Path planning locale per robot mobili basato su potenziali artificiali alternati

Contents

1 Introduzione

È innegabile che in questi anni si stia assistendo ad un aumento vertiginoso di sviluppo ed uso della robotica. È importante però evidenziare una distinzione tra due concetti apparentemente simili, ma per certi versi opposti, che caratterizzano due macro-categorie della robotica: automazione e autonomia. Il primo riguarda quei robot, tipicamente industriali, che operano in ambienti noti a priori ed eseguono in loop un compito predefinito; automatizzare, perciò, vuol dire sostituire l'essere umano in compiti ripetitivi e solitamente privi di eventi inaspettati. L'autonomia, invece, ben più complessa da realizzare, é caratteristica di quei sistemi che hanno un certo grado di inconsapevolezza sul proprio futuro e l'ambiente circostante. Il robot, quindi, é definito autonomo se é un agente intelligente situato nello spazio fisico, dove un agente intelligente si definisce come un'entità che **osserva** l'ambiente e prende delle **azioni** per massimizzare il raggiungimento del suo **obiettivo**[4]. Nel caso specifico di questa tesi, l'ambiente del robot autonomo é lo spazio bidimensionale (una superficie), e il suo obiettivo é un punto in questo spazio. Massimizzare il raggiungimento di questo punto vuol dire arrivarci nel minor tempo possibile, senza collidere con eventuali ostacoli. Quindi, il robot autonomo deve compiere una serie di azioni, non note a priori e definite da un algoritmo di pianificazione che si basa sui dati osservati dai sensori, per spostare la sua traiettoria, al fine di evitare collisioni e raggiungere comunque l'obiettivo. La tipica architettura di navigazione di un robot autonomo é data perciò da quattro moduli, detti anche primitive:

Percezione Prende in input le informazioni derivanti dai sensori, le processa e le restituisce

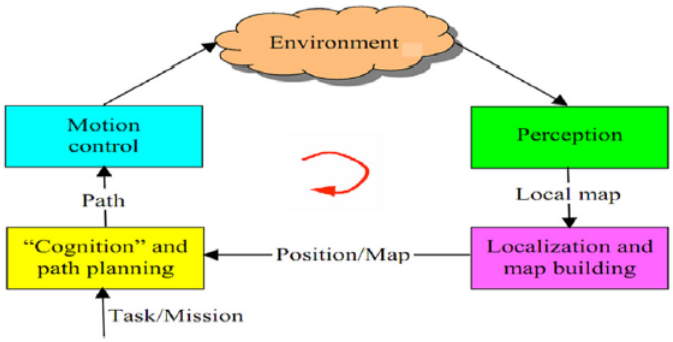
Localizzazione e Mapping Con le informazioni sensoriali, il robot costruisce una rappresentazione del proprio intorno basandosi sulla propria posizione e ciò che osserva. Il risultato globale, dopo aver fatto varie osservazioni di intorni diversi, sarà una mappa dell'ambiente, rispetto alla quale il robot può localizzarsi. (Per scopi esemplificativi, questo modulo verrà tralasciato nell'algoritmo di questa tesi, e si userà descrivere la posizione del robot con coordinate assolute e non rispetto ad una mappa.)

Pianificazione In base alle informazioni sensoriali e cognitive in possesso, produce in output delle decisioni ad un livello di astrazione alto. Nel caso del motion planning, la direttiva da produrre é il percorso da seguire.

Azione Prende in input le direttive del modulo di pianificazione e produce dei comandi a basso livello per gli attuatori del robot.

L'architettura utilizzata in questa tesi é la cosiddetta gerarchica: le quattro primitive vengono eseguite in ordine e in loop. È particolarmente indicata per problemi in cui l'obiettivo finale é ben definito a priori. In altre parole, non vi é nessun meccanismo di apprendimento nel robot, ma semplicemente pianificazione deterministica orientata al goal. In figura é visivamente sintetizzato quanto appena detto.

Figure 1: Architettura di navigazione[2]



In questa tesi viene affrontato un problema che rientra nel terzo modulo: il path planning, un problema di grande importanza e argomento di molta ricerca.

1.1 Path planning

Una sua rapida formulazione potrebbe essere la seguente: data la posizione iniziale (del robot) A e una posizione finale B, imposta da chi fa uso del robot, il path planning consiste nel calcolare un percorso fisicamente realizzabile e ottimale per arrivare da A a B. All'interno dei metodi esistenti (e non), ci sono due importanti distinzioni da fare: sulla formulazione del problema e sulla soluzione al problema.

La prima é tra online e offline path planning, o anche locale e globale. Il path planning globale riguarda quelle situazioni in cui l'ambiente considerato é interamente noto a priori, per cui é possibile calcolare il percorso da seguire ancor prima che il robot inizi a muoversi; quello locale é inerente ai casi in cui il robot debba fare i conti lungo il suo percorso con eventi inaspettati, quali ostacoli dinamici,per cui é necessario reagire localmente, aggiornando ripetutamente le informazioni derivanti dai sensori e aggiustando la traiettoria al fine di evitare l'ostacolo e poter raggiungere in tempi ottimali l'obiettivo.

Chiaramente, la maggior parte dei problemi di robotica autonome deve fare i conti con una situazione del secondo tipo.

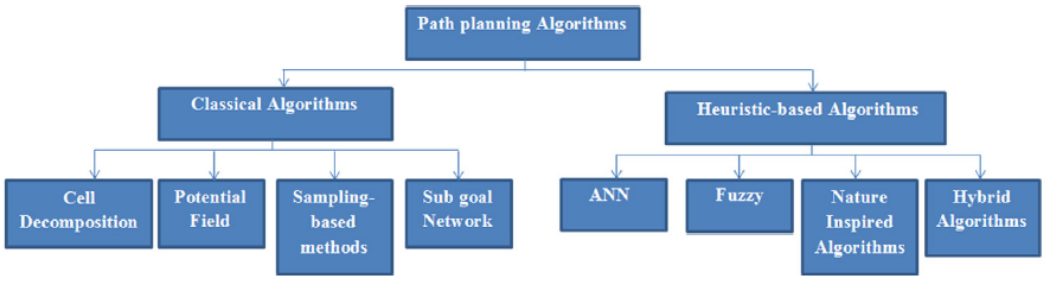
La seconda distinzione é tra soluzioni basate su tecniche di intelligenza artificiale - la cui trattazione esula dagli scopi di questa tesi - e soluzioni classiche. Queste ultime possono ulteriormente essere suddivise in [5]:

- Subgoal (o anche roadmap)
- Decomposizione in celle
- Sampling based
- Potenziali artificiali

Nota : ancora da fare breve descrizione dei metodi

La totalità dei metodi appena descritti risolvono adeguatamente il problema del path planning globale[1], ma risultano inefficienti nel caso del path planning locale. Il metodo più versatile e anche più longevo nell'utilizzo é sicuramente quello dei potenziali artificiali.

Figure 2: Classificazione algoritmi di path planning[2]



Come da titolo, questa tesi ha come obiettivo specifico quello di esporre un lavoro di progettazione, implementazione e simulazione di un algoritmo di path planning **locale** basato su **potenziali artificiali alternati**.

1.2 Formalizzazione del problema

Chiamerò $r(t) = [x_r(t), y_r(t), \theta_r(t)]^T$ la posizione del robot nell'istante t e $O_i(t) = [x_{O,i}(t), y_{O,i}(t)]^T, i = 1...N$ la posizione degli N ostacoli che, per scopi esemplificativi in fase di prototipazione dell'algoritmo, saranno di forma circolare e con raggio R_i. Quest'ultima ipotesi non provoca una perdita di generalità, visto che per un ostacolo di forma generica si può considerare la sua circonferenza circoscritta. Esisterà inoltre un punto $G = [G_x, G_y]$ che indica l'obiettivo del robot. Il problema consiste nel

voler raggiungere il punto G dalla posizione iniziale $r(0)$, tenendo conto degli N ostacoli in movimento. Il robot é dotato di un raggio di visione di R_v metri, entro il quale é capace di rilevare un ostacolo. Inoltre, si presuppone che valga la seguente condizione

$$\left\| \begin{bmatrix} x_r(t) \\ y_r(t) \end{bmatrix} - O_j(t) \right\| \leq R_v$$

, ovvero $R_i \leq R_v, \forall i$. Ciò vuol dire che nel momento in cui il robot incontra un ostacolo, il centro di quest'ultimo é incluso in R_v.

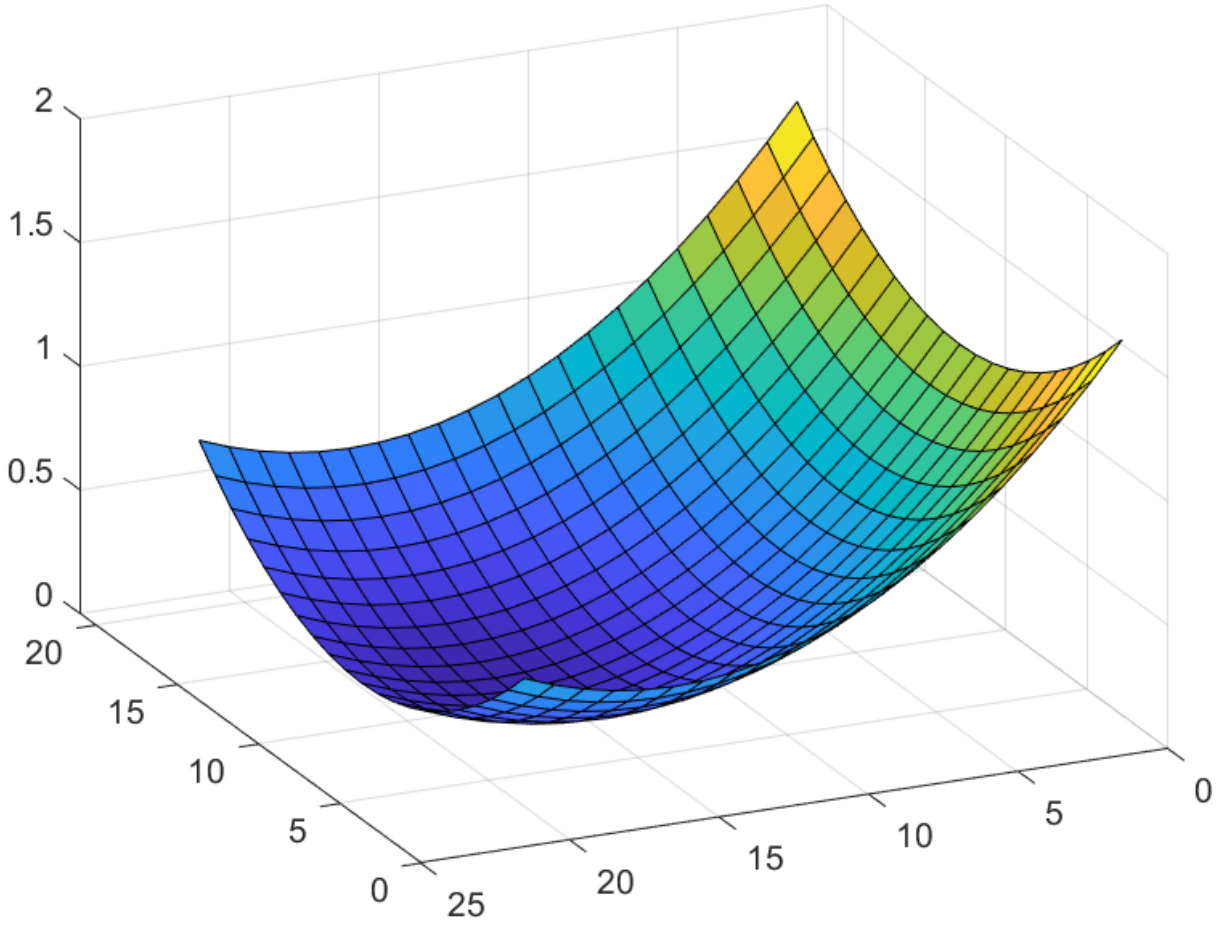
2 Potenziali artificiali

2.1 Metodo classico

Tradizionalmente, nel path planning basato su potenziali artificiali il robot viene fatto muovere mediante una funzione continua in due variabili, ovvero un potenziale scalare, che nasce dalla somma di due potenziali: attrattivo e repulsivo. Questi due potenziali sono chiamati artificiali perché generano (per l'appunto artificialmente) una forza che guida il robot in ogni sua configurazione $r(t)$. Nello specifico, la forza generata dal potenziale é il suo antigradiente, ovvero il gradiente cambiato di segno, che indica al robot la direzione di moto localmente più promettente[3], cioè verso il punto di minimo della funzione. Di conseguenza il potenziale attrattivo assume una forma tale da avere un unico punto di minimo posizionato proprio nel punto di arrivo, mentre il repulsivo ha un unico punto di massimo corrispondente alla posizione dell'ostacolo.

Il potenziale attrattivo ha di solito la forma in figura 3

Figure 3: Potenziale attrattivo

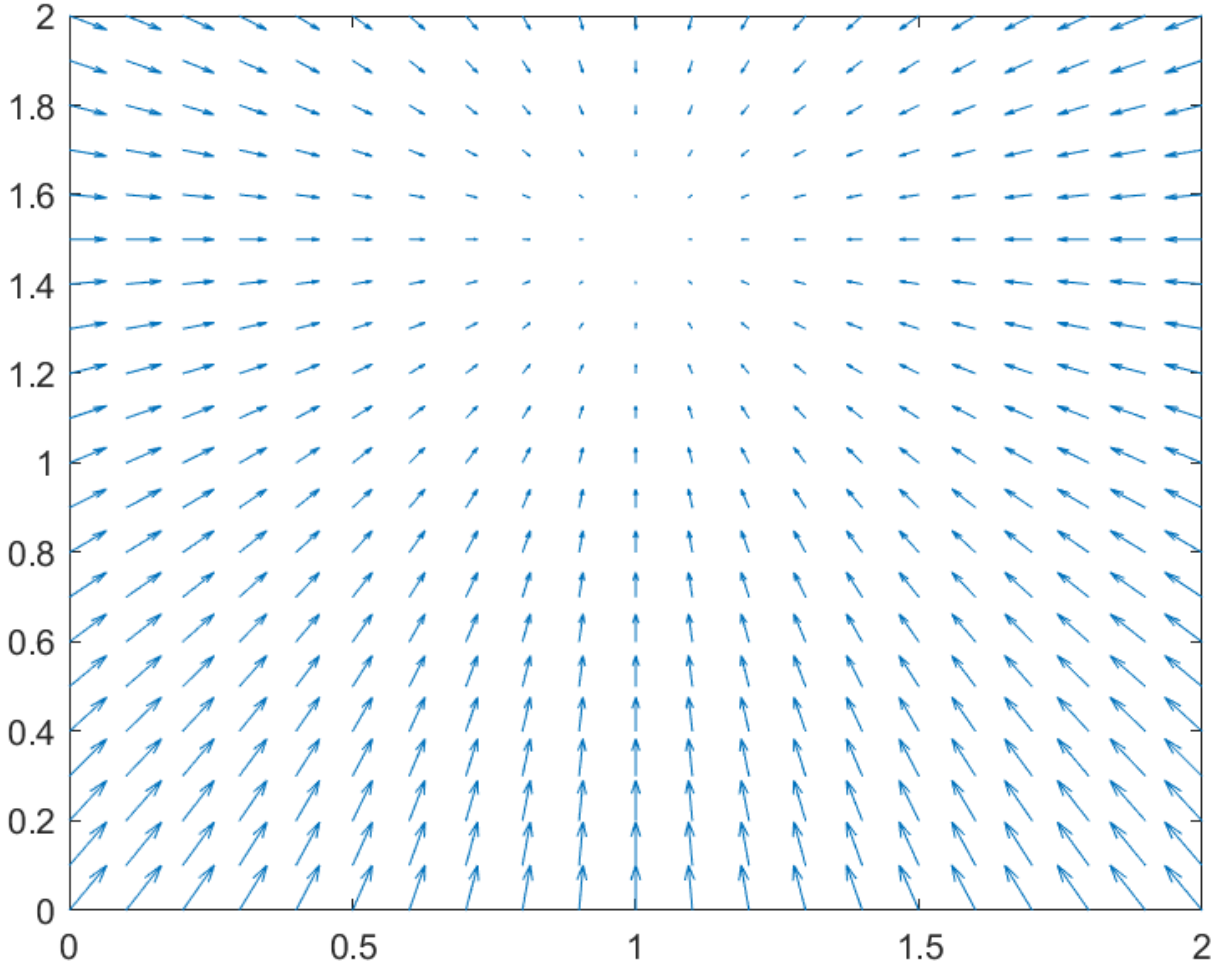


a cui corrisponde la funzione

$$U_a(r(t), G) = \frac{1}{2} \left\| G - \begin{bmatrix} x_r(t) \\ y_r(t) \end{bmatrix} \right\|^2 \tag{1}$$

Il suo antigradiente di conseguenza é formato da tanti vettori che, con un'intensità proporzionale alla distanza dal goal, puntano verso quest'ultimo, in figura 4 con coordinate $[1, 1.5]$

Figure 4: Antigradiente del potenziale attrattivo



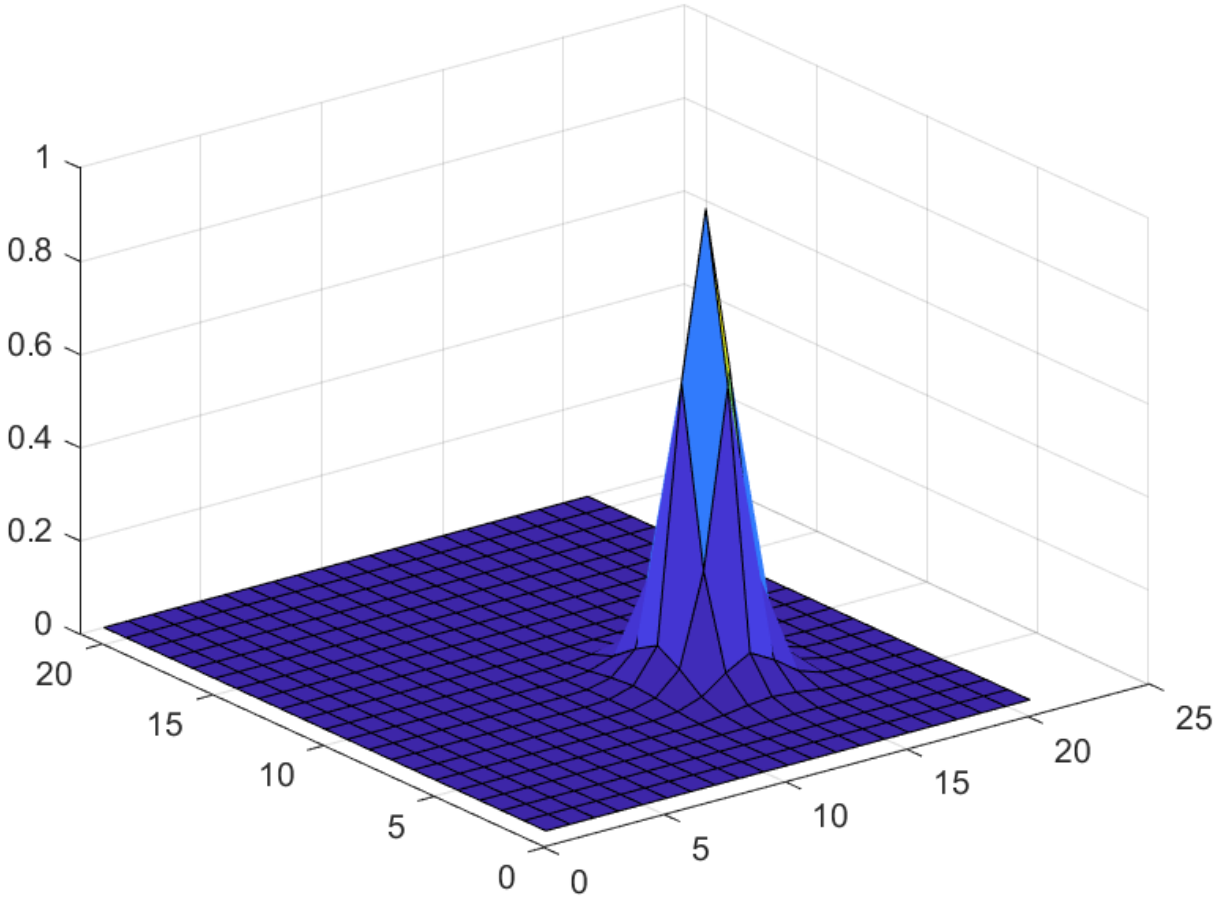
e matematicamente si esprime come il vettore delle derivate parziali (del potenziale) cambiato di segno, ovvero

$$-\nabla U_a(r(t), G) = \left\| G - \begin{bmatrix} x_r(t) \\ y_r(t) \end{bmatrix} \right\| \tag{2}$$

Perciò, la forza esercitata dal potenziale sul robot punta verso il goal e converge a zero quando la configurazione $r(t)$ tende alla destinazione G, esprimendo difatti un errore lineare tra goal e posizione del robot.

Il potenziale repulsivo ha una forma duale a quello attrattivo, come in figura 5

Figure 5: Potenziale repulsivo



a cui corrisponde la seguente funzione a tratti

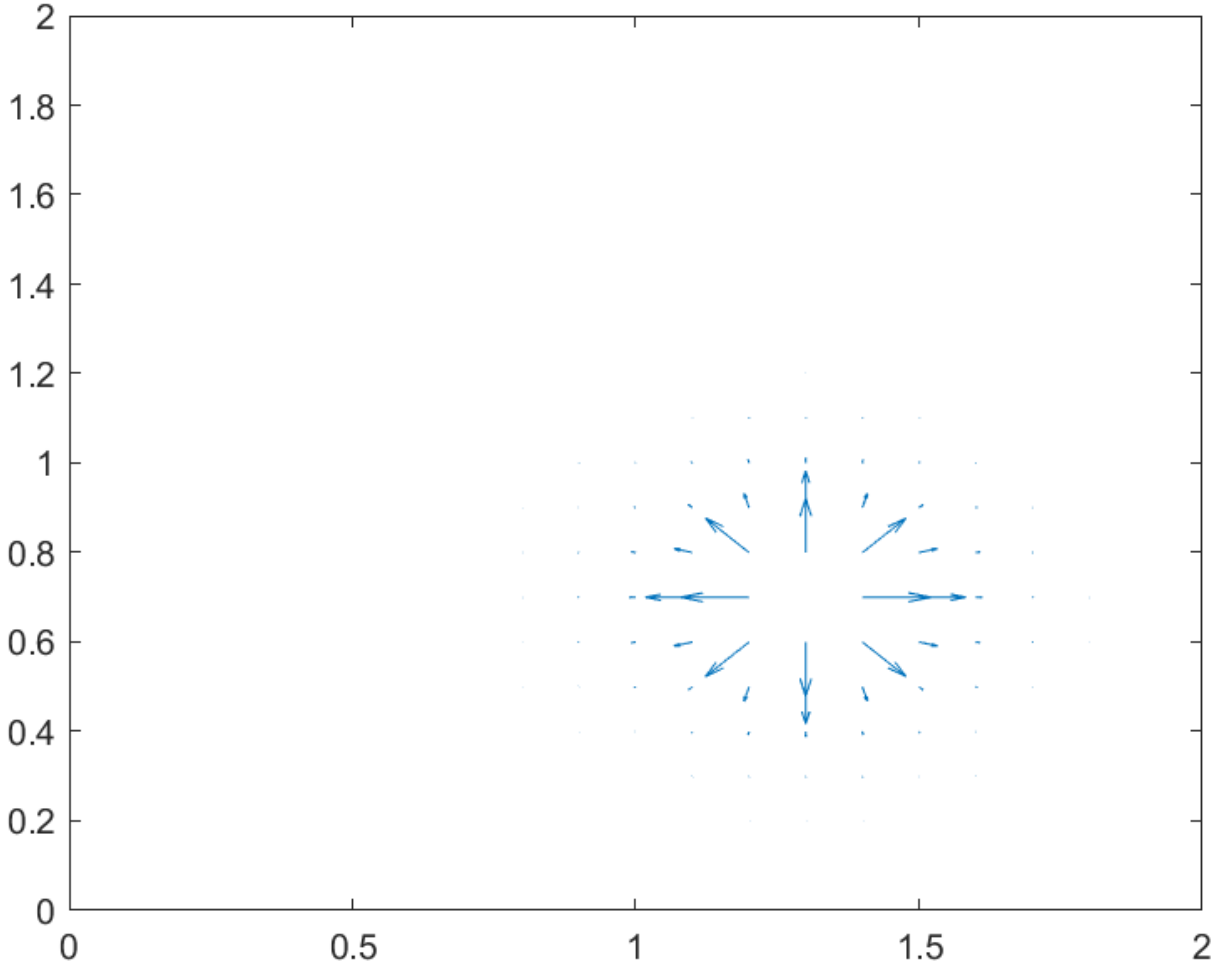
$$U_r(r(t), O_j(t)) = \begin{cases} \frac{1}{2}\left(\frac{1}{d(r(t),O_j(t)} - \frac{1}{\eta}\right) & d(r(t), O_j(t)) \leq \eta \\ 0 & altrimenti \end{cases} \quad (3)$$

dove

$$d(r(t), O_j(t) = \left\| O_j(t) - \begin{bmatrix} x_r(t) \\ y_r(t) \end{bmatrix} \right\|$$

Il compito della forza generata dal potenziale repulsivo é quello di spingere via il robot dalla posizione dell'ostacolo. In figura 6 si vedono le linee di campo dell'antigradiente che puntono verso fuori rispetto alla posizione dell'ostacolo, qui con coordinate $[1.5, 1]$.

Figure 6: Antigradiente del potenziale repulsivo



Una volta calcolato il potenziale repulsivo per ogni singolo ostacolo (in questo caso uno solo), si ottiene il potenziale totale:

$$U(t) = U_a(r(t), G) + \sum_{j=1}^{n} U_r(r(t), O_j(t)) \tag{4}$$

Il robot in ogni sua configurazione $r(t)$, seguendo l'antigradiente di questo potenziale, viene guidato verso il goal e **contemporaneamente** allontanato dagli ostacoli. Chiaramente, il metodo é idoneo sia al path planning globale che a quello locale (basta ricalcolare il potenziale totale in presenza di ostacoli).

Figure 7: Potenziale totale

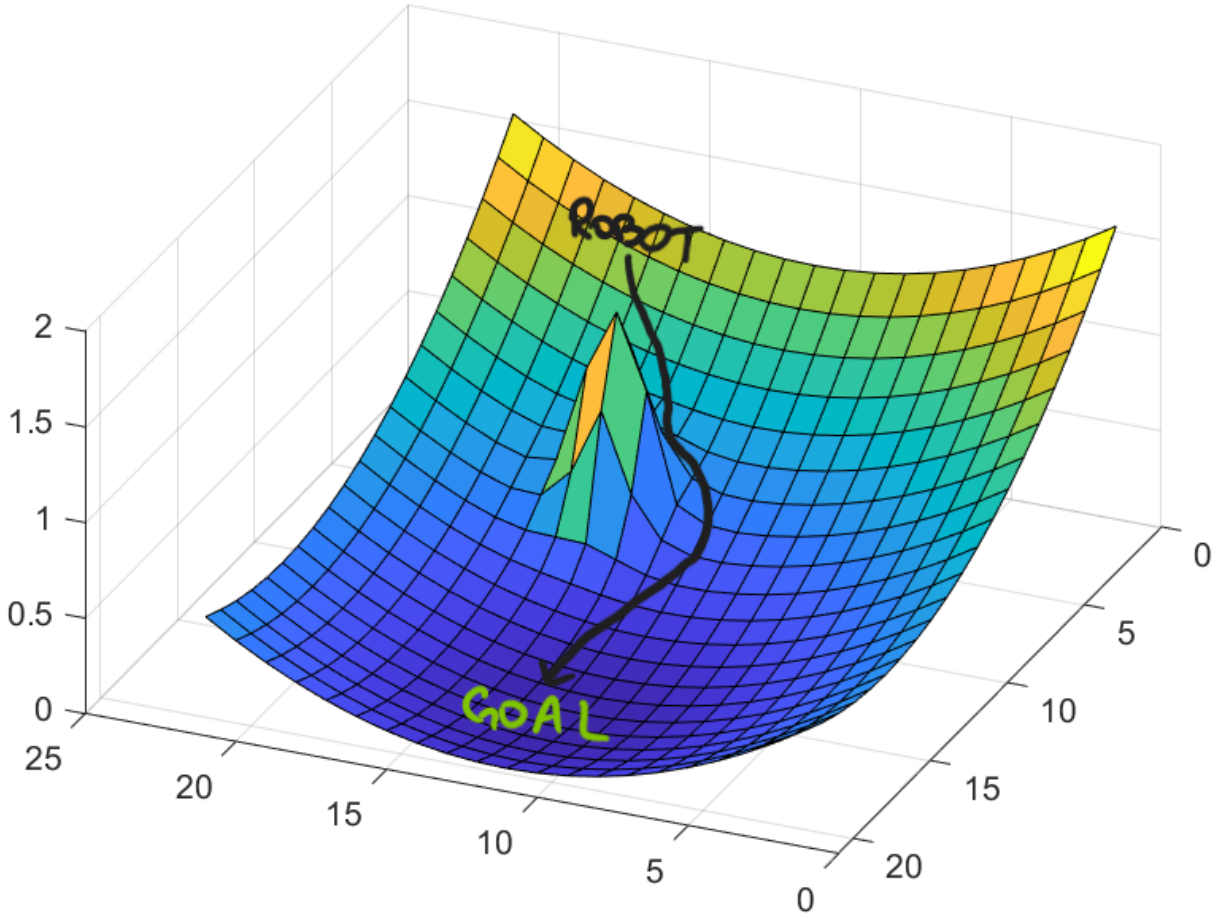


Figure 8: Antigradiente del potenziale totale

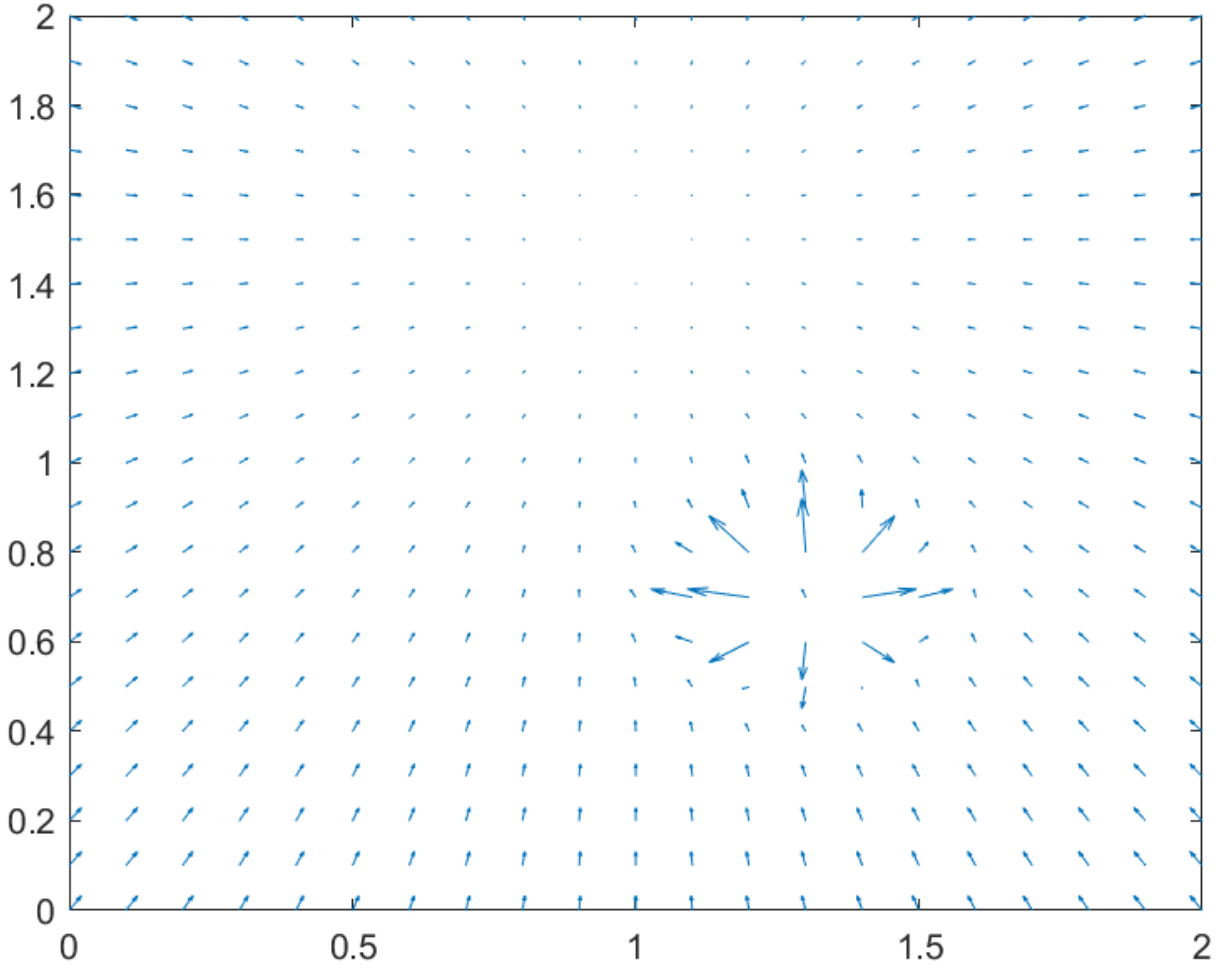


Figure 9: Potenziale con minimo locale

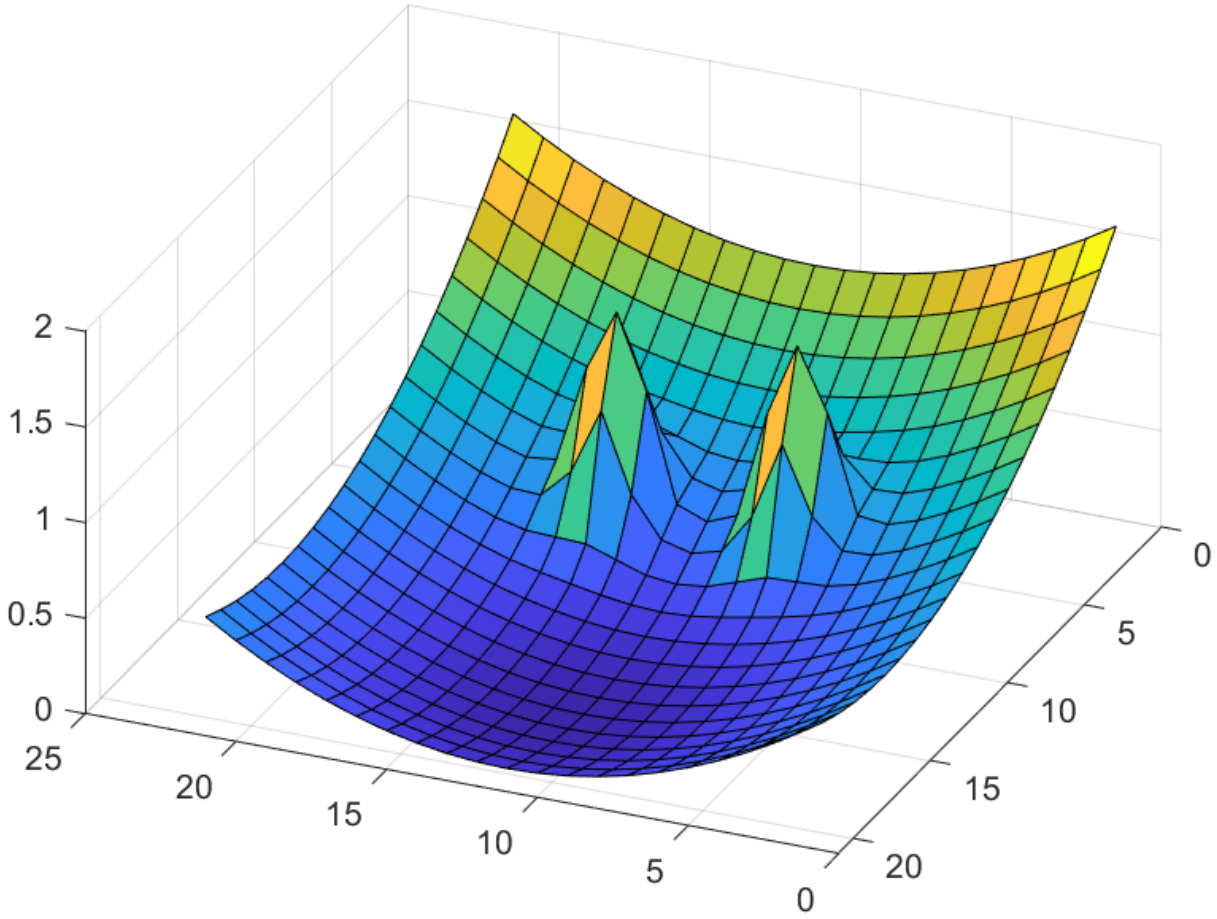


Figure 10: Antigradiente del potenziale con minimo locale

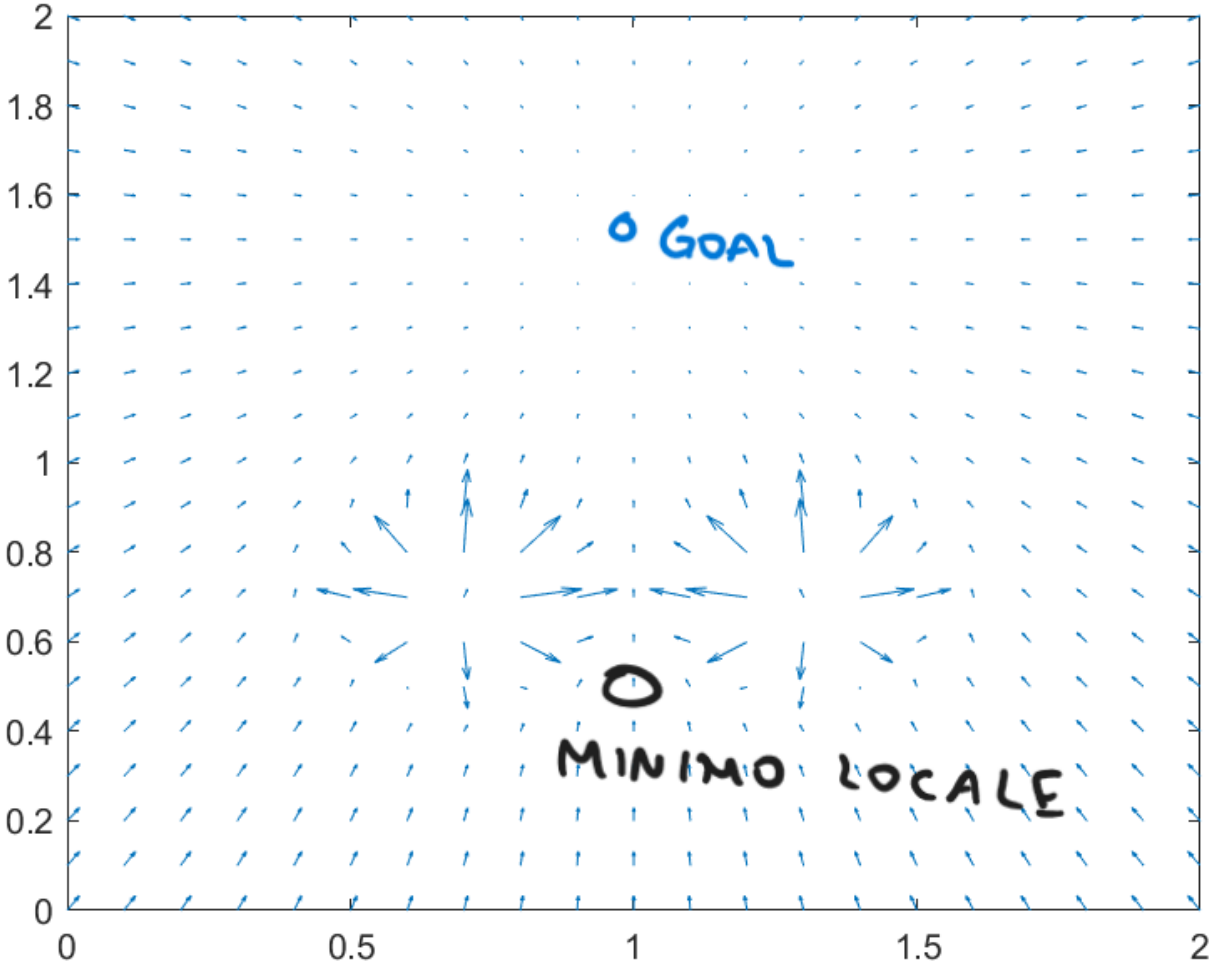


Illustrazione potenziale bypassante (potenziale e antigradiente)

3 Algoritmo di navigazione[1]

La strategia é semplice: nella posizione iniziale, il robot sonda l'ambiente attorno a sé. Se non vi sono presenti ostacoli a impedirne l'avanzamento verso il goal, la traiettoria da seguire é quella imposta dal potenziale attrattivo; altrimenti, é necessario "switchare" dal potenziale attrattivo a quello vorticoso, al fine di aggirare l'ostacolo, per poi tornare a seguire la traiettoria. Perciò, la peculiarità di questo algoritmo é che in ogni istante il robot seguirà un solo potenziale alla volta, evitando così il problema dei minimi locali. Inoltre, le informazioni necessarie a pianificare lo switch non richiedono informazioni globali, ma solo quelle riguardanti l'ostacolo da aggirare.

3.1 Interazione con l'ambiente

Come viene costruito l'ambiente (grid)

3.1.1 Percezione

Come vengono percepiti gli ostacoli e le loro velocità
Come viene percepita la propria posizione (accenno localizzazione)

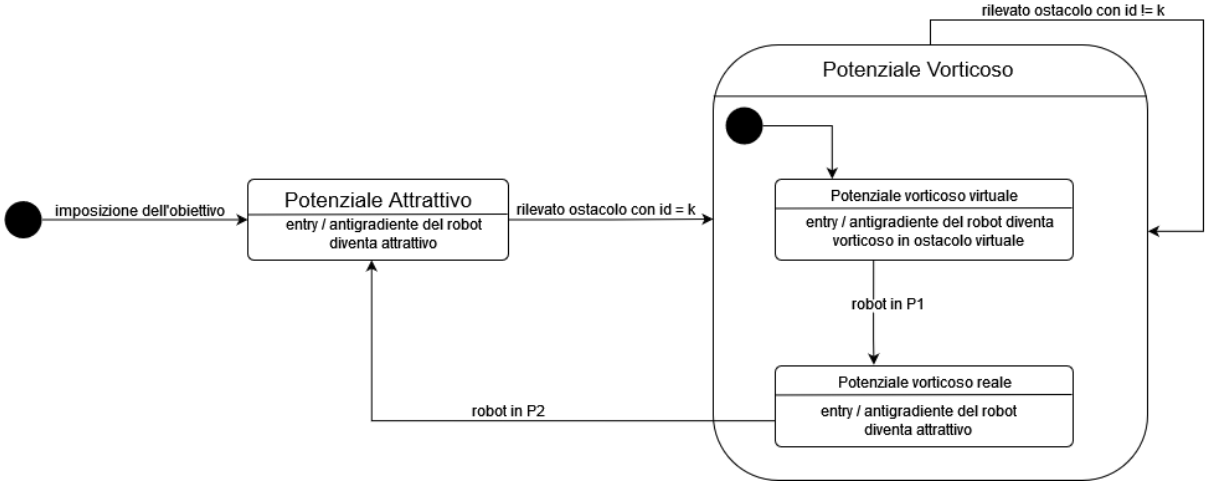
3.1.2 Azione

Modello cinematico
Legge di controllo
Come faccio muovere il robot nella grid

3.2 Pianificazione

Data la struttura intrinseca della strategia di navigazione che si basa su un meccanismo di switching tra due stati, per definizione mutuamente esclusivi, un modo per modellare visivamente l'algoritmo é lo statechart diagram in figura. Il robot può trovarsi o nello stato corrispondente al potenziale attrattivo, oppure in quello corrispondente al potenziale vorticoso. In quest'ultimo esistono altri due sottostati, la cui natura verrà spiegata meglio in seguito.

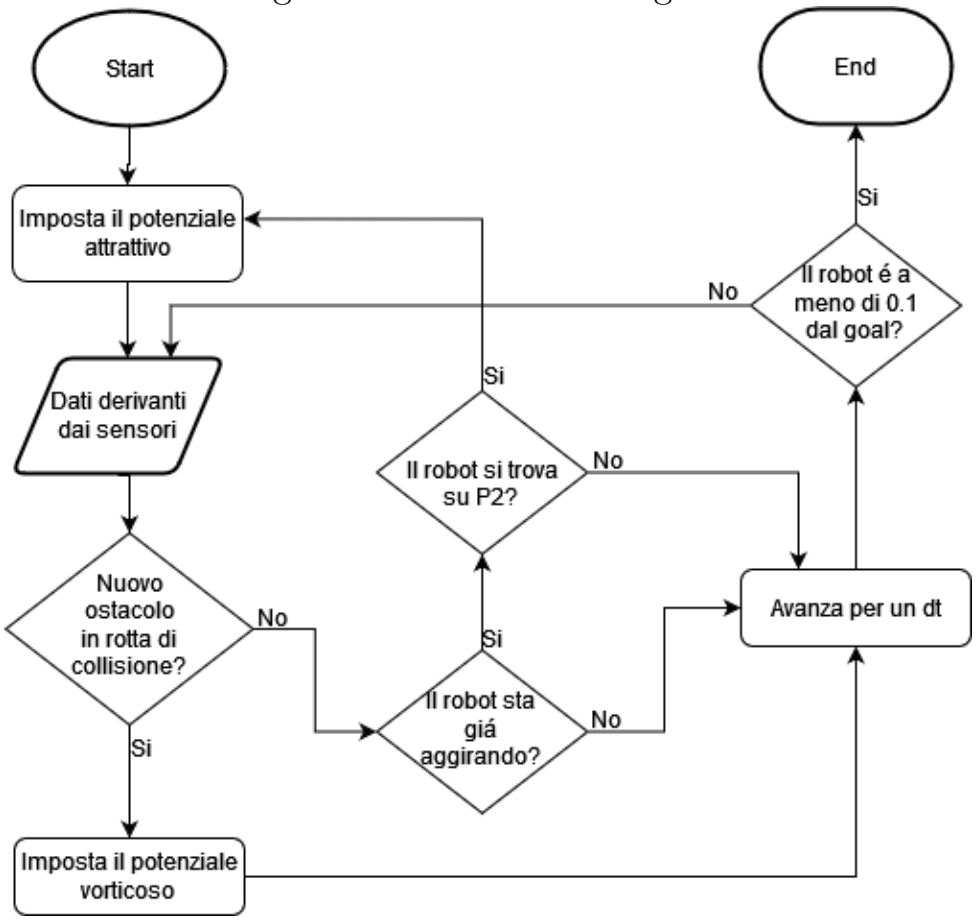
Figure 11: Statechart Diagram



Note : caso 1 sto seguendo il potenziale attrattivo e passo al vorticoso (calcolo verso, calcolo xOmega e yOmega, calcolo P1 e P2, calcolo

due potenziali con relativi c), caso 2 sto seguendo il vorticoso e passo all'attrattivo (reimposto attrattivo)

Figure 12: Flowchart Diagram



4 Simulazione sul Turtlebot

5 Testing e risultati

Parametri utilizzati
Un ostacolo fermo e in movimento
Minimi locali e confronto con potenziali artificiali classici
Tre in movimento

6 Applicazioni e sviluppi futuri

List of Figures

References

[1] Luigi D'Alfonso et al. "Obstacles avoidance based on switching potential functions". In: *Journal of Intelligent & Robotic Systems* ().

[2] Thi Thoa Mac et al. "Heuristic approaches in robot path planning: A survey". In: *Robotics and Autonomous Systems* ().

[3] Claudio Medio and Giuseppe Oriolo. "Robot Obstacle Avoidance Using Vortex Fields". In: *Advances in Robot Kinematics*. 1991.

[4] Robin Murphy. *Introduction to AI Robotics*.

[5] Bruno Siciliano et al. *Robotica. Modellistica, pianificazione e controllo*.