sarebbe possibile modificare le soluzioni implementative presentabili in modo tali da renderle efficaci anche nel caso in cui la SS fosse mobile all'interno di una certa zona anziché fissa;
- l'implementazione sperimentale dell'algoritmo $SF2$ che presenta un buon trade-off fra complessitá di programmazione e prestazioni ottenute; ovviamente tale possibilitá richiede una maggior attenzione all'implementazione pratica viste le complicazioni introdotte dal codice.

APPENDIX A
RIFERIMENTI AL CODICE MATLAB

Si riporta, per ogni algoritmo descritto, un breve elenco delle funzioni MATLAB implementate con una sintetica descrizione.

A. Funzioni matlab utilizzate nell'algoritmo CF

- *accordo.m*: discrimina con quale agente effettuare un eventuale adozione;
- *affigliaggio.m*: routine per la sovrapposizione dei nodi perdenti sui vincenti durante il processo di fusione fra due catene;
- *aggiorna_segnali.m*: funzione preposta all'aggiornamento dei segnali inviati e ricevuti ad ogni iterazione dai veicoli e dalla BS;
- *avanzamento_sincronizzato.m*: implementa la fase a turni del processo di fusione;
- *calcolo_gradiente.m*: routine di stima del gradiente di un segnale;
- *copia_movimento.m*: routine creata per controllare il movimento circolare di un qualsiasi discendente: essa prevede il controllo della posizione del veicolo rispetto al padre ad al figlio, in modo tale da mantenere la connettivitá, ed imposta la velocitá del robot in base alla sua posizione all'interno della catena;
- *creazione_BS.m*: routine preposta alla creazione dell'oggetto Base Station;
- *creazione_veicoli.m*: permette di posizionare in maniera random un certo numero di robots e, tenendo conto del loro raggio di ingombro, garantisce che non ci siano sovrapposizioni;
- *creazione_SS.m*: routine preposta alla creazione dell'oggetto Source Signal, collocata in una posizione casuale all'interno del raggio di copertura del network;
- *disegno_BS.m*, *disegno_veicoli.m* e *disegno_SS.m*: curano la rappresentazione grafica, con annesso campo di visibilitá degli oggetti indicati;
- *avanzamento_sincronizzato.m*: implementa la fase a turni del processo di fusione;
- *esegui_CF.m*: main eseguibile, da questa funzione possono essere settati tutti i parametri per ciascuna simulazione;
- *genealogia.m*: discrimina la posizione ed il ruolo di ogni singolo nodo all'interno di un processo di fusione
- *initial_conditions.m*: genera aleatoriamente le condizioni iniziali del problema: collocazione ed angolazione dei robot, distanza della SS dalla BS;
- *movimento_circolare.m*: 'funzione di controllo' del movimento che puó essere utilizzata da un qualsiasi veicolo $V(k)$ in contatto con un robot $V(z)$, per effettuare uno

spostamento 'a gradiente costante' rispetto al segnale $S(V(z))$;
- *movimento_rettilineo.m*: implementa il moto rettilineo di ogni robot;
- *muovi_veicoli.m*: funzione principale del movimento, raccoglie tutte le casistiche di moto e richiama le funzioni ausiliari (quali *movimento_circolare.m* o *movimento_rettilineo.m*) in base allo stato del singolo agente.

B. Funzioni matlab utilizzate in algoritmo $SF1$ ed $SF2$

Non sono riportate le funzioni giá utilizzate nell'algoritmo CF, vale la medesima descrizione.

- *aggiorna_segnali_first.m*: analoga alla funzione di aggiornamento del segnale per l'algoritmo CF, modificata per le peculiaritá di $SF1$;
- *disegno_ripetitori.m*: fa un grafico del raggio di visibilitá dei nodi, una volta che questi fanno parte della catena comunicativa e fungono per l'appunto da ripetitori;
- *esegui_SF1.m*: main eseguibile, da questa funzione possono essere settati tutti i parametri per ciascuna simulazione (tale funzione esterna di fatto non fa altro che richiamare *main_first.m*, che effettivamente implementa l'algoritmo);
- *muovi_veicoli_first.m*: funzione alla quale é preposta ogni funzione di movimento dell'algoritmo;
- *verify_end_search.m*: verifica se il ponte comunicativo con SS é stato effettivamente stabilito e termina l'algoritmo. Per l'algoritmo $SF2$ di fatto vengono utilizzate le medesime funzione di $SF1$: *aggiorna_segnali_first.m*, *esegui_SF1.m* e *muovi_veicoli_first.m* vengono rispettivamente sostituite da *aggiorna_segnali.m*, *esegui_SF2.m* e *muovi_veicoli.m*.

C. Funzioni matlab utilizzate in algoritmo $SF3$

Vengono riportate solo routin non elencate in precedenza.

- *esplorazione.m*: utilizzata dagli agenti nella perlustrazione dello spazio circostante alla BS;
- *ricerca_anelli.m*: funzione usata, dopo l'elezione del leader, dai robot che si trovano all'interno della $C_{rv}(BS)$, per riportarsi ad una posizione prossima al punto in cui ha origine la catena;
- *posizionamanto.m*: permette ad un veicolo di eseguire la procedura di aggiunta alla catena attraverso i tre movimenti fondamentali in figura (20);
- *copia_movimento.m*: funzione 'di controllo' applicata ai robot durante il 'movimento a catena' che assicura il mantenimento della connettivitá.

APPENDIX B
RISULTATI DELLE SIMULAZIONI

Le seguenti tabelle contengono i risultati ottenuti da ogni simulazione eseguita: per ogni caso contemplato sono state effettuate 50 simulazioni, per ogni procedura: i valori Inf corrispondono alle simulazioni 'fallite', secondo i criteri illustrati nella sezione V.

SERIUOS ACKNOWLEDGMENT

Ovviamente il professor Schenato per la disponibilitá ad ogni chiarimento. Enrico desidera ringraziare Claudio per l'aiuto ed il sostegno nel NAVLab. Gian Antonio e Massimiliano ringraziano Angy e Cristina per la sopportazione di tutti i weekend dedicati a questo progetto e non passati con loro! Infine un ringraziamento da parte di tutti e tre i componenti del gruppo ai nostri i genitori ed il loro supporto.

NOT SO SERIUOS ACKNOWLEDGMENT

Si dichiara che durante tutto il progetto é stato utilizzato solo software legale e non sono stati maltrattati animali.

The authors would like to thank... le tastiere dei nostri personal computer per aver sopportato la scrittura e riscrittura di 4265 righe di codice; $CTRL + Z$ áncora di salvezza nei momenti bui; gli amici di MindManager per la versione trial di 21 giorni; il caro amico Burlaz (l'ing. MM) per la password di OSW; Don Vito Corleone per la protezione; i coperchi dei barattoli del caffé di casa Lungaro per le stupefacenti capacitá balistiche; il maestro Kenobi per il prestito della spada laser; Michael Jackson (da parte del solo Enrico); i baffi sporchi di 'nutella'; il manipolatore piano; l'inventore del formato PDF (in realtá ti odiamo con tutto noi stessi...); le applicazioni di FaceBook; lo pseudo-ing. OI: limite minimo e worst-case per tutto; non si ringraziano: il Furbo, lo Scudiero, HITH, Skizzo e il Pinguino.

'La vita é integrabile perché continua su un intervallo limitato'.

REFERENCES

[1] Burgard, W.; Moors, M.; Fox, D.; Simmons, R.; Thrun, S. *Collaborative multi-robot exploration,* Robotics and Automation, 2000. Proceedings. ICRA apos;00. IEEE International Conference on Volume 1, Issue , 2000 Page(s):476 - 481 vol.1, 2000.

[2] Di Chio, C.; Poli, R.; Di Chio, P. *Extending the Particle Swarm Algorithm to Model Animal Foraging Behaviour,* International Workshop on Ant Colony Optimization and Swarm Intelligence, Brussels, Belgium, September 4-7, pp. 514-515, 2006.

[3] Kantor, G.; Singh, S.; Peterson, R.; Rus, D.; Das, A.; Kumar, V.; Pereira, G.; Spletzer, J. *Distributed search and rescue with robot and sensor teams,* Proc. of the 4th Intl. Conf. on Field and Service Robotics, Japan, 2003.

[4] Kopka, H.; Daly, P. W. *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[5] Mondada, F.; *Introduction to the e-Puck robot; Robotics Systems Laboratory,* IPR - STI - EPFL.

[6] Moteiv, T-mote Sky; *Ultra low power IEEE 802.15.4 compliant wireless sensor module - Product information* 2006.

[7] Moteiv, T-mote Sky; *Ultra low power IEEE 802.15.4 compliant wireless sensor module - QuickStart Guide* 2006.

[8] Nouyan, S.; Dorigo, M. *Chain Based Path Formation in Swarms of Robots,* ANTS Workshop 2006: Brussels, Belgium, Page(s): 120-131, 2006.

[9] Pugh, J.; Martinoli, A. *Inspiring and Modeling Multi-Robot Search with Particle Swarm Optimization,* Swarm Intelligence Symposium, SIS 2007, IEEE Volume , Issue , 1-5 April 2007 Pages: 332 - 339, 2007.

[10] Rekleitis I.M.; Dudek G.; Milios E.E.; *Multi-Robot Exploration of an Unknown Environment, Efficiently Reducing the Odometry Error*. Proc. of the Int. Joint Conf. on Artificial Intelligence, Nagoya, Japan, August, 1997.

[11] Shell, M. *How to Use the IEEEtran LaTeXClass,* Journal of LaTeXClass Files, vol. 6, no. 1, January 2007.

[12] http://en.wikipedia.org/wiki/Particle_swarm_optimization

Massimiliano Lungaro M.L. is old inside: when he was 3 years old he started to complain about everything and, until now, he never stopped. Lungaro's research areas are: the weather, people unable to do their job in the correct way, the progressive absense of Spring and Fall. All these topics had been fully developed at 'Da Mario Barbiere' Department of Capelli and Basette. He his going to left Engineering in the next few months to partecipate at the prestigious Master in 'Watching Scavi'. He is the only person in the world who think that MATLAB is able to understand his insults: his way to express disappoiment or happiness towards the software are the sarcastic sentences 'spaba spaba spaba', 'cia', 'aaaaaaah' and 'va ben dai, me amigo MATLAB'. During the nigths of Winter, Mr. Lungaro became a famous guitar hero: unforgettable the duets between Massimiliano and Jimi Hendrix, Eric Clapton, Rocco Tanica and Ken il Guerriero.

Enrico Maran He is going to go in Tirolo... YELLOLAAAAAAAAAAAAAAAAAA HIII HIIIIIIIIIIIIII !

Gian Antonio Susto Escaped from Snow White and the Seven Dwarfs' magic world, accused of persecutions towards the only twelve years old innocent Snow White, G.A.S. spent his youth learning the Hattrick art and the cheating in cards and fantacalcio games arts. At the age of seventeen he decided to give vent to all his fury becaming a rugby player. When he realized that there was something of more hazardous than rugby he enrolled at engineering. In summer 2007 he receveid the prestigious diploma of *Official Taster of Whisky* at the Old Jameson Distillery of Dublin: that was G.A.'s greatest achievement and the thing he is most proud about. Great world traveller, in the 2007 G.A. was in Italy, Ireland, United Kingdom, France, Spain, Tunisy, Two Carrares, Undermarina, Carmignano-upon-the-Brenta and the Eugan Necks: in all these countries he had only learn in which local TV channel were transmitted the football matches.

Alessia Susto Gian's sister, commonly known in the universitary world like 'chea santa de so sorea de gian'. She helped Gian and Massi to remember the real meaning of life during all the project. However her presence give some problems to the catholic Enrico who hates people of *'larghe vedute'*. Today, at 22 years from her day of birth, someone still believe that Alessia doesn't exist... but someone else do (best regards to eng. D. Marin) !

$limit \rightarrow$	$limit_{SF(1-2)}(6)$			$limit_{SF(1-2)}(N)$		
$N \downarrow$	$SF1$	$SF2$	$SF3$	$SF1$	$SF2$	$SF3$
8	617	506	520	973	813	672
14	573	480	467	2.440	2.345	1.766
18	612	544	571	3.707	3.450	2.342
22	359	548	528	4.339 4 sim. fallite	3.793	2.802

TABLE I
RISULTATI DELLE SIMULAZIONI RELATIVE ALL'ALGORITMO $SF1$ NEL CASO 1 ($limit_{max}(N) = limit_{SF(1-2)}(N)$).

$limit \rightarrow$	$limit_{SF(1-2)}(N)$			$limit_{SF3}(N)$		
$N \downarrow$	$SF1$	$SF2$	$SF3$	$SF1$	$SF2$	$SF3$
6	333	288	390	1.415	1.281	1.005
10	1.138	919	831	4.489 28 sim. fallite	4.055 20 sim. fallite	2.747

TABLE II
RISULTATI DELLE SIMULAZIONI RELATIVE ALL'ALGORITMO $SF1$ NEL CASO 2 ($limit_{max}(N) = limit_{SF3}(N)$).

$limit \rightarrow$	$limit_{SF(1-2)}(6)$		$limit_{SF(1-2)}(N)$	
$N \downarrow$	$SF1$	$SF2$	$SF1$	$SF2$
8	304	417	695	737
14	308	361	1.748	1.636

TABLE III
RISULTATI DELLE SIMULAZIONI RELATIVE ALL'ALGORITMO $SF1$ NEL CASO 3: INTENSITÁ DEI SEGNALI RELATIVI ALLE UNITÁ ROBOTICHE ANISOTROPA.

$N = 8$, $limit = limit_{SF(1-2)}(6)$		
$SF1$	$SF2$	$SF3$
423	361	305
685	806	802
906	227	201
435	395	247
861	789	336
224	224	364
224	540	317
419	237	333
219	218	412
1320	664	739
441	221	371
663	663	830
433	394	470
248	231	294
441	222	401
440	238	465
645	584	573
1092	1092	1219
676	547	364
667	231	278
1355	1537	1784
502	440	506
852	619	615
688	606	343
709	885	909
232	232	391
468	402	352
1319	1214	1040
267	230	214
222	222	460
1067	659	739
226	226	225
1132	1120	551
223	223	282
645	576	469
458	403	390
912	448	570
1083	882	1116
216	231	290
669	218	260
956	273	252
615	553	441
631	539	236
641	423	575
732	732	891
364	302	253
1120	464	539
807	436	468
1150	1310	1541
641	632	552

TABLE IV
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 1 ($limit_{max}(N) = (2/5)Nr_v$).

$N = 8$, $limit = limit_{SF(1-2)}(8)$		
$SF1$	$SF2$	$SF3$
888	374	604
872	668	501
1693	1625	578
1110	872	1422
439	419	554
641	661	683
1265	1117	497
863	570	609
837	687	710
1087	933	1143
863	897	568
1538	1484	482
1505	777	1046
1272	1213	482
460	466	512
869	801	585
863	667	1042
1487	1426	1168
1087	1173	388
860	696	687
1123	668	684
1099	928	729
1316	1193	484
1071	878	495
439	522	670
1473	1518	744
441	429	519
1059	781	671
649	445	680
1291	626	845
1104	1047	500
480	365	613
1310	1200	638
453	437	540
693	853	688
906	845	797
667	734	490
909	414	494
1099	474	446
852	675	898
859	725	920
1236	1392	615
1371	1177	790
630	725	889
1264	1378	572
696	341	485
1025	985	941
896	774	925
1074	1028	769
857	774	485

TABLE V
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 1 ($limit_{max}(N) = (2/5)Nr_v$).

$N = 14$, $limit = limit_{SF(1-2)}(6)$		
$SF1$	$SF2$	$SF3$
424	472	375
51	385	359
421	384	358
1121	544	524
377	377	352
385	385	361
380	438	356
389	389	364
435	558	496
69	388	362
413	389	430
755	655	469
1533	1525	1525
385	389	363
382	432	378
1128	388	396
1486	754	754
765	765	748
414	394	487
385	377	364
393	390	616
389	389	364
385	385	360
743	545	519
388	388	363
410	389	364
417	540	359
400	398	746
397	384	360
59	386	359
1477	385	427
388	388	603
383	426	358
743	392	366
380	489	354
755	701	504
1163	1008	725
1142	900	366
440	391	383
386	389	363
578	514	596
554	841	836
508	385	302
545	587	553
635	778	634
373	674	658
158	243	185
569	517	556
1180	1089	854
555	448	286

TABLE VI
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 1 ($limit_{max}(N) = (2/5)Nr_v$).

$N = 14$, $limit = limit_{SF(1-2)}(14)$		
$SF1$	$SF2$	$SF3$
2298	2521	1611
2250	1977	1799
2277	1907	1560
3378	2971	1404
2319	2692	2821
2995	2524	1871
2697	2679	1830
2216	1745	2035
2253	2120	1388
2682	3139	1401
2318	2042	1467
1910	2287	1443
2658	2461	2016
2637	1714	1312
1918	1902	1535
2276	2028	2096
1550	2948	1308
2972	2517	2105
2286	2012	1397
2654	2684	1818
1938	1424	1254
2266	2006	1839
3086	3245	1836
2289	1330	1412
2308	2098	1946
3034	2681	2076
3004	3120	2436
2273	2236	2086
2264	1876	1810
2660	2914	1773
2257	1692	1357
2703	2621	2048
1929	2243	1349
3011	2733	2559
2674	2844	2005
1571	1553	1238
2673	1735	1433
2589	2612	1993
2277	2815	2016
2258	1813	1540
2074	1452	1178
2585	2464	1814
2271	2558	1676
2578	2287	1640
1754	2085	1296
2496	2271	1843
2845	1931	1593
2785	2341	1689
2858	2874	2046
2045	1541	1868

TABLE VII
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 1 ($limit_{max}(N) = (2/5)Nr_v$).

$N = 18$, $limit = limit_{SF(1-2)}(6)$		
$SF1$	$SF2$	$SF3$
1021	1002	1071
50	488	464
544	505	718
51	499	472
486	498	471
61	493	468
1502	501	617
484	493	466
539	497	469
489	496	590
530	499	471
495	495	467
553	497	620
50	497	609
965	691	789
982	496	479
992	978	1077
495	495	468
487	502	609
497	482	470
509	498	470
504	497	469
1005	497	469
490	490	462
54	498	470
1009	494	467
1958	983	1076
491	498	471
1466	496	468
1035	987	1096
495	495	468
55	495	467
532	497	469
501	497	470
530	496	470
500	500	618
540	666	592
555	509	610
510	495	468
492	490	462
754	614	688
430	402	385
614	646	518
495	425	587
851	856	878
614	725	894
1202	442	418
485	441	446
549	414	424
434	483	573

TABLE VIII
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 1 ($limit_{max}(N) = (2/5)Nr_v$).

$N = 18$, $limit = limit_{SF(1-2)}(18)$		
$SF1$	$SF2$	$SF3$
3344	2859	2008
3873	3280	2064
3882	3921	2176
3883	3564	2445
3864	4179	1849
3871	3741	2364
3393	2808	2072
3890	4533	1963
4419	4858	2643
3457	3574	2377
3859	3563	1956
3912	3375	2280
4372	3703	2513
2953	2875	2370
4395	4660	3825
3461	2590	2116
4368	4721	2360
2936	2625	2467
2935	2785	1796
3411	3107	2411
3340	2945	2182
3390	3256	2326
3884	2738	2397
3365	2834	2411
3423	3235	2101
3849	2703	2394
3876	3758	2775
4383	4379	2495
3418	3583	2078
3427	3340	2519
3866	2900	1816
3437	4095	2342
2950	2394	2302
2907	2133	2176
4359	4571	2959
3428	2916	2285
4354	4453	1966
4830	4252	3518
3895	3011	1972
3426	3392	2636
4858	4712	2587
3458	3489	2142
2885	2575	1235
2412	2324	1347
3412	3727	2452
3110	2914	2245
3242	3115	2754
2142	2787	2241
3966	3144	3744
4457	3789	3747

TABLE IX
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 1 ($limit_{max}(N) = (2/5)Nr_v$).

$N = 22$, $limit = limit_{SF(1-2)}(6)$		
$SF1$	$SF2$	$SF3$
68	549	523
74	549	523
68	550	525
81	546	519
547	547	522
544	551	524
75	546	521
541	546	522
541	545	519
547	549	522
550	550	524
546	546	520
552	552	526
550	550	524
75	550	523
1074	555	648
542	550	524
74	538	512
69	551	525
69	551	524
575	548	521
549	551	524
74	545	519
538	546	519
68	551	525
540	549	755
73	551	525
596	550	525
555	549	654
562	549	524
68	548	524
541	552	525
539	544	519
545	551	525
45	551	525
548	548	522
544	551	524
542	546	519
46	547	522
540	546	520
547	523	510
75	520	503
506	511	498
72	563	525
569	521	517
69	532	536
514	501	503
76	545	586
568	521	501
542	564	521

TABLE X
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 1 ($limit_{max}(N) = (2/5)Nr_v$).

$N = 22$, $limit = limit_{SF(1-2)}(22)$		
$SF1$	$SF2$	$SF3$
3742	3252	2993
4293	3887	3523
3768	3834	2545
4842	4001	2368
3824	3311	2524
4889	3995	2796
4361	3861	2784
3761	3408	2165
4828	3853	2679
4826	3994	2672
Inf	4036	3010
4292	3827	3191
4815	3520	2102
4354	3789	2860
4788	3433	2118
4689	3945	4160
3761	3686	2522
4309	4002	2638
3244	3986	3072
Inf	4811	2214
3742	3263	2979
4241	3887	3553
3752	3823	2565
4878	4035	2343
3863	3368	2523
4833	3936	2722
4374	3868	2742
3789	3424	2165
4873	3863	2616
4812	3942	2656
Inf	4076	3098
4296	3863	3134
4879	3578	2167
4323	3724	2825
4712	3413	2187
4692	3965	4174
3775	3613	2553
4345	4065	2635
3263	3968	3014
Inf	4845	2263
3778	3267	2913
4248	3853	3554
3753	3876	2532
4881	4022	2432
4371	3765	2854
4763	3445	2157
4645	3967	4183
3768	3626	2141
4363	4024	2653
3298	3987	3092

TABLE XI
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 1 ($limit_{max}(N) = (2/5)Nr_v$).

$N = 6$, $limit = limit_{SF(1-2)}(6)$		
$SF1$	$SF2$	$SF3$
162	167	223
359	322	307
324	310	449
603	603	653
1170	916	1146
159	162	261
158	157	202
180	153	239
474	154	322
184	157	308
300	300	437
323	160	346
583	460	572
465	447	740
160	160	333
150	150	356
335	399	333
887	543	340
149	294	370
439	402	253
471	157	217
165	161	273
157	157	390
153	155	257
175	162	237
161	160	235
313	318	332
344	314	259
148	151	238
471	457	581
603	592	760
156	162	375
185	163	317
164	168	248
714	463	694
157	157	317
159	154	233
325	329	376
152	154	250
616	601	830
114	146	225
147	187	246
458	147	328
271	158	258
167	114	288
368	214	341
175	174	279
557	445	625
128	156	399
158	198	225

TABLE XII
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 2 ($limit_{max}(N) = (N-2)r_s$).

$N = 6$, $limit = limit_{SF3}(6)$		
$SF1$	$SF2$	$SF3$
1476	1441	614
758	746	562
1983	1958	1928
1507	1408	1076
1243	918	539
1264	958	990
1234	999	672
2013	1975	757
1226	1035	836
2247	2411	2472
553	728	821
1274	1287	995
1215	597	940
1218	1137	603
1748	1588	1071
1958	1922	1610
1259	914	618
1481	1397	528
1007	929	909
1267	1481	1533
1980	Inf	729
1494	1040	997
1767	1513	1918
1259	1239	888
1505	1465	779
770	921	728
774	760	626
1738	1601	752
1514	1436	706
2000	1362	1842
1777	1928	1244
532	640	792
1727	1731	1178
2001	1192	772
1715	1745	1404
521	739	812
983	975	970
1471	1472	1425
1680	1145	586
1468	1249	1001
1474	1859	1252
1752	1141	1744
2281	1282	788
1588	1855	1444
1948	1788	1298
1928	1985	1758
1714	1314	1018
2080	1252	992
1858	1171	971
1744	1555	1588

TABLE XIII
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 2 ($limit_{max}(N) = (N-2)r_s$).

$N = 10$, $limit = limit_{SF(1-2)}(10)$		
$SF1$	$SF2$	$SF3$
556	506	743
990	738	564
1488	1257	1016
1516	490	494
990	745	822
1267	1012	1221
1727	1691	609
1724	643	816
990	961	782
978	755	864
1044	653	727
516	491	699
1286	1371	787
501	501	553
742	588	643
773	718	719
2018	1822	761
1469	1165	1445
1236	1208	831
991	948	667
789	901	662
737	693	789
982	765	1228
1201	539	693
988	978	949
540	493	554
1000	886	931
1700	1547	1375
1250	786	986
979	668	686
1759	1356	780
954	896	573
1759	1510	1501
1222	998	656
1470	1485	1438
1451	1233	551
1246	1176	1260
726	730	794
996	808	589
1007	500	485
745	558	696
1025	858	1073
1592	1486	625
1578	596	753
868	669	1078
1401	856	775
896	927	958
1576	1145	825
1276	1245	1287
1295	1047	586

TABLE XIV
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 2 ($limit_{max}(N) = (N-2)r_s$).

$N = 10$, $limit = limit_{SF3}(10)$		
$SF1$	$SF2$	$SF3$
2962	3204	2450
Inf	4505	3057
4917	4299	1709
Inf	Inf	2998
Inf	Inf	2737
4512	4255	1956
Inf	Inf	2050
4348	3608	2617
Inf	Inf	3534
Inf	4152	2852
4944	4458	2042
Inf	Inf	3166
Inf	Inf	2611
4434	3562	2263
Inf	Inf	3590
4425	4215	2482
4904	3956	2067
4883	4472	2814
Inf	4914	2123
4481	4258	2253
Inf	Inf	4466
Inf	Inf	2710
Inf	4874	3087
Inf	Inf	3193
4933	4153	2227
4356	3093	2011
Inf	Inf	3914
4559	4354	2135
4947	4630	3746
Inf	3807	2256
Inf	Inf	2708
3898	3311	3422
4424	3471	1890
Inf	Inf	4372
Inf	Inf	3424
4956	4871	3101
3933	4025	2027
Inf	Inf	4758
Inf	Inf	2700
Inf	4572	2729
3546	3178	2154
Inf	Inf	4759
4658	3974	2945
Inf	Inf	4567
Inf	3988	3015
4512	4002	3157
Inf	Inf	4932
4581	4175	2986
Inf	Inf	3987
Inf	4567	3621

TABLE XV
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 2 ($limit_{max}(N) = (N-2)r_s$).

$N = 8$			
$limit = limit_{SF(1-2)}(6)$		$limit = limit_{SF(1-2)}(8)$	
$SF1$	$SF2$	$SF1$	$SF2$
211	320	806	609
203	338	1042	1246
208	169	597	845
379	560	801	1046
603	643	801	1078
196	329	609	917
206	481	798	1509
407	562	788	969
405	427	1403	1320
201	309	819	359
207	204	398	264
200	196	621	1223
209	327	984	197
613	962	1051	708
231	298	806	945
404	678	791	196
244	322	650	929
209	212	849	1109
236	185	401	349
201	306	824	168
406	483	474	525
208	305	1243	499
197	298	1023	886
209	553	1023	1159
212	197	448	1314
601	842	1030	988
205	176	278	173
1269	1589	403	690
403	368	590	609
210	175	211	197
201	326	599	1277
209	318	245	242
80	247	412	385
785	1021	596	896
194	208	401	410
205	300	395	515
204	773	221	338
214	323	399	331
197	175	787	1101
201	202	1191	987
245	475	508	804
145	389	886	1563
578	693	623	963
325	571	1002	996
392	458	1145	503
139	203	859	889
178	300	1236	1385
369	785	241	539
124	261	496	571
298	268	267	543

TABLE XVI
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 3: INTENSITÁ DEI SEGNALI RELATIVI ALLE UNITÁ ROBOTICHE ANISOTROPA.

$N = 14$			
$limit = limit_{SF(1-2)}(8)$		$limit = limit_{SF(1-2)}(14)$	
$SF1$	$SF2$	$SF1$	$SF2$
358	342	1061	821
351	352	2114	2205
50	451	1043	1256
360	312	1704	1691
347	343	1708	1044
348	342	1077	637
353	343	2073	1829
339	343	1719	1448
354	344	1725	2235
357	317	1385	2089
362	392	1761	949
349	349	1738	991
78	352	1707	1666
350	444	1082	1241
355	361	1426	956
352	347	2107	2332
352	466	1758	1605
354	351	1707	991
347	383	1716	1940
346	350	1394	1840
355	463	2444	2926
347	350	2409	3385
342	339	1719	1191
344	348	1033	1382
76	347	1451	1534
77	320	2051	1485
346	353	1745	1854
347	344	1769	1293
352	343	1394	1504
345	342	2035	1922
61	330	2718	1568
354	343	2055	1859
346	355	1726	1347
352	320	1700	1444
350	344	2379	1752
351	393	1423	2361
341	342	1352	1229
357	345	2350	1694
367	490	2456	2221
71	357	1720	1745
348	390	1502	889
347	375	1756	1402
324	333	2405	2278
355	331	1542	1654
347	326	1678	1167
374	401	1967	2123
342	456	1884	1546
332	347	2154	2031
78	312	2895	1643
364	399	1846	1667

TABLE XVII
RISULTATI DELLE SIMULAZIONI RELATIVE AL CASO 3: INTENSITÁ DEI SEGNALI RELATIVI ALLE UNITÁ ROBOTICHE ANISOTROPA.