UNIVERSITÀ DEGLI STUDI DI PALERMO

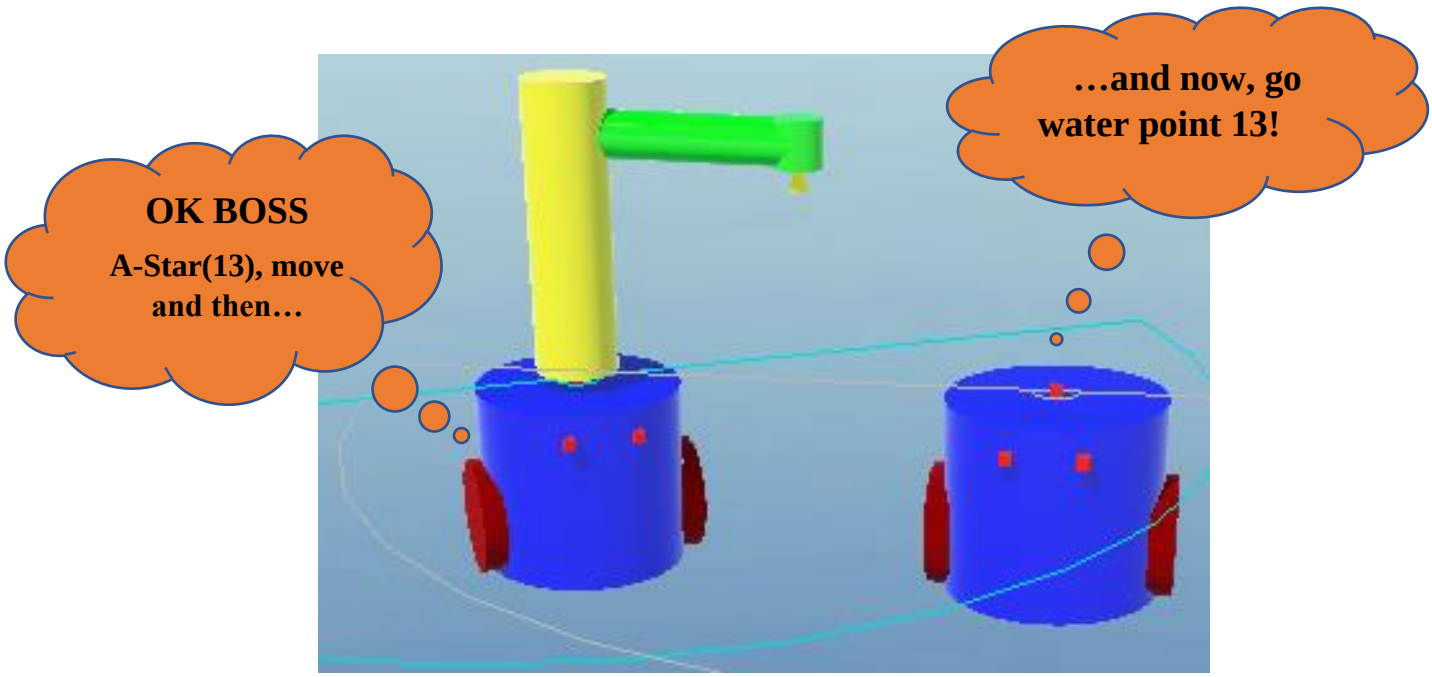
Dipartimento di Ingegneria

Corso di Laurea Magistrale in Ingegneria Informatica

ROBOTICA

Κήπουρομπότ

Kípourompót



ALLIEVI INGEGNERI:

Mario Caruso
Vincenzo Guglielmo La Mantia

DOCENTE:

Prof. Ing. Antonio Chella

ANNO ACCADEMICO
2021 - 2022

Sommario

1. Introduzione

Sempre più spesso la robotica offre spunti preziosi e significativi nell'industria 4.0. Anche in ambito agricolo, sempre più spesso, si sente parlare del suffisso indicativo di alta tecnologia 4.0, ossia la rivoluzione che inizia ad abbraccia tutte le fasi fondamentali: dalla semina, alla rimozione degli infestanti e alla raccolta dei frutti.

1.1 Agricoltura 4.0 e 5.0, verso l'agricoltura di precisione

L'agricoltura incide significativamente nella produzione economica mondiale, tanto che in parecchi paesi del mondo è il settore dominante dell'economia [1]. Per chi non vive il mondo dell'agricoltura può sembrare una realtà lontana dalla tecnologia, ma in verità, le aziende agricole, sempre più spesso, cercano nuovi approcci per migliorare la resa dei raccolti.

Un rapporto di The Robot Report, che valuta l'andamento dell'impiego dei robot in agricoltura asserisce che questo aumenterà significativamente nei prossimi anni, infatti passerà dalle 32.000 unità del 2016 a 594.000 unità annue nel 2024, arrivando a coinvolgere fatturato annuo di decine di miliardi di dollari.

Il crescente interesse sui robot agricoli è dato da diversi fattori trasversali a componenti economiche sociali e ambientali:

- preoccupazioni per la sicurezza alimentare a causa di eventi Socio-Politici
- la crescita della popolazione globale
- la diminuzione della disponibilità dei lavoratori agricoli
- i costi e le complessità del lavoro agricolo,
- il cambiamento dei terreni agricoli e quello climatico
- la crescita dell'agricoltura indoor
- l'automazione del settore agricolo.

Con questi obiettivi, si è dato vita all' agricoltura di precisione, ossia un sistema di gestione della produzione agricola che utilizza strumenti e tecnologie per fare la cosa giusta, nel posto giusto, al momento giusto [2]; con queste idee nasce l'ambizione di una industria europea sostenibile, humancentric e resiliente, dove le tecnologie 4.0 e 5.0 rappresentano un fattore chiave per lo sviluppo e la trasformazione dell'agricoltura

Questo fenomeno economico ambientale sociale è di scala planetaria e per questo riceve attenzione a livello globale.

2. Stato dell'arte e sviluppi nel mondo

2.1 Robot in agricoltura, i paesi leader

Il Giappone è al primo posto per quanto concerne ricerca e sviluppo di robot agricoli. Entrati nel XXI secolo, si sono diffusi velocemente e sempre più spesso sostituiranno più le attività agricole manuali, per produrre il 70% di cibo in più a parità di risorse (terra e acqua).

Negli Stati Uniti, l'azienda Abundant Robotics, ha sviluppato un robot per la raccolta delle mele ma in grado di funzionare, anche, altri tipi di frutta. Questo, può lavorare 24 ore al giorno e ciò consente di migliorare drasticamente l'efficienza della produzione agricola. Il progetto della Abundant Robotics ha tenuto in considerazione:

- completare la raccolta nel periodo di tempo migliore
- garantire l'integrità del frutto durante la raccolta

La soluzione adottata è un braccio robotico dove la frutta viene "risucchiata" dal dispositivo terminale, la frutta rimane solo a contatto con l'aria per evitare di danneggiarla con un'attrezzatura meccanica.

2.2 In Cina, la scuola di agricoltura cloud

In Cina si parla di agricoltura intelligente, il "Tanjiawan Cloud Agricultural Experimental Site" a Wuzhen usa un connubio di 5G, Internet of Things e cloud computing per realizzare "cervello agricolo", in grado di raccogliere dati sulla crescita delle pintagioni, sull'umidità, tipi di terra e aria.

In questo modo coltivatori e agricoltori possono seguire la crescita del raccolto in tempo reale, da remoto a migliaia di chilometri di distanza.

Il successo è stato così grande che è stato il governo ha deciso di istituire una "scuola di agricoltura cloud"per avere nuovi agricoltori che conoscono la tecnologia e che comprendano i dati; gli esperti formati da questa scuola utilizzano le tecnologie da remoto per "entrare" nelle serre degli agricoltori da qualsiasi luogo, partecipare all'intero processo di produzione e rispondere alle domande degli agricoltori.

2.3 I robot nell'agricoltura indoor

Per l'agricoltura indoor si producono miliardi di piante in vaso ogni anno e quasi tutte richiedono una manipolazione ripetuta con un duro lavoro da svolgere ricurvi e monotono.

Anche in questo campo la robotica è venuta incontro, infatti, la Harvest Automation ha progettato "Harvey"[2], un piccolo robot mobile per l'uso in vivai e serre che localizza, trasporta e organizza autonomamente piante in vaso sia all'interno degli edifici che sui campi all'aperto.

2.4 La Agriculture Robotics in Europa

Anche l'Europa punta molto sulla "Agriculture Robotics":

- In un recente European Robotics Forum è stata dedicata una sessione intera alla robotica ed all'intelligenza artificiale per lo sviluppo rurale.
- La startup spagnola Agrobot ha ricevuto un finanziamento per sviluppare una raccoglitrice di fragole completamente automatica:
  - 24 "bracci" che possono raccogliere le fragole in modo indipendente.
  - utilizza l'apprendimento automatico per misurare la maturità delle fragole.
- A Bordeaux, i robot sono nei vigneti
- A Cognac, i robot attraversano il suolo dei cru Grand Champagne e Petite Champagne [3].
- il robot Ted, sviluppato da Naïo Technologies a Tolosa , fornisce risposte ai problemi dell'agricoltura sostenibile e sullle attività ad alta intensità di manodopera.

2.5 I robot nella lotta alle erbe infestanti

Nell'ambito della lotta alle erbe infestanti, data la portata dell'agricoltura, trattare chimicamente con utilizzare trattori o aeroplani per coprire grandi aree in un breve lasso di tempo le erbacce era finora l'unico modo per tenerle sotto controllo.

Bosch [3] ha sviluppato con la start-up Deepfield Robotics un robot agricolo in grado di rilevare autonomamente ed eliminare le singole infestanti.

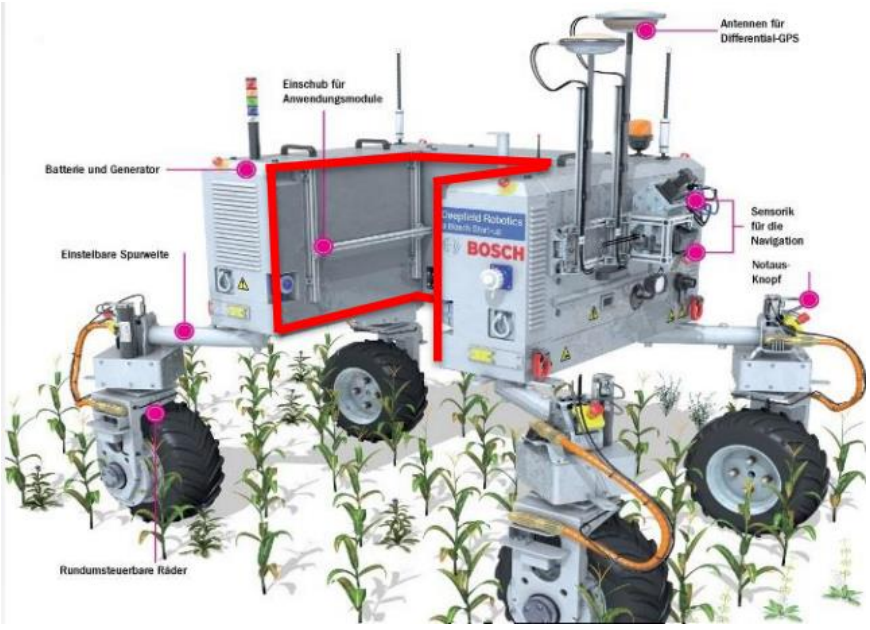


Figura 1- https://roscon.ros.org/2016/presentations/ROSCon2016_Agricultural_Robotics.pdf

Infatti, questo robot rimuove meccanicamente le piante infestanti attraverso uno strumento che schiaccia meccanicamente le piante infestanti dopo averle riconosciute attraverso un sistema avanzato di riconoscimento di visione artificiale [4].

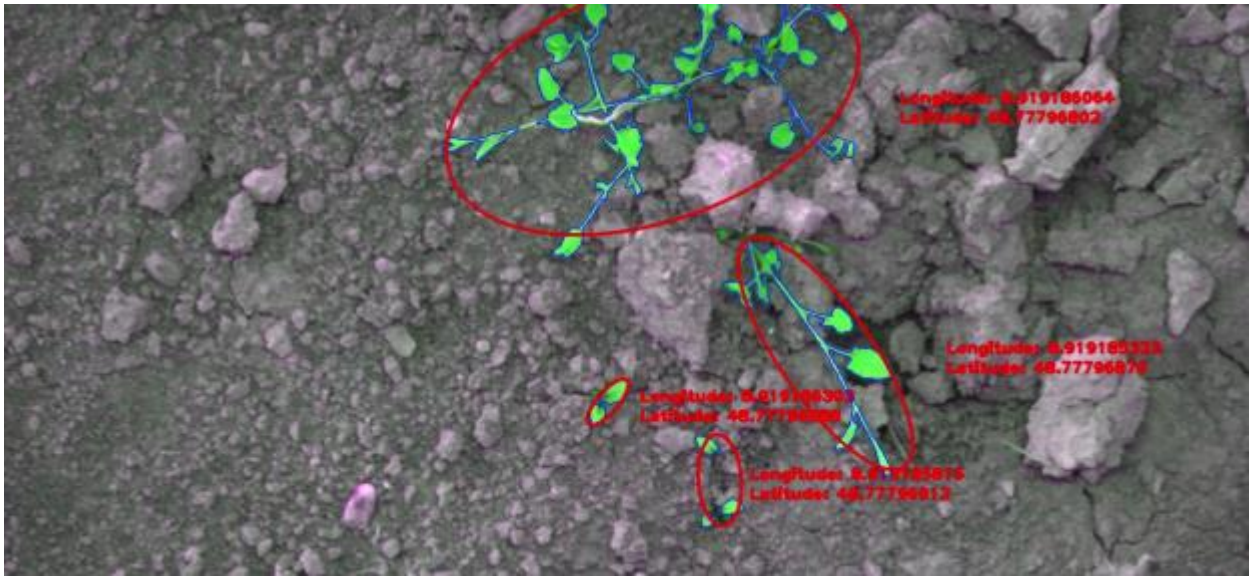


Figura 2 - https://roscon.ros.org/2016/presentations/ROSCon2016_Agricultural_Robotics.pdf

I droni robotici potrebbero essere usati per:

- l'ispezione e la sicurezza di infrastrutture
- la creazione di modelli 3D e di mappe.

Sciami robotici potrebbero essere efficacemente usati in ambito agricolo per l'agricoltura di precisione e le applicazioni agricole su larga scala.

È chiaro, quindi, come la robotica possa offrire degli spunti preziosi e significativi nell'agricoltura. I robot possono svolgere ruolo in tutte le fasi fondamentali:

- semina
- la rimozione degli infestanti
- raccolta dei frutti
- mappatura di punti di interesse.

Inoltre, si potrebbero realizzare sciami robotici per l'agricoltura di precisione e le applicazioni agricole su larga scala.

3. Il nostro obiettivo

Il corso di Robotica ci ha permesso, anche, di comprendere:

- l'interdisciplinarità di questa scienza (ingegneria, programmazione informatica, psicologia, automazione, meccanica, biologia) nelle fasi di progettazione, programmazione e sviluppo dei robot
- come questa scienza sia molto promettente per il futuro

Essendo così importante, la robotica è ormai diffusa in ogni settore, tuttavia la nostra attenzione è ricaduta, come lo scorso anno con il nostro progetto SERRA di "Linguaggi e Traduttori", nel mondo dell'agricoltura perché, ancora oggi, riteniamo che sia un settore che può avvantaggiarsi molto dalle nuove tecnologie e che queste tecnologie possono fare la differenza in un campo dove sempre più spesso si sente parlare di:

- cambiamento climatico
- cambiamento dei terreni agricoli
- crescita della popolazione globale
- preoccupazioni per la sicurezza alimentare a causa di eventi Socio-Politici
- diminuzione della disponibilità dei lavoratori agricoli

Infatti, grazie alla robotica, come si apprende dalla letteratura, le raccolte, a parità di acqua utilizzata ed estensione territoriale, possano aumentare del 70%.

4. Scenario di lavoro

L'applicazione che abbiamo immaginato per i nostri robot è quella dove più robot lavorano per un fine comune: curare un'area agricola; in questo scenario, la collaborazione non ha una struttura flat ma abbiamo preferito un robot coordinatore in grado fare valutazioni ad alto livello così da assegnare i compiti specifici al robot giusto (che nel nostro progetto prevederà un robot operaio in grado di irrigare). È stato scelto questo modo di operare perché le attività da svolgere possono essere molto diverse tra loro e quindi occorrono robot specializzati in determinate attività.

Abbiamo, quindi, inserito due tipi di robot:

- Robot Supervisore
    - mapping l'ambiente di lavoro
    - stabilisce i compiti da fare
    - invia le comunicazioni al robot operaio contenete
        - la mappa dell'ambiente di lavoro
        - il compito da svolgere
- Robot "operaio", è un robot specializzato in una determinata attività. Ad esempio, uno potrebbe irrigare, uno spruzzare diserbanti, uno raccogliere i frutti, uno individuare piante infestanti, etc...
    - Riceve le comunicazioni da parte del robot supervisore
        - Mappa area di lavoro
        - Compiti da svolgere
    - Definisce il percorso ottimo per raggiungere l'area di lavoro
        - A-Star
    - Raggiunge il punto di lavoro
    - Esegue il suo compito specifico (irrigazione, raccolta, etc...)

4.1 Altre caratteristiche

Stabilito lo scenario di lavoro del nostro progetto, abbiamo stabilito che dovesse integrare:

- esplorazione dell'ambiente di lavoro ignoto
    - il robot non ha una conoscenza priori dell'ambiente dove deve operare, tranne il numero delle celle totali del world, deve quindi individuare:
        - spazi attigui non percorribili, ossia celle libere ma separate, ad esempio, da uno steccato:

Figura 3 - oggetto webot

        - spazi occupati, ossia celle non percorribili

Figura 4 - oggetto webot

        - individuazione dei punti da lavorare e compiti da eseguire
- la collaborazione tra robot
    - I robot comunicano
        - Ogni robot dispone di un emettitore e di un ricevitore
    - si coordinato sul lavoro da fare
    - si scambiano informazioni sull'area di lavoro
- determinazione del cammino ottimale per raggiungere il punto da lavorare
    - basato sull'algoritmo A-star

4.2 Utenti finali

Il nostro progetto si rivolge a tutte quelle realtà agricole dove si desidera precedenza:

- un'agricoltura di precisione
    - ottimizzazione della produzione a parità di acqua e terreni
    - più sostenibile per l'ambiente
- applicazioni agricole su larga scala
- difficoltà nel reperimento della manodopera
- interventi tempestivi sulle piantagioni; ad esempio, si potrebbe inserire un sistema avanzato di riconoscimento di visione artificiale per bloccare sul nasce:
    - infestazioni da cocciniglia
    - erbe infestanti
    - etc…

5. I nostri Robot

Per la progettazione dei robot, e dell'ambiente di lavoro abbiamo usato Webots 2021b, un'applicazione con un ambiente di sviluppo completo per modellare, programmare e simulare i robot.

In particolare, per non ci siamo avvalsi di robot precostruiti ma abbiamo preferito crearne uno da zero che implementasse unicamente le caratteristiche da noi desiderate.

Sono presenti due Robot, un supervisore e un operaio; questi condividono la medesima base ma per l'operaio è stato previsto, in aggiunta, un hardware dedicato per le funzioni specifiche, ad esempio potrebbe irrigare:

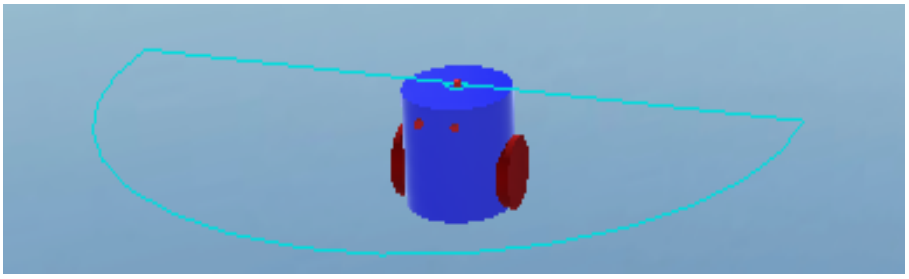


Figura 5 – Kípourompót Supervisore

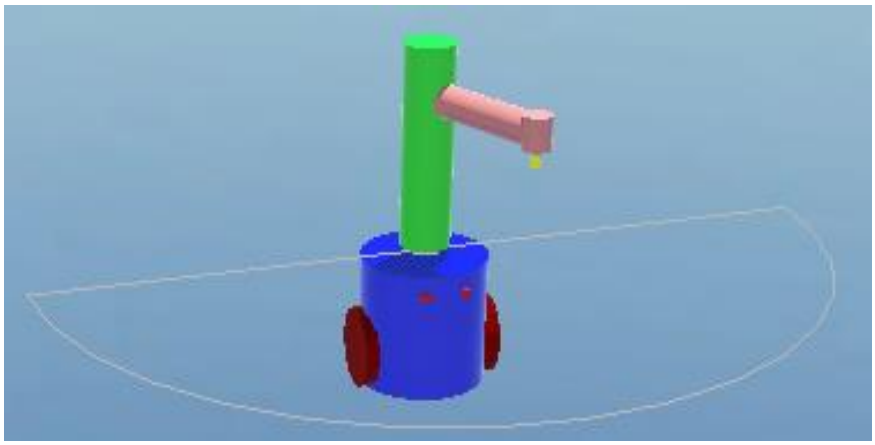


Figura 6 – Kípourompót Operaio

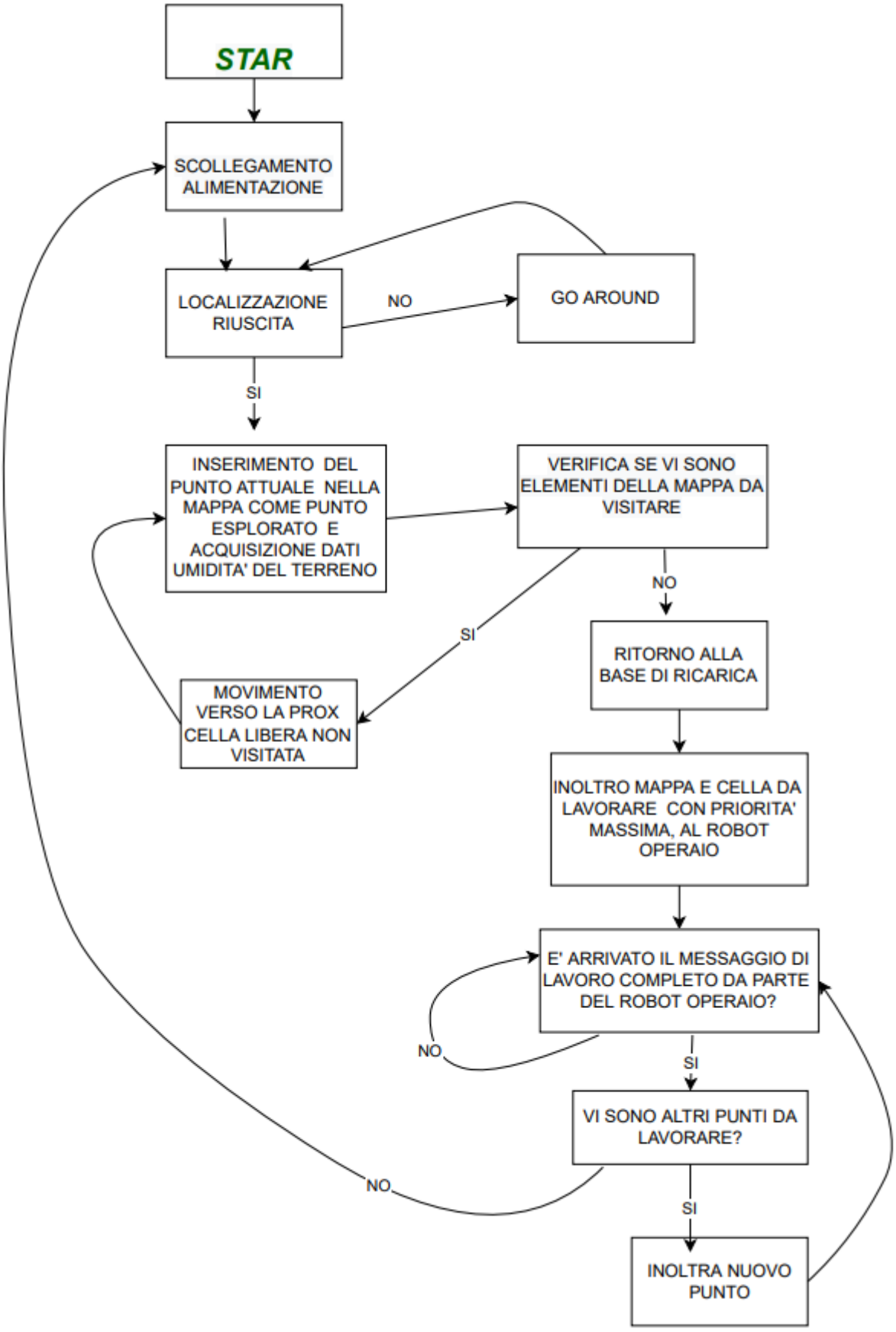
5.1 Compiti dei robot

I robot Supervisore e Operai svolgono compiti diversi, nello specifico:

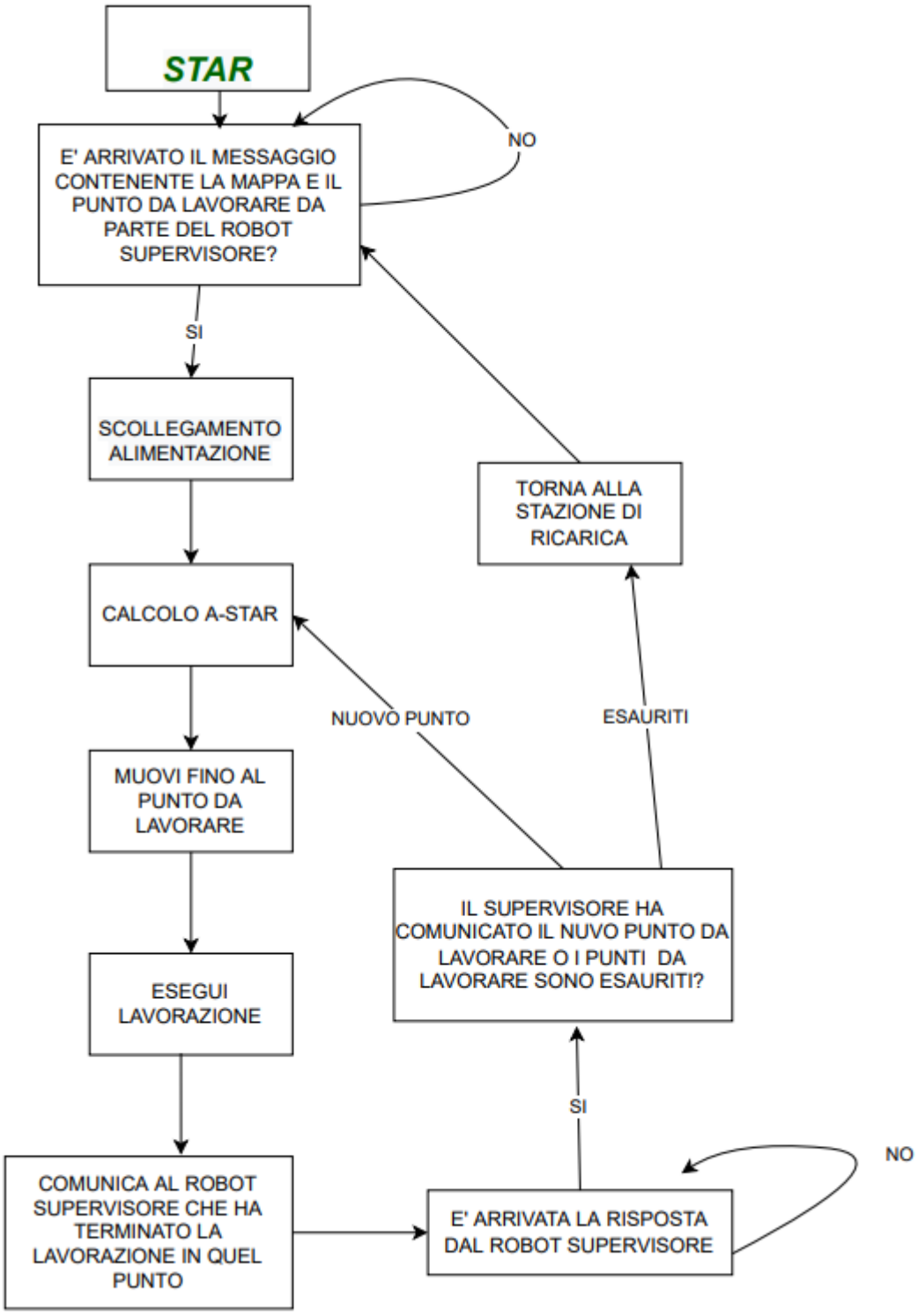
- Robot Supervisore
  - mappa l'ambiente di lavoro e si localizza utilizzando LiDAR e LandMark (alberi colorati alle estremità dell'arena), discriminando
    - celle percorribili e occupate
    - celle attigue comunicanti e non comunicanti
  - stabilisce i compiti per gli operai
  - comunica con gli operai per inoltrargli
    - mappa di lavoro
    - compito da svolgere
- Robot Operaio
  - Comunica col robot Supervisore per ottenere mappa di lavoro e compiti da svolgere

- È in grado di determinare il percorso migliore per ottenere il goal utilizzando l'algoritmo ASTAR e raggiungerlo
- Compie l'azione specifica (ad es. irrigazione)

5.2 Finite-State Machine Robot Supervisore

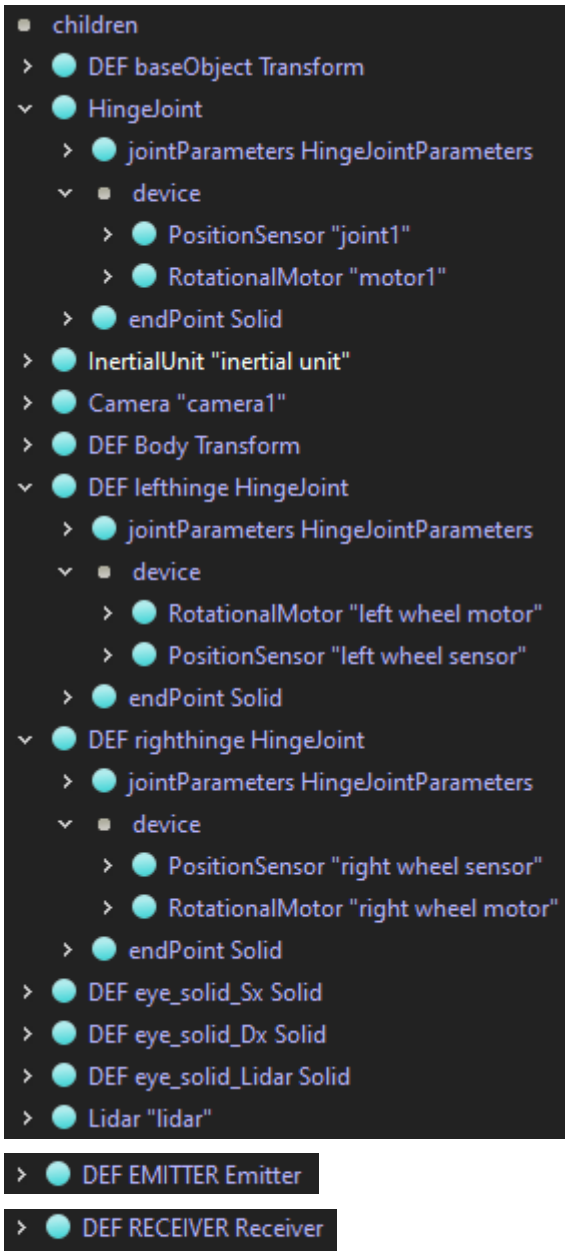


5.3 Finite-State Machine Robot Operaio



5.4 Specifiche tecniche

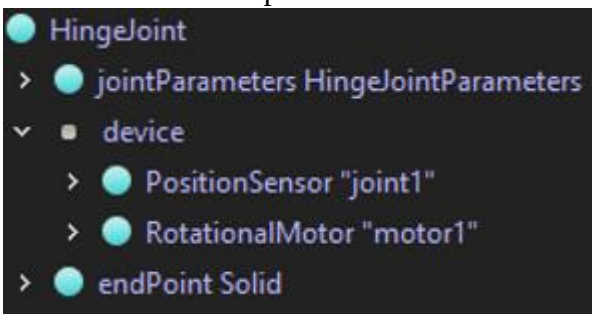
Per i nostri robot abbiamo previsto il seguente Hardware:



5.4.1 HINGEJOIN

Questo componente ci ha permesso di collegare:

- Il motore per il braccio che consente l'irrigazione



- Il motore SX e DX collegati alle ruote per il movimento

I nodi HingeJoint non sono visibili nella scena e possono essere utilizzati per modellare un giunto che consente solo un movimento rotatorio attorno a un determinato asse (1 grado di libertà).

Questo campo specifica opzionalmente un RotationalMotor, un PositionSensor angolare.

Se non viene specificato alcun motore, il giunto è un giunto passivo. [8]

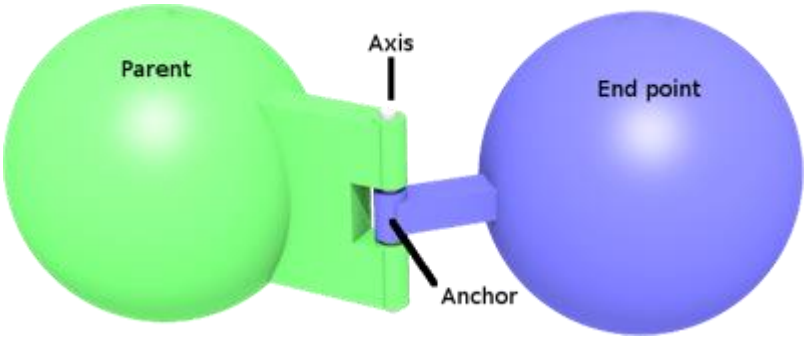
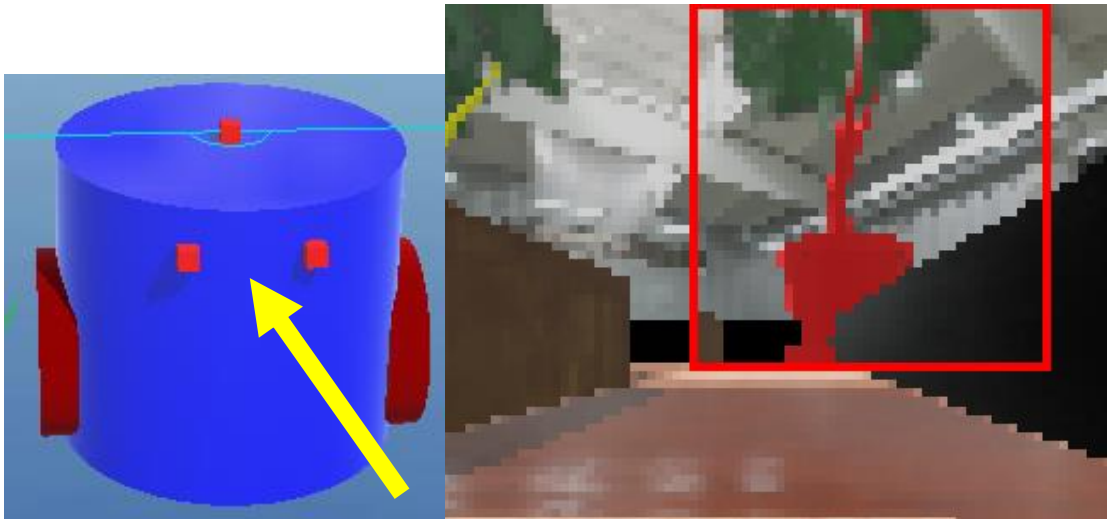


Figura 7- https://cyberbotics.com/doc/reference/hingejoint

5.4.2 CAMERA

Questo componente, posto tra i bottoni rossi, ci ha permesso di riconosce i LandMark costituiti da alberi, inseriti agli estremi dell'ambiente:

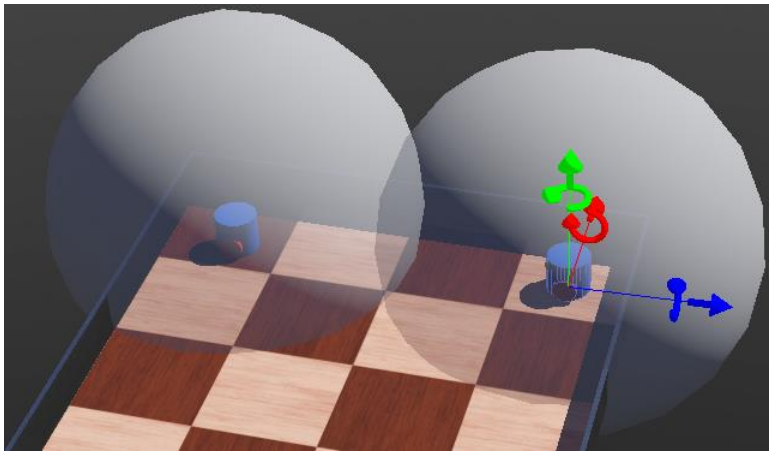


Il nodo Telecamera viene utilizzato per modellare la telecamera di bordo del robot. L'immagine risultante può essere visualizzata nella finestra 3D. A seconda della sua configurazione, il nodo Camera può modellare una telecamera lineare, una tipica telecamera RGB o anche un fish eye con distorsione sferica. [8]

5.4.3 RECEIVER - EMITTER

Questi componenti ci hanno permesso di scambiare dati tra i robot, ad esempio:

- La mappa di lavoro
- Gli ostacoli
- I tipi di lavoro da eseguire
- Lo stato dei lavori



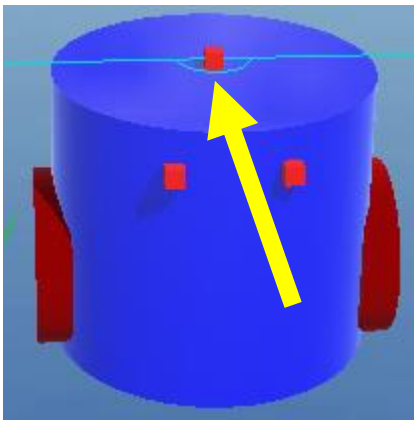
Il nodo Ricevitore viene utilizzato per modellare ricevitori radio, seriali o a infrarossi. Un nodo Ricevitore deve essere aggiunto ai figli di un robot o di un supervisore. Un Ricevitore può ricevere dati, ma non può inviarli. Per ottenere una comunicazione bidirezionale, un robot deve avere a bordo sia un Emettitore che un Ricevitore.[8]

Il nodo Ricevitore viene utilizzato per modellare ricevitori radio, seriali o a infrarossi. Un nodo Ricevitore deve essere aggiunto ai figli di un robot o di un supervisore. Un Ricevitore può

ricevere dati, ma non può inviarli. Per ottenere una comunicazione bidirezionale, un robot deve avere a bordo sia un Emettitore che un Ricevitore.[8]

5.4.4 LIDAR

Questo componente, posto nella parte centrale superiore del robot, ci ha permesso di identificare la posizione degli ostacoli presenti nell'area di lavoro:



Il nodo Lidar è usato per modellare il lidar (laser-scanner) a bordo del robot.

Il lidar misura le informazioni di profondità (in metri) da un rendering OpenGL, come fa il nodo RangeFinder.

Lidar rileva gli oggetti semitrasparenti come se non fossero trasparenti. Un oggetto può essere semitrasparente sia se la sua texture ha un canale alfa, sia se il suo campo di trasparenza Materiale non è uguale a 1.

Per impostazione predefinita, il nodo Lidar fornisce i valori di profondità in un array, ordinati da sinistra a destra e dal livello superiore a quello inferiore (come fa il nodo RangeFinder). Si tenga presente che la modalità nuvola di punti è costosa dal punto di vista computazionale e può quindi rallentare la velocità della simulazione. [8]

Figura 8 - https://cyberbotics.com/doc/guide/lidar-sensors

5.4.5 INERTIALUNIT

Il nodo InertialUnit simula un'unità di misura inerziale (IMU). L'InertialUnit calcola e restituisce gli angoli di rollio, beccheggio e imbardata rispetto a un sistema di coordinate globali definito nel nodo WorldInfo. Se si desidera misurare un'accelerazione o una velocità angolare, utilizzare invece il nodo Accelerometro o Giroscopio. Gli angoli di rollio, beccheggio e imbardata per i sistemi di coordinate ENU e NUE (campo coordinateSystem del nodo WorldInfo) sono rappresentati nella figura seguente. [8]

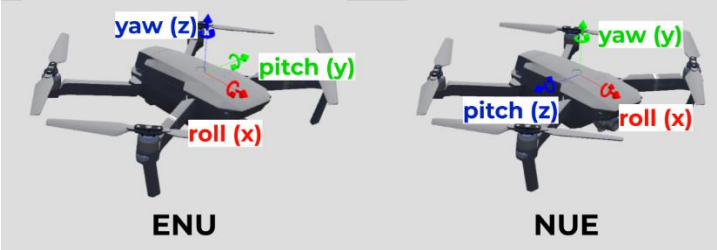


Figura 9- https://cyberbotics.com/doc/reference/inertialunit

6. World di lavoro

L'ambiente di lavoro sul quale operano i Robot è stato creato, anch'esso, con Webots. Gli ambienti di lavoro per eseguire i test sono stati 4x4 e 8x8 dove ogni cella ha una dimensione di 1 metro x 1 metro; tuttavia, la maggior parte del codice per i controllori e i vari moduli richiamati dal main sono in grado di lavorare con qualsiasi dimensione specificata in ingresso.

Inizialmente, il world si presenta sotto questa forma:

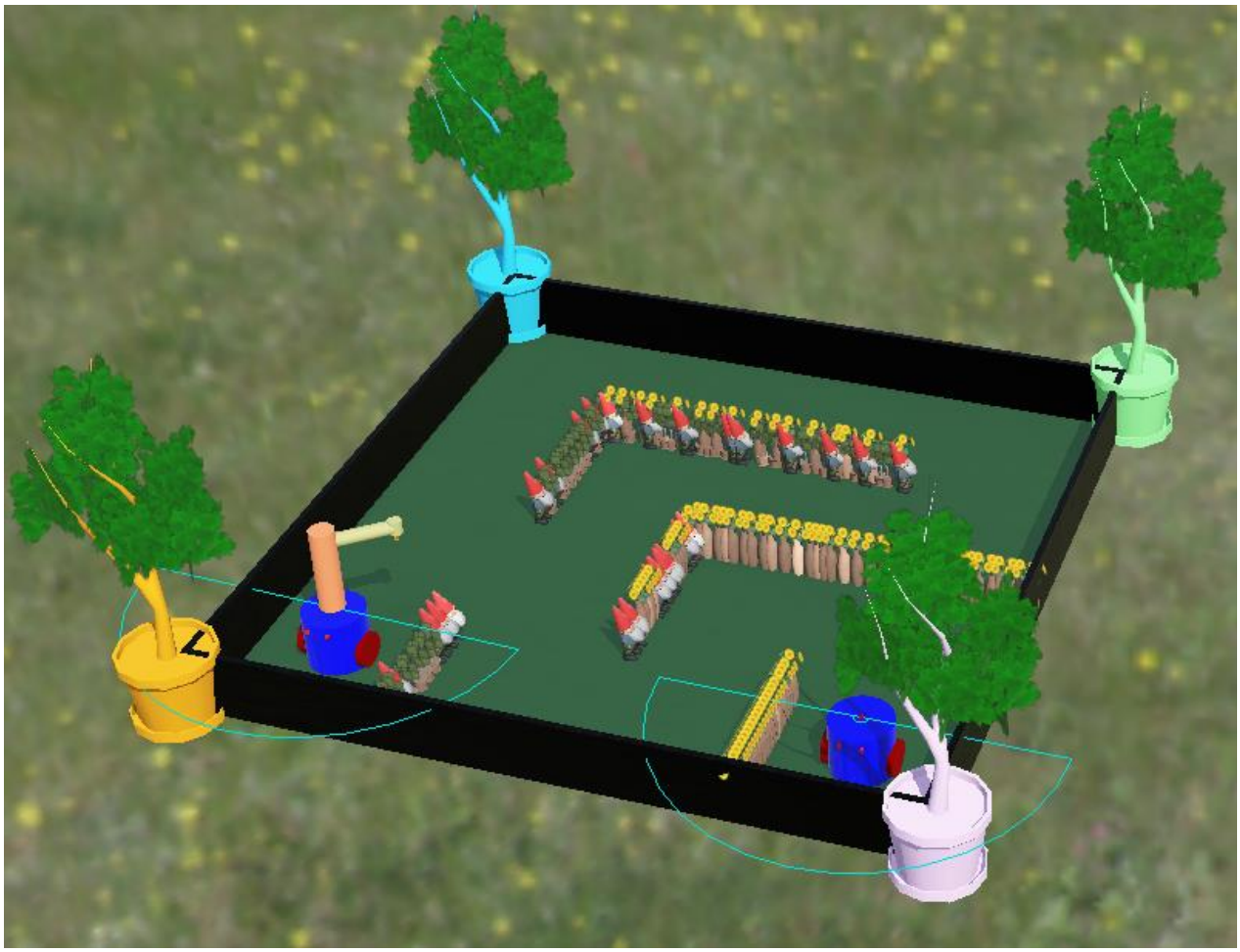


Figura 10 - world

I robot, inizialmente, sono nelle loro postazioni di ricarica :

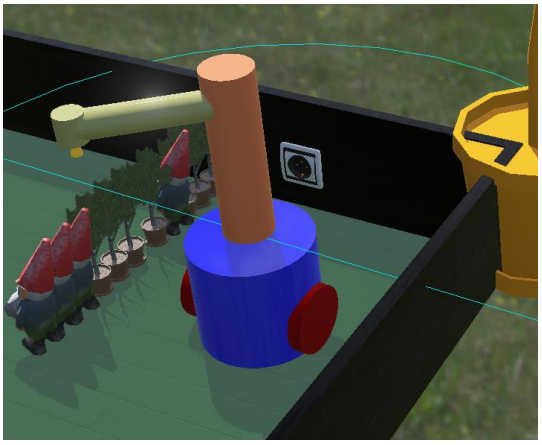


Figura 11 - Plug robot

L’ambiente di lavoro dei robot è costellato da ostacoli costituiti da tre tipi di oggetti:

Figura 12 - ostacoli presenti nel World

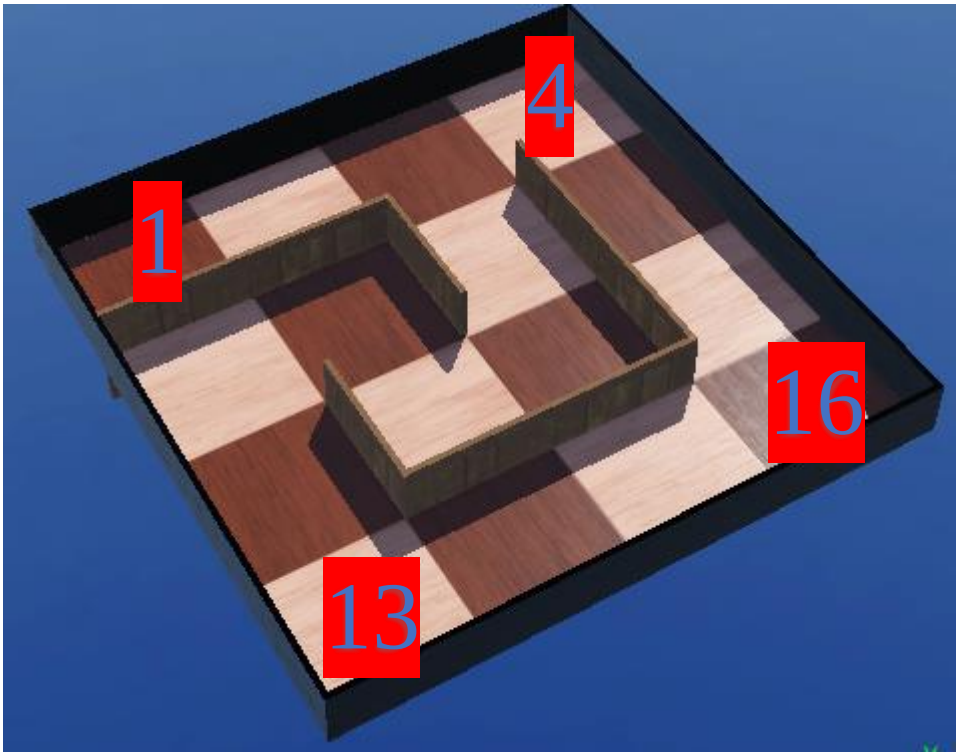
Il robot ha contezza dell’ambiente attraverso una lista di liste dove ogni sotto lista rappresenta una cella della mappa; quindi, per una 4x4 sarà quindi del tipo:

```
world = [[0, 1, 1, 0, 0], [0, 0, 1, 0, 1], [0, 0, 1, 0, 0], [0, 0, 1, 1, 0],
         [0, 1, 0, 1, 1], [0, 1, 1, 1, 0], [0, 1, 0, 1, 0], [0, 1, 0, 1, 0],
         [0, 1, 1, 1, 0], [0, 1, 0, 0, 1], [0, 0, 0, 1, 0], [0, 1, 0, 1, 0],
         [0, 1, 0, 0, 1], [0, 0, 1, 0, 1], [0, 0, 0, 0, 1], [0, 0, 0, 1, 1]]
```

Dove ogni sotto lista indica:

1. Se la cella è stata già visita (per il Robot Supervisore), oppure il passo della path per il Robot operaio.
2. Ostacolo a Ovest rispetto alla cella
3. Ostacolo a Nord rispetto alla cella
4. Ostacolo ad est rispetto alla cella
5. Ostacolo a Sud rispetto alla cella

Quindi, ad esempio, con questa lista di liste il world sarà:



Dunque, le celle 1, 4, 13, 16 saranno descritte dalle seguenti liste:

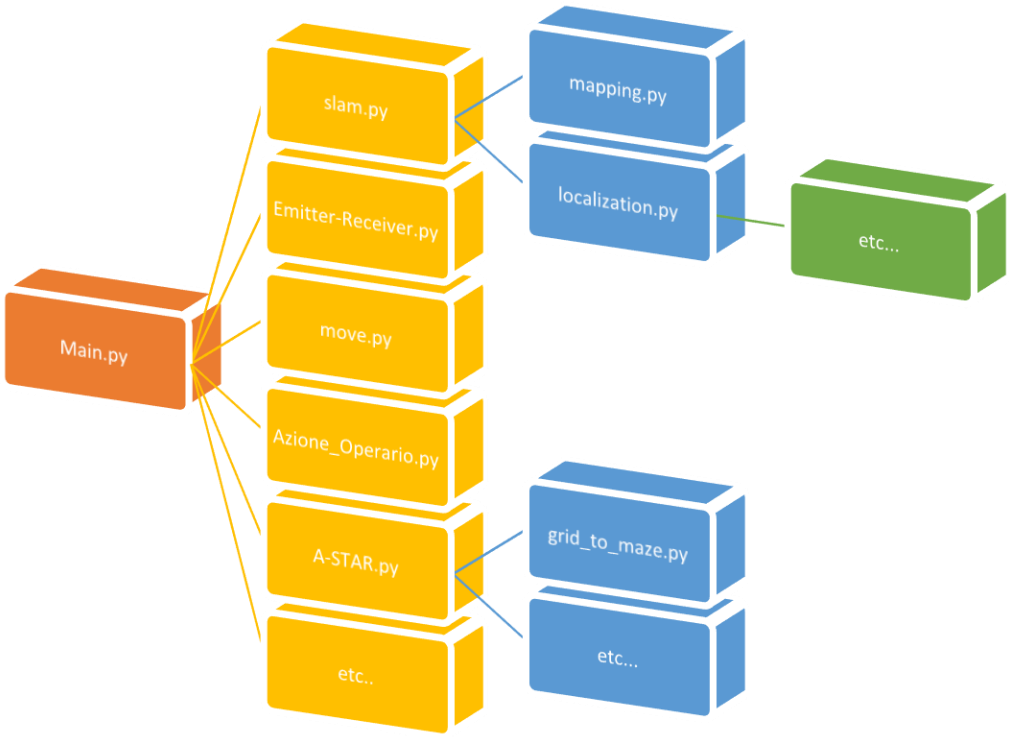
- `[0, 1, 1, 0, 0]`
- `[0, 0, 1, 1, 0]`
- `[0, 1, 0, 0, 1]`
- `[0, 0, 0, 1, 1]`

7. Scelte progettuali e Algoritmi utilizzati

Dopo avere definito cosa dovessero fare i nostri robot, è stato scelto di usare un approccio modulare per scrivere il codice. Tale scelta è stata presa per garantire:

- Scalabilità
- Facilità di identificazione e di correzione dei bug
- Facilità di inserimento di nuove funzioni

Al nostro codice è stata data una struttura di questo tipo:



Ossia, prevede un file main.py che va a richiamare i diversi moduli a seconda dei compiti che devono essere svolti.

La gran parte del codice (main e moduli) è stata scritta per accettare in ingresso qualsiasi dimensione dell'ambiente di lavoro così da adattarsi alle diverse esigenze di dimensionamento.

Tuttavia, a causa di mancanza di tempo, resta una parte del codice che è specifica per funzionare su arene 4x4 e 8x8.

Abbiamo scelto di scrivere il codice interamente in Python 3.10 perché:

- volevamo un linguaggio di programmazione orientato agli oggetti
- per la sua elevata leggibilità sia per la sua
- fornisce una libreria standard che gestisce in modo automatico la memoria
- può eseguire lo stesso codice su molteplici piattaforme.

Sono state integrate le librerie:

- Numpy
- Time
- Math

- Random
- Struct

I moduli Webots importati per il corretto funzionamento sono stati:

- Lidar
- Emitter
- Receiver
- Robot
- Camera
- CameraRecognitionObject
- InertialUnit
- PositionSensor

7.1 Slam

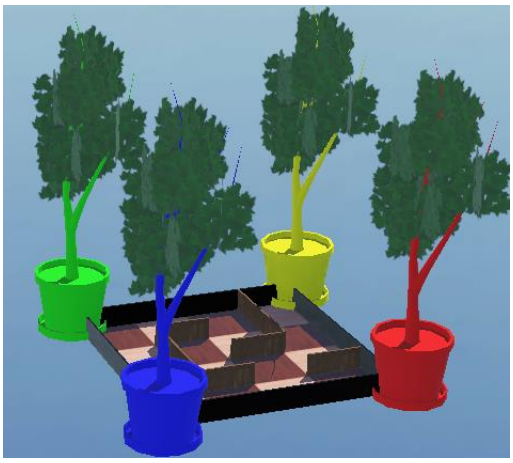
La localizzazione e la mappatura simultanee (SLAM) è il problema computazionale dove si deve costruire o aggiornare una mappa di un ambiente sconosciuto tenendo contemporaneamente traccia della posizione di un agente al suo interno. Gli algoritmi SLAM sono ritagliati sulle risorse disponibili, quindi non finalizzati alla perfezione, ma alla operatività del sistema. Sono utilizzati nella navigazione robotica, nella mappatura robotica e nell'odometria, trovando, quindi, applicazione in diversi campi: auto a guida autonoma, veicoli aerei senza pilota, veicoli subacquei autonomi, rover planetari, nuovi robot domestici.

7.1.1 Slam in Κήπουρομπότ

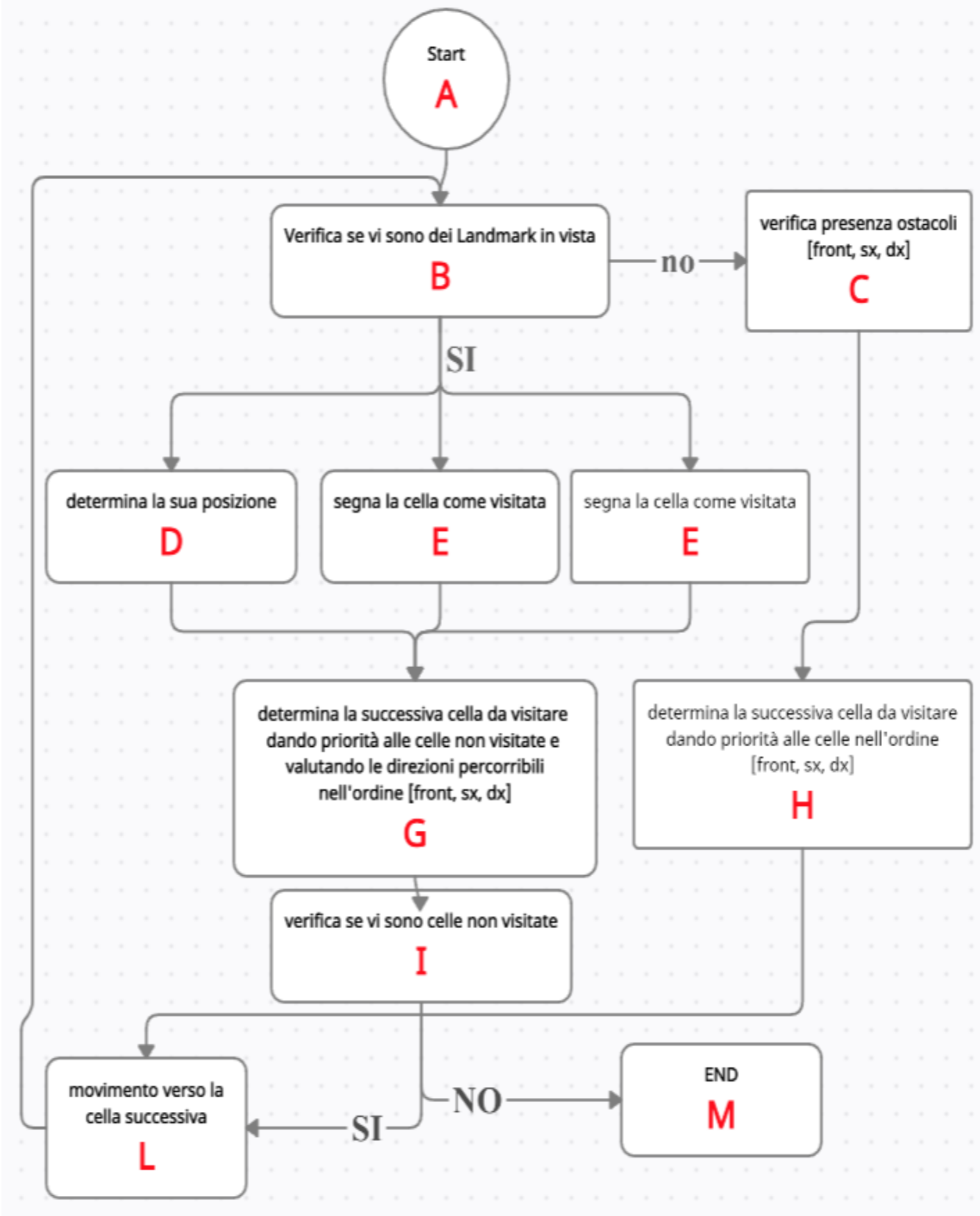
Il nostro robot, come spiegato in maniera più esaustiva nelle Conclusioni, non integra un vero SLAM perché, come anticipato, gli algoritmi sono ritagliati sulle risorse disponibili.

Siamo, quindi, scesi a un compromesso progettuale che prevede l'utilizzo di una matrice, dove il robot:

- Conosce il numero di celle presenti nel mondo
- si muove di cella in cella ed esamina, attraverso il LIDAR a tiro corto se vi sono ostacoli
- Approssima la propria posizione rispetto alla mappa utilizzando degli alberi come Punto di riferimento:



7.1.2 Schema di funzionamento



7.2 A-Star

Abbiamo deciso di utilizzare come, path search algorithm, l'algoritmo A-STAR perché è un algoritmo:

- Completo
  - Se una soluzione esiste, l'algoritmo termina e la
- Efficace
  - In presenza di soluzioni multiple, l'algoritmo è in grado di trovare la migliore
- Ottimale
  - Data un'euristica h(.), nessun altro algoritmo garantisce l'esplorazione di un minor numero di stati.

La componente euristica è una strada obbligata per risolvere problemi molto difficili come "il problema del commesso viaggiatore"; infatti l'euristica, in Informatica, è una delle tecniche usate per risolvere problemi ottimizzando il parametro tempo quando esso risulta molto alto con altre tecniche classiche (esaustive) che si usano normalmente. Infatti, mentre i metodi classici esaustivi trovano la soluzione esatta, l' euristico trova una soluzione approssimata e quindi si usa quando il metodo classico fallisce nel trovare una soluzione in tempi ragionevoli.

Nel caso pessimo, una funzione euristica costante, A* diviene un algoritmo di ricerca molto simile a Dijkstra.

7.2.1 A-STAR in Κήπουρομπότ

Per il nostro progetto si è scelto di non usare librerie già esistenti per Python perché ci si è resi conto che quanto già presente non era adatto al nostro utilizzo. Infatti, le librerie presenti in Python calcolano il percorso su una matrice dove le celle possono essere considerate o percorribili o non percorribili:

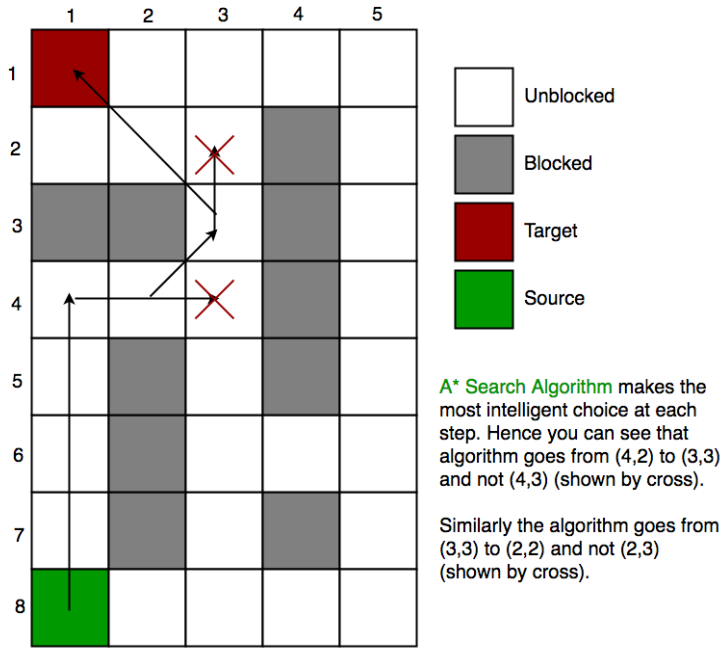


Figura 13 - https://media.geeksforgeeks.org/wp-content/uploads/a_-search-algorithm-2.png

Figura 14- oggetto webots che occupa una cella intera

Quello che si è deciso di fare è stato di considerare, anche, la situazione nella quale due celle adiacenti singolarmente percorribili siano separate da un ostatolo:

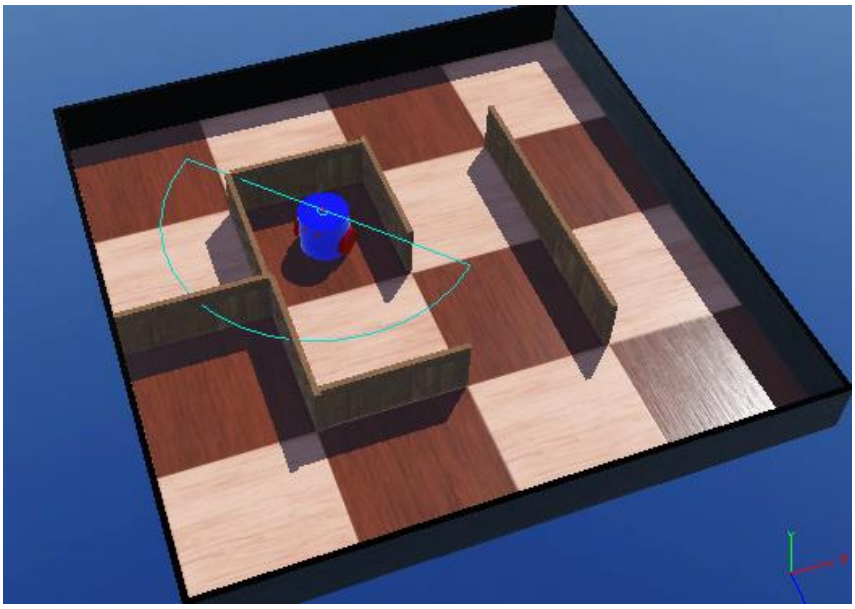


Figura 15 - mappa di lavoro dovetutte le celle sono percorribili ma non tutte sono comunicanti

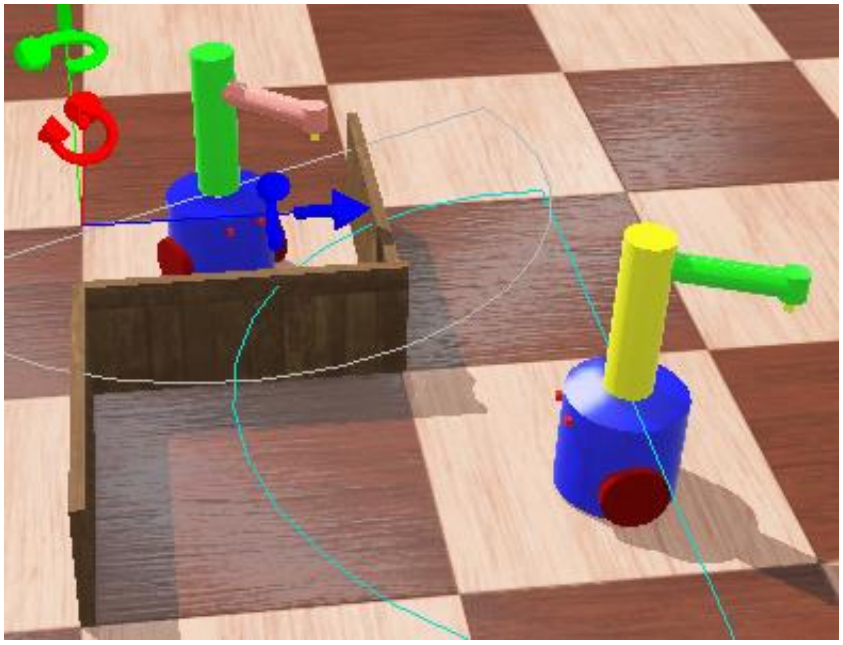


Figura 16 - mappa di lavoro dove tutte le celle sono percorribili ma non tutte sono comunicanti

Per ottenere questo risultato, partendo da un codice didattico di A-Star essenziale e operante in maniera classica su una matrice 4x4, si è modificato per farlo funzionare nella maniera voluta.

7.2.2 Meccanismo di funzionamento

L'Algoritmo A-Star viene avviato dalla funzione "def a_star2(punto_finale, punto_iniziale, mappa_da_convertire)" che:

- riceve in ingresso lo start point, l'end point e la mappa elaborata dal robot supervisore
- restituisce la mappa aggiornata che contiene i punti da visitare per andare dallo start point all'end point

All'interno vengono eseguite le seguenti operazioni:

- riceve la mappa dove ogni sottolista indica [Visitata, Ovest, Nord, Est, Sud]

```
[[0, 1, 1, 0, 0], [0, 0, 0, 0, 1], [0, 0, 1, 0, 0], [0, 0, 1, 1, 0],
 [0, 1, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 1, 0],
 [0, 1, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 1, 0],
 [0, 1, 0, 0, 1], [0, 0, 0, 0, 1], [0, 0, 0, 0, 1], [0, 0, 0, 1, 1]]
```

- vengono applicata una maschera pesi fittizi infiniti verso tutti i percorsi per indicare all'algoritmo che ancora non sono stati presi in esame:

```
[['inf', 'X', 'X', 0,   0], ['inf', 0, 'X', 0,   0], ['inf', 0, 'X', 0,   0], ['inf', 0, 'X', 'X',   0],
 ['inf', 'X',   0, 0,   0], ['inf', 0,   0, 0,   0], ['inf', 0,   0, 0,   0], ['inf', 0,   0, 'X',   0],
 ['inf', 'X',   0, 0,   0], ['inf', 0,   0, 0,   0], ['inf', 0,   0, 0,   0], ['inf', 0,   0, 'X',   0],
 ['inf', 'X',   0, 0, 'X'], ['inf', 0,   0, 0, 'X'], ['inf', 0,   0, 0, 'X'], ['inf', 0,   0, 'X', 'X']]
```

- viene creata la maze dove righe e colonne dispari indicano i muri fittizi che servono per potere identificare le celle attigue dalle quali non è possibile passare da una all'altra

```
[[0, 0, 0, 0, 0, 0, 0],
 [0, 1, 0, 1, 0, 1, 0],
 [0, 0, 0, 0, 0, 0, 0],
 [0, 1, 0, 1, 0, 1, 0],
 [0, 0, 0, 0, 0, 0, 0],
 [0, 1, 0, 1, 0, 1, 0],
 [0, 0, 0, 0, 0, 0, 0]]
```

quindi per una mappa 4x4 la maze avrà una dimensione 7x7, mentre per una mappa 8x8 sarà mappata in una 15x15

- vengono inseriti gli ostacoli reali nella maze:
  [[0, 0, 0, 0, 0, 0, 0],
  [0, 1, 1, 1, 0, 1, 0],
  [0, 1, 0, 1, 0, 1, 0],
  [1, 1, 0, 1, 0, 1, 0],
  [0, 1, 0, 0, 0, 1, 0],
  [0, 1, 1, 1, 0, 1, 0],
  [0, 0, 0, 0, 0, 0, 0]]

- l'algoritmo ASTAR calcola il percorso su quest'ultima lista di liste:
  [[-1, -1, -1, -1, -1, -1, -1],
  [-1, -1, -1, -1, -1, -1, -1],
  [-1, -1, 0, -1, -1, -1, -1],
  [-1, -1, 1, -1, -1, -1, -1],
  [12, -1, 2, 3, 4, -1, -1],
  [11, -1, -1, -1, 5, -1, -1],
  [10, 9, 8, 7, 6, -1, -1]]

  dove il percorso da seguire è dato dai numeri >= 0

- vengono scartati i punti contenenti i muri virtuali, si torna quindi ad una 4x4:

```
[[-1 -1 -1 -1]
 [-1  0 -1 -1]
 [12  2  4 -1]
 [10  8  6 -1]]
```

- viene ricomposta in una sequenza ordinata:
  [[-1,-1,-1,-1], [-1,2,-1,-1], [8,3,4,-1], [7,6,5,-1]]

- viene restituita la grid_maze al controllore, aggironata e contenente come primo elemneto un numero che indica come valore il passaggio nel percorso tra il punto iniziale e finale :
  [
  [2, 'X', 'X', 0, 0], [3, 0, 'X', 0, 1], [4, 0, 'X', 0, 0], ['inf', 0, 'X', 'X', 0],
  ['inf', 'X', 0, 1, 1], ['inf', 1, 1, 1, 0], [5, 1, 0, 1, 0], ['inf', 1, 0, 'X', 0],
  ['inf', 'X', 1, 1, 0], ['inf', 1, 0, 0, 1], [6, 0, 0, 1, 0], ['inf', 1, 0, 'X', 0],
  ['inf', 'X', 0, 0, 'X'], ['inf', 0, 1, 0, 'X'], [7, 0, 0, 0, 'X'], [8, 0, 0, 'X', 'X']]

7.2.3 Emitter – Receiver

Per inviare un pacchetto di dati ai potenziali ricevitori, la funzione wb_emitter_send aggiunge alla coda dell'emettitore un pacchetto di byte di dimensioni pari all'indirizzo indicato da data. I pacchetti di dati in coda saranno inviati ai potenziali destinatari (e rimossi dalla coda dell'emettitore) alla velocità specificata dal campo baudRate del nodo Emitter. La coda è considerata piena quando la somma dei byte di tutti i pacchetti attualmente in coda supera la dimensione del buffer specificata dal campo bufferSize.

In particolare, con Python, la funzione send invia una stringa; in alternativa si possono inviare tipi di dati primitivi in questa stringa con il modulo struct.

Quindi, nel nostro caso, prevedendo lo scambio di informazioni tra i robot, anche, con:

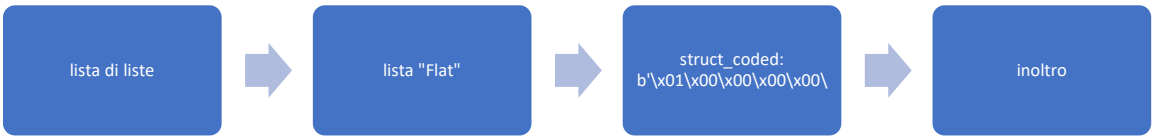
- lista di liste
- array

si è reso necessario ripensare la struttura dati in un formato del tipo:

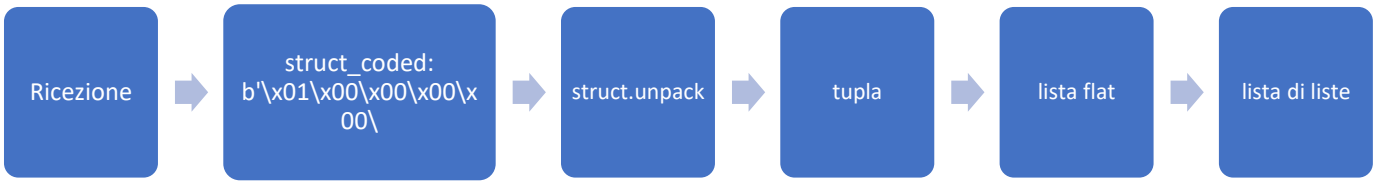
```python
def divide_chunks(l, n):
    for i in range(0, len(l), n):
        yield l[i:i + n]

def flatten_list(_2d_list):
    flat_list = []
    for element in _2d_list:
        if type(element) is list:
            for item in element:
                flat_list.append(item)
        else:
            flat_list.append(element)
    return flat_list

grid_maze = [[1, 2, 3, 4, 5], [6, 7, 8, 9, 10],[11, 12, 13, 14, 15]]
flat_grid_maze = flatten_list(grid_maze)

struct_coded = struct.pack("<%uQ" % len(flat_grid_maze), *flat_grid_maze)
struct_decoded = struct.unpack("<%uQ" % len(flat_grid_maze), struct_coded)

grid_maze_decoded_flat = list(struct_decoded)

grid_maze_decoded_chunks = list(divide_chunks(grid_maze_decoded_flat, 5))
```

ossia, in modo schematico:

- Lato Emitter:



- Lato Receiver:



I messaggi scambiati avranno una struttura del tipo:

- Lato Supervisore

```
ROBOT ["supervisore"] ha ricevuto il seguente
messaggio:
            id_message:  wfOnLVKmGKra6Ol0
            mittente:  operaio
            destinatario:  supervisore
            operazione:  SEND_ME_OP
            goalPosOp:  -1
            time_send:  103.04000091552734
            mappa:  [[1, 1, 1, 0, 1], [1, 0, 1, 0,
1], [1, 0, 1, 1, 0], [3, 1, 1, 1, 0], [1, 1, 1, 0,
0], [1, 0, 1, 1, 0], [1, 1, 0, 0, 0], [1, 0, 0, 1,
0], [1, 1, 0, 1, 0], [1, 1, 0, 0, 1], [1, 0, 0, 1,
1], [1, 1, 0, 1, 0], [1, 1, 0, 0, 1], [1, 0, 1, 0,
1], [1, 0, 1, 0, 1], [1, 0, 0, 1, 1]]
```

- Lato Operaio

```
ROBOT ["operaio"] ha ricevuto il seguente messaggio:
            id_message:  dvVORxDSzfW3Mulb
            mittente:  supervisore
            destinatario:  operaio
            operazione:  OP_SENDED
            goalPosOp:  12
            time_send:  103.0719985961914
            mappa:  [[1, 1, 1, 0, 1], [1, 0, 1, 0,
1], [1, 0, 1, 1, 0], [3, 1, 1, 1, 0], [1, 1, 1, 0,
0], [1, 0, 1, 1, 0], [1, 1, 0, 0, 0], [1, 0, 0, 1,
0], [1, 1, 0, 1, 0], [1, 1, 0, 0, 1], [1, 0, 0, 1,
1], [1, 1, 0, 1, 0], [1, 1, 0, 0, 1], [1, 0, 1, 0,
1], [1, 0, 1, 0, 1], [1, 0, 0, 1, 1]]
```

8. Conclusioni

Abbiamo raggiunto il nostro obiettivo, ossia, creare un ambiente simulato di lavoro dove diversi robot riescono a cooperare per portare avanti un compito comune: la cura della coltura. È stato possibile realizzarlo grazie all'ideazione di uno scenario dove sono presenti robot specializzati in determinate attività:

- Robot supervisore
    - Si localizza e Mappa un ambiente sconosciuto
    - Definisce i punti del giardino dove intervenire
    - Contatta via radio i robot operai comunicando i punti di lavoro e il compito da eseguire
- Robot operaio
    - Riceve via radio l'ambiente di lavoro sul quale operare e il compito da eseguire
    - Definisce con l'algoritmo A-Star il percorso per arrivare sul punto di lavoro
    - Esegue il compito nel quale è specializzato
    - Contatta il robot supervisore per informarlo che ha esaurito il compito
    - Resta in attesa di ulteriori istruzioni.

Riteniamo, inoltre, che il nostro progetto abbia una buona scalabilità grazie alle scelte progettuali fatte:

- Programmazione modulare
    - La struttura che è stata data al codice consente velocemente di integrare nuovi sensori e funzioni nei robot
    - La maggior parte del codice funziona con qualsiasi dimensione dell'area di lavoro per adeguarsi ad ambienti diversi, e solo una piccola parte deve essere, ancora, generalizzata.
- A-Star
    - Benché, i test siano stati eseguiti considerando un costo uniforme tra un nodo e il successivo, l'algoritmo A-star integrato è in grado di lavorare con pesi differenti. È quindi, possibile inserire maggiori informazioni per ottenere risulti migliori in realtà diverse.

8.1 Compromesso di progettazione

In avvio di progettazione, nello SLAM, si è deciso di affidare l'acquisizione delle informazioni al Laser Imaging Detection and Ranging e al GPS. L'obiettivo dell'implementazione di questi potentissimi strumenti era il mapping tridimensionale ad altissima risoluzione e il rilevamento dello spostamento.

Tuttavia, dopo settimane sviluppo, ci siamo resi conto che la quantità di informazioni acquisite e poi inserite in un array-3d, non ci consentivano di continuare di questa direzione, perché volendo parzializzare nell'ordine dei mm l'acquisizione diventava un'operazione complessa, così come anche l'inoltro della mappa da un robot all'altro.

Esistono diverse librerie (ROS2) che avrebbero consentito di acquisire e usare strutture dati già pronte, ma si è preferito di non intraprendere questa strada a favore dell'adozione di un compromesso nella progettazione e di riservare la riscrittura della mappatura per gli sviluppi futuri. Ciò che è stato deciso, è stato di considerare la presenza o meno di un ostacolo

parzializzando l'area di lavoro a 100 cm, questo consente di lavorare con array dove ogni cella indica se entro 1 metro è presente un ostacolo.

9. Sviluppi Futuri

Per il nostro progetto, in futuro, ci piacerebbe migliorarlo e ampliarlo con nuove caratteristiche:

- renderlo più vicino ad un uso reale
    - occorrerebbe rivedere la componente che lavora per lo SLAM in modo che il robot realizzi una mappa fedelissima alla realtà così da potersi muovere con la massima precisione in un modo reale. A tale scopo si potrebbe pensare di utilizzare strutture dati più efficienti basate su schemi di hashing.
- Renderlo maggiormente scalabile per aree più estese
    - pensando ad una mappatura non più eseguita da un singolo robot ma più robot potrebbero mappare contemporaneamente aree diverse per poi essere unite.
- Implementare, attraverso un sistema avanzato di riconoscimento di visione artificiale, identificare piante infestanti, parassiti, etc..

10. Bibliografia

1. Pathan, M., et al., Artificial cognition for applications in smart agriculture: A comprehensive review. 2020.
2. Agricoltura di Precisione Linee Guida x, A.d. Precisione, Editor. 2015: www.politicheagricole.it.
3. KRAVITZ, M., How Robots are Taking Over Vineyards, in Wine Enthusiast. 2019.
4. Michaels, A., S. Haug, and A. Albert. Vision-based high-speed manipulation for robotic ultra-precise weed control. in 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2015. IEEE.
5. https://www.agricultural-robotics.com/
6. https://robohub.org/how-regenerative-agriculture-and-robotics-can-benefit-each-other/
7. https://spectrum.ieee.org/automaton/robotics/industrial-robots/bosch-deepfield-robotics-weed-control
8. www.webots.com