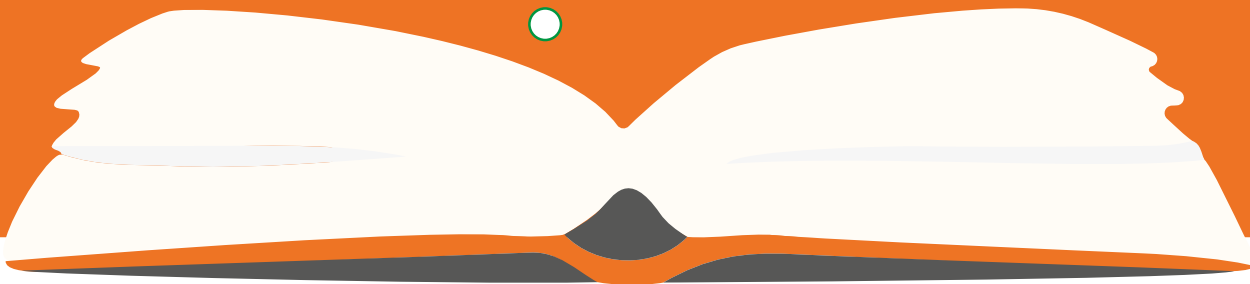


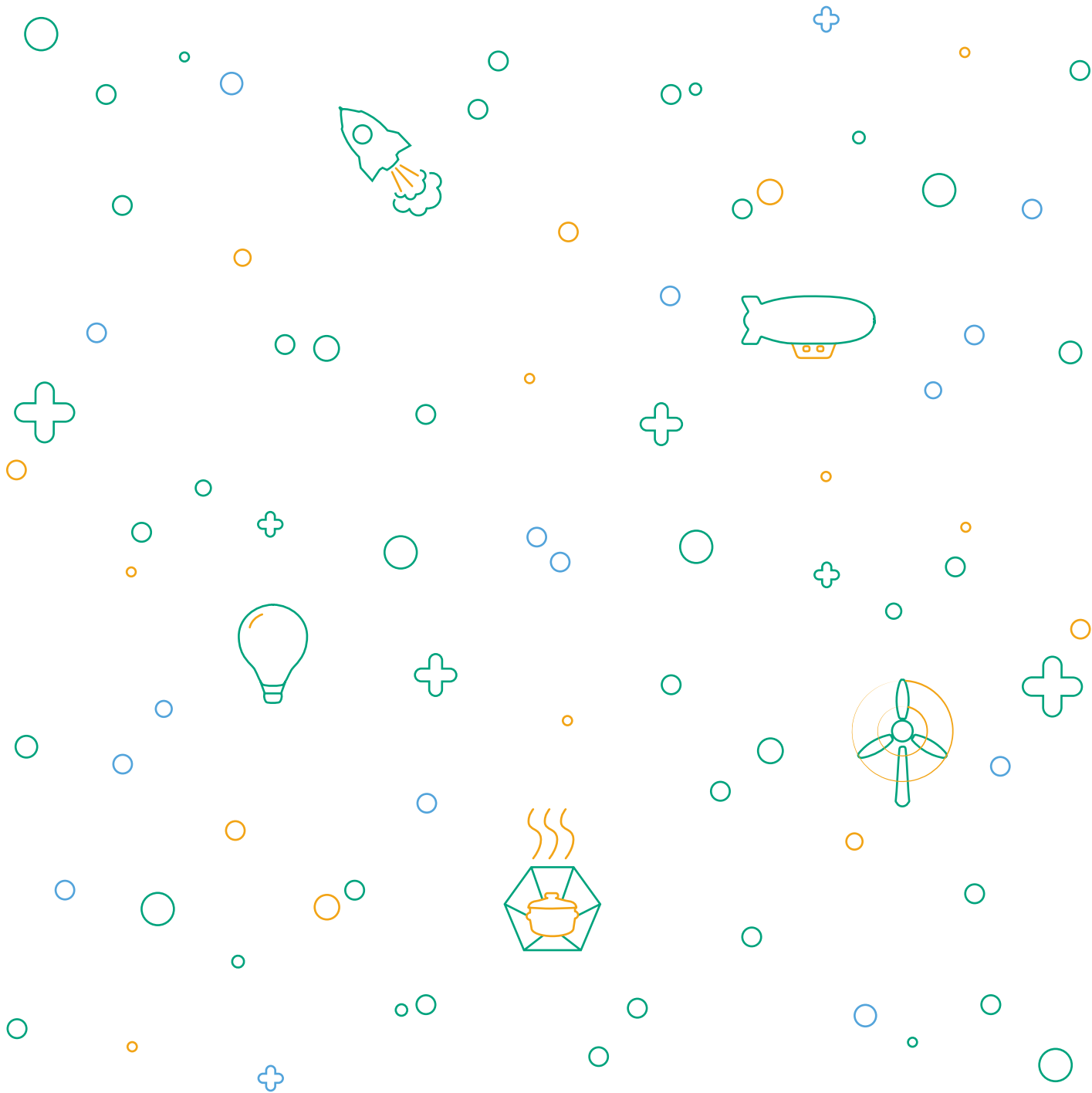
partnership con



VIAGGIARE SU MARTE !



MANUALE PER L'USO
ROBOT MARZIANO





MANUALE PER L'USO ROBOT MARZIANO

Edizione redatta da
@P7i7Plum3 e Damien Vallot
Impaginata e illustrata da
Laura Venezia e Diana Khalipina

Diritti d'autore riservati
• Edizione 2021 •
Stampato in Italia

ID Card
ARES XVI



Cognome: **Rophile**.....

Nome: **Chloé**.....

Professione:.....
Esperta informatica

Authorization
level 5

ID Card
ARES XVI



Cognome: **Fort**.....

Nome: **Rock**.....

Professione:.....
Meccanico

Authorization
level 5

**I NOSTRI DUE COLONI
MARZIANI PRONTI A
RACCOGLIERE TUTTE
LE SFIDE!**

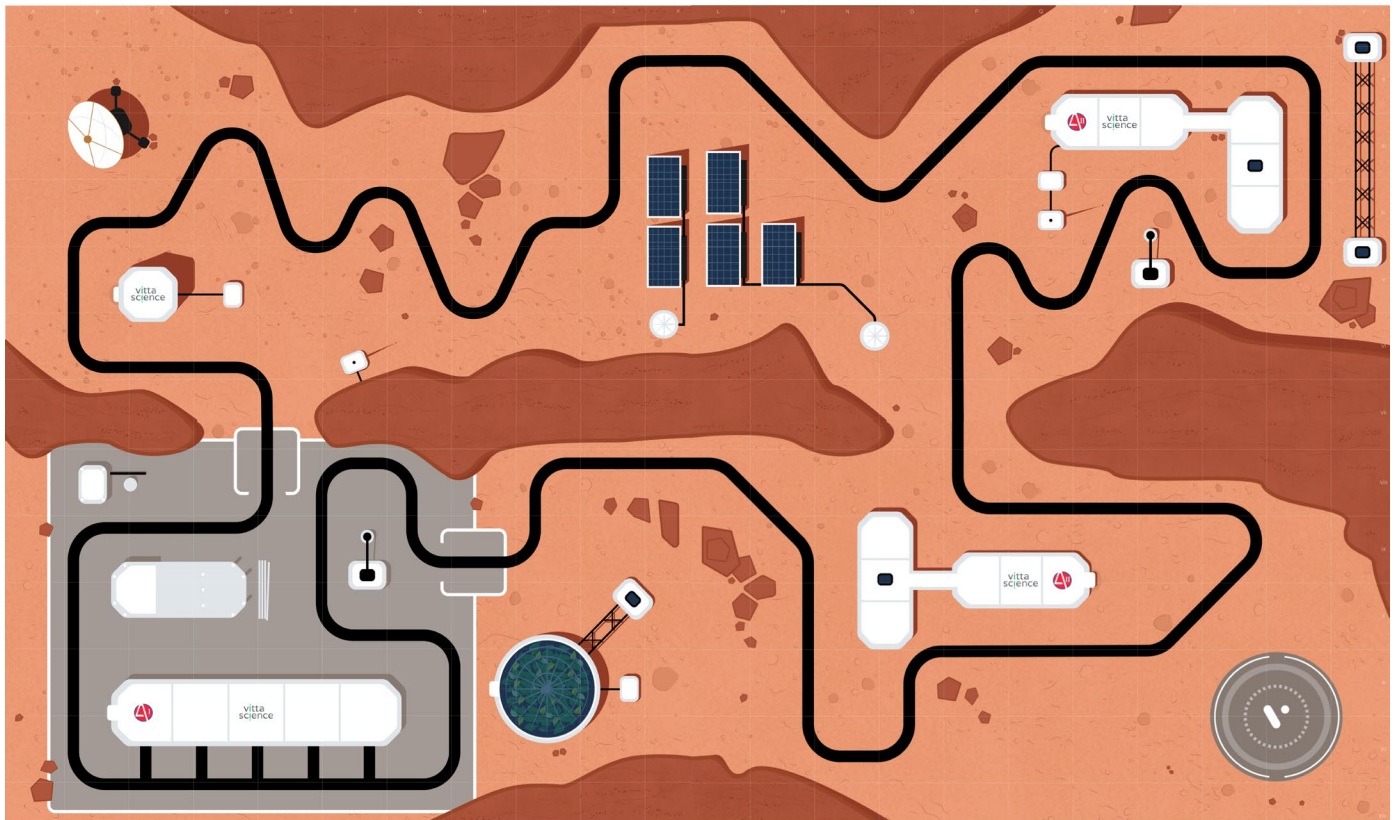
Siamo nel 2041. Il successo della missione su Marte di Ingenuity nel 2021 è ancora nella mente di tutti. Quel robot volante della NASA, schierato a fianco del rover Perseverance, aveva convalidato il concetto di robot collegato ad un centro di pilotaggio su Marte.

A seguito della riuscita di questa missione, molte altre sono state realizzate con successo. Dall'installazione su Marte della base sotterranea Ares I nel 2039, stiamo cercando di colonizzare la superficie.

A questo scopo, nel 2040 è stato installato uno sciame di robot in superficie per pattugliare, analizzare e costruire la futura base Ares II prevista per il 2042. L'ASM (Agenzia Spaziale Marziana) basata sulla Terra pilota l'intero progetto con l'aiuto dei primi coloni marziani sul posto.

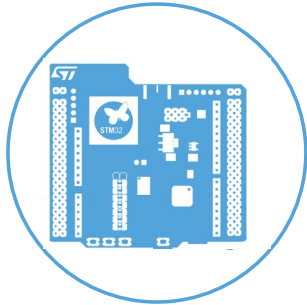
Nel corso delle missioni, avremo a che fare con due dei nostri più eminenti coloni della base Ares I. Infatti, a seguito di un incidente avvenuto sulla base, essi affrontano diverse sfide!

IL TEATRO DELLE OPERAZIONI

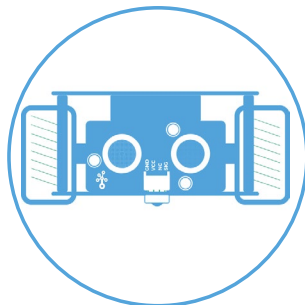


MATERIALE NECESSARIO ALLA COSTRUZIONE E ALL'IMPIEGO DEL KIT "ROBOT MARZIANO"

Contenuto del KIT :



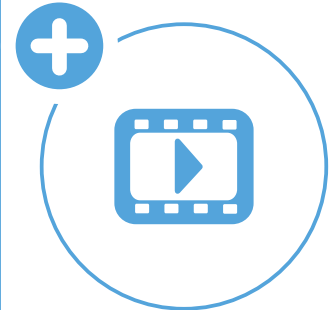
**Scheda ST
NUCLEO-WB55RG**



Robot AlphaBot



Pista per robot



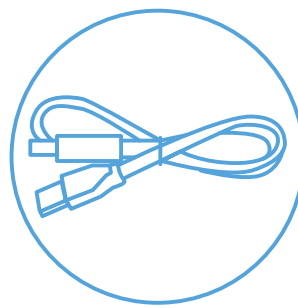
**Risorse digitali
online**



**Sensore di
distanza ST
VL53L0X**



Shield Grove



**Elementi per il
montaggio**



**Prevedere un
computer**

AVVERTENZE PER L'USO DEL KIT



ATTENZIONE !

Presenza di piccoli elementi, non ingerire (rischio di soffocamento).



Età minima

Non adatto a bambini di età inferiore ai 7 anni.

ISTRUZIONI PER UNA CORRETTA RACCOLTA DIFFERENZIATA

La seguente infografica descrive le istruzioni per lo smaltimento dei vari elementi del kit. Per maggiori informazioni, visita il sito www.consignesdetri.fr



SOMMARIO

Qui ASM. Chloé Rophile e Rock Fort, appuntamento al laboratorio di robotica per un inventario del materiale.

Ricevuto ASM.

Ci dirigiamo subito verso il laboratorio.

PAGINA
14

PAGINA
31

Il kit Robot Marziano e l'ambiente di programmazione

- Il kit Robot Marziano
 - Il contenuto del kit
 - La scheda ST-NUCLEO WB55RG
- L'ambiente Vittascienze
 - Creazione di un account
 - L'interfaccia di programmazione
- Panoramica delle parti del robot
- Istruzioni per il montaggio
- Programmazione del robot
- Funzionare del robot

Missione 1 : Scoperta e uso del materiale

- Missione 1-1 : Visualizzare del testo sullo schermo OLED
- Missione 1-2 : Uso del joystick 5
- Missione 1-3 : Uso dei sensori a infrarossi frontali



PAGINA
34

PAGINA
38

Missione 2 : Pilotaggio dei motori

- **Missione 2-1 : Pilotare i motori OLED**
- **Missione 2-2 : Imparare a girare**
- **Missione 2-3 : Seguendo un percorso**

Missione 3 : Prova del rivelatore di ostacoli

- **Missione 3-1 : Scoperta del sensore ad ultrasuoni**
- **Missione 3-2 : Scoperta del Time of Flight (ToF)**
- **Missione 3-3 : Sensori di distanza e motori**

PAGINA
41



PAGINA
44

Missione 4 : Controllo in tempo reale

- Missione 4-1 : Il telecomando
- Missione 4-2 : Test in condizione "Marziana" reale

Missione 5 : Spostamento in modalità casella

- Missione 5-1 : Spostamento in modalità casella
- Missione 5-2 : Individuazione di ostacoli prima dello spostamento

Arès I in ascolto



ARÈS I



ASM

Chloé Rophile e Rock Fort sono stati designati per sostituire il vostro programmatore-robotico attualmente in rianimazione a seguito dell'incidente di depressurizzazione. Dovranno sospendere le loro missioni di informatica e di meccanico. Che si tengano pronti a ricevere le nostre istruzioni. Fine della trasmissione.

Ricevuto. Fine della trasmissione.



ARÈS I

PAGINA
50

PAGINA
59

Missione 6 : Tracker di linea

- Missione 6-1 : Seguire una linea
- Missione 6-2 : Seguire una linea con 3 sensori
- Missione 6-3 : Seguire una linea ed evitare gli ostacoli

Missione 7 : Collaudo in situazione complessa

- Missione 7 : Finalmente la routine !

Laboratorio • 1 a discrezione del supervisore

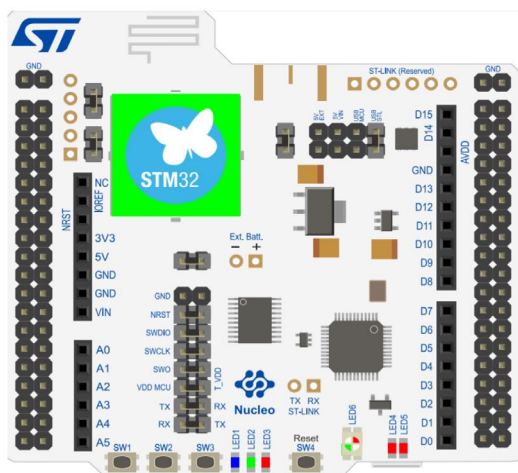
Presentazione del microcontrollore e dell'interfaccia Vittascience

Il kit "pianta connessa" comprende tutti gli elementi necessari per realizzare un montaggio che permette d'irrigare automaticamente una pianta.

Viene fornito con una scheda ST NUCLEO-WB55RG sviluppata da STMicroelectronics. La scheda è dotata di microcontrollore STM32 con Bluetooth low energy BLE.

• Presentazione della scheda NUCLEO-WB55RG 15 min

La seguente illustrazione mostra le entrate e le uscite della scheda NUCLEO-WB55RG. Questa può servire da supporto a diversi montaggi con i componenti forniti.

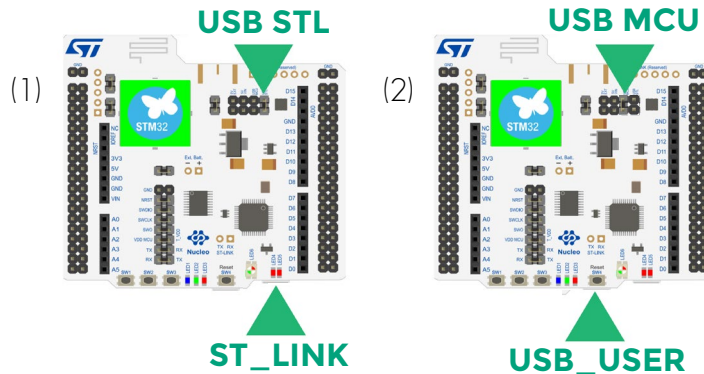


La scheda si programma tramite l'interfaccia Vittascience in MicroPython.

Prima del primo utilizzo, occorre scaricare un firmware per programmare la scheda NUCLEO-WB55RG (vedi riquadro).

RIQUADRO SUL CARICAMENTO DEL PROGRAMMA :

Per programmare la scheda in MicroPython, per codice o blocco dal sito Vittascience, occorre caricare il firmware appropriato.



Ecco le fasi da seguire :

1. Per lampeggiare la scheda, controllare che il ponticello (pezzo metallico circondato da plastica nera) sia posizionato su USB STL nella parte alta della scheda a livello degli alimentatori. In caso contrario, spostare il ponticello su USB STL. Vedi illustrazione (1) .
2. Collegare il cavo USB alla porta ST _ LINK per caricare il firmware. 2 LEDs rosse si accendono.
3. Scaricare il firmware all'indirizzo : <https://stm32python.gitlab.io/fr/docs/Micropython/Telechargement>, poi lasciate questo nella scheda, che è apparsa come una chiavetta USB chiamata "NOD-WB55".
4. Quando lo scaricamento è finito, una LED verde si accende (LED 6 a destra del pulsante "Reset").
5. Scollegare il cavo USB.
6. Spostare il ponticello utilizzato precedentemente (vedi passo n°1) su USB MCU. Vedi illustrazione (2) .
7. Ricollegare il cavo alla porta USB _ USER (oppure l'altra porta USB). Vedi illustrazione (2) .
8. Collegare la scheda al computer, la LED 5 è accesa in rosso. La scheda è quindi alimentata.
9. Utilizzare l'interfaccia Vittascience per programmare la tua scheda :
 - utilizzare preferibilmente un browser Chromium,
 - andare su Vittascience - scheda "Programmare",
 - selezionare l'interfaccia STM32,
 - cliccare "Connetti" e selezionare la scheda,
 - il file main.py caricato viene eseguito continuamente.



Consiglio : Un problema, una domanda ? Siamo qui per risponderti : support@vittascience.com

• Programmazione della scheda a discrezione del supervisore

In questa sezione viene descritto il funzionamento dell'interfaccia di programmazione online [Vittascienze.com](https://vittascienze.com).

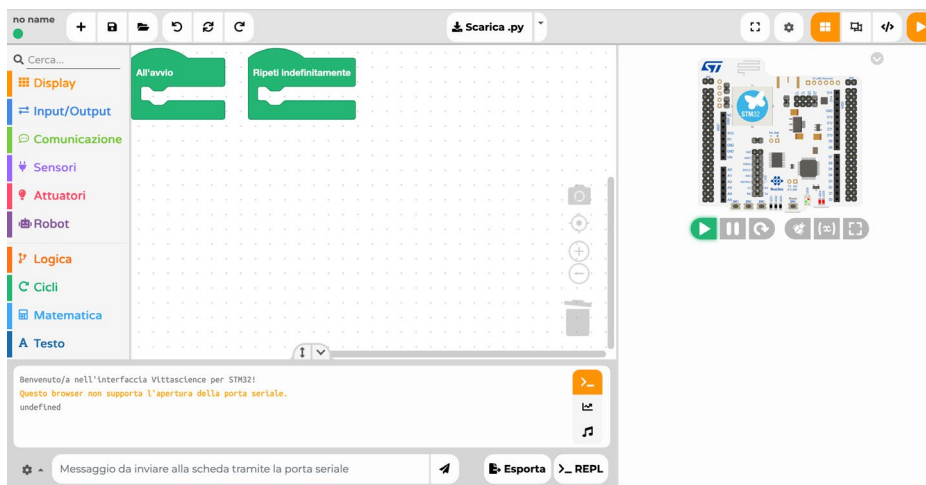
È anche possibile programmare la scheda utilizzando il software Arduino (linguaggio C++). Tutorial per questo software sono disponibili nella libreria delle risorse del sito [Vittascienze.com](https://vittascienze.com).

1 • Creazione di un account

Innanzitutto, ti consigliamo di creare un account sul nostro sito. Non è necessario per usufruire del kit, ma ti permetterà di salvare e condividere i tuoi programmi, risorse e feedback.

Per questo, visita il nostro sito [Vittascienze.com](https://vittascienze.com) e clicca sull'icona verde in alto a destra per registrarti.

2 • L'interfaccia




L'interfaccia permette di programmare in blocco con una trascrizione in parallelo in linguaggio Python.



Attenzione : La scheda ST NUCLEO-WB55RG è necessaria per eseguire il programma una volta i sensori posizionati.


Trovi sul sito [Vittascienze.com](https://vittascienze.com) delle risorse e dei programmi per imparare a programmare con la scheda.


Selezione della porta :  collegando la scheda al computer, l'interfaccia rileverà automaticamente su quale porta è collegata. Il menu a discesa permette di selezionare la porta giusta se più schede sono collegate al computer.

Trasferire il programma sulla scheda :  il codice viene subito eseguito sulla scheda al termine del trasferimento.

Annullare o ripristinare :  le azioni precedenti o seguenti.

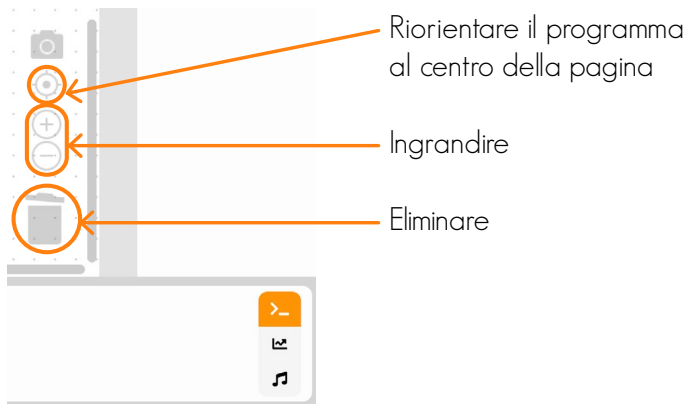
Iniziare un nuovo progetto :  per iniziare un nuovo progetto vergine, cliccare su questo pulsante.

Salvare il progetto :  per registrare il progetto per salvarlo, cliccare su questo pulsante. Per chi dispone di un account, è possibile condividere il programma con la comunità.

Aprire un progetto esistente :  se si desidera aprire programmi già realizzati o far lavorare gli studenti su una trama che hai creato, possono accedervi cliccando su questo pulsante.

Codificare in Python :  se vuoi codificare direttamente in Python.

Accesso rapido :

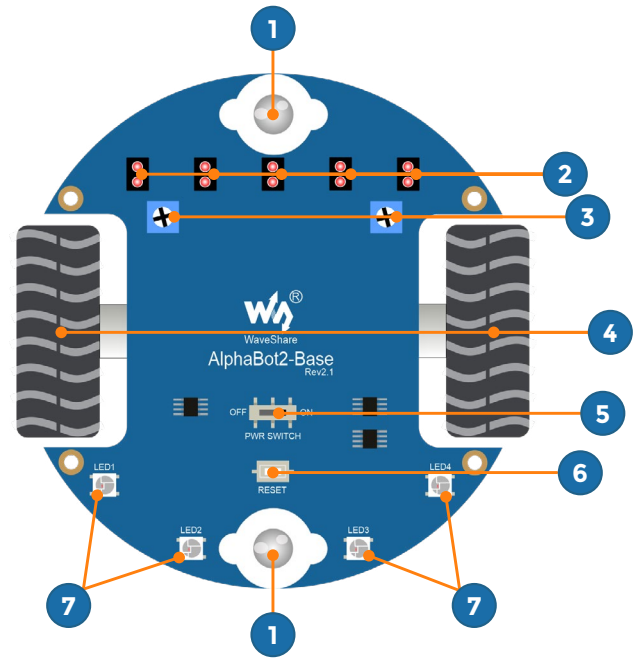
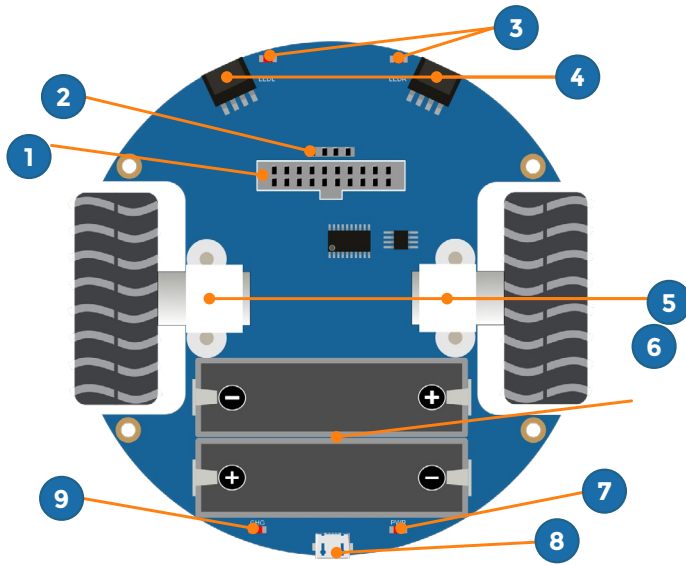


Consiglio : Questa interfaccia di programmazione è progettata per essere molto semplice da usare, non esitare a testarla e proporla ai tuoi studenti.

Panoramica dell'AlphaBot2-Base

• Figura 1 : Base inferiore, lato anteriore

• Figura 2 : Base inferiore, retro

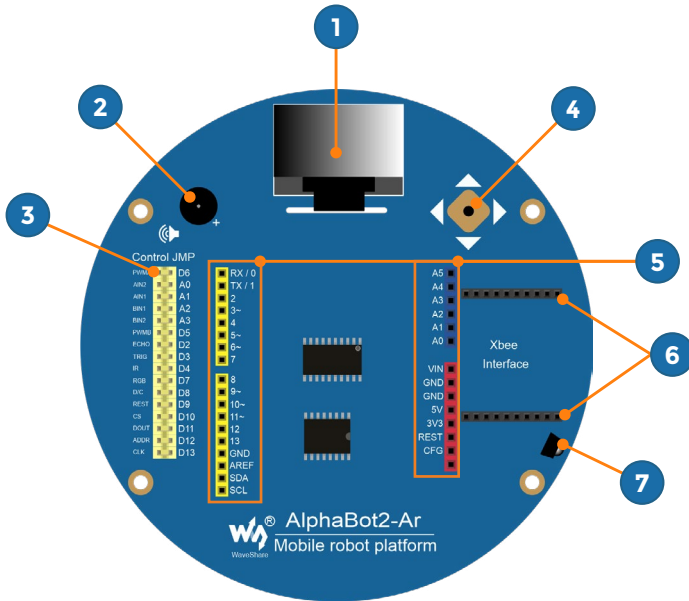


Fonte : [AlphaBot2-Ar - Waveshare Wiki](#)

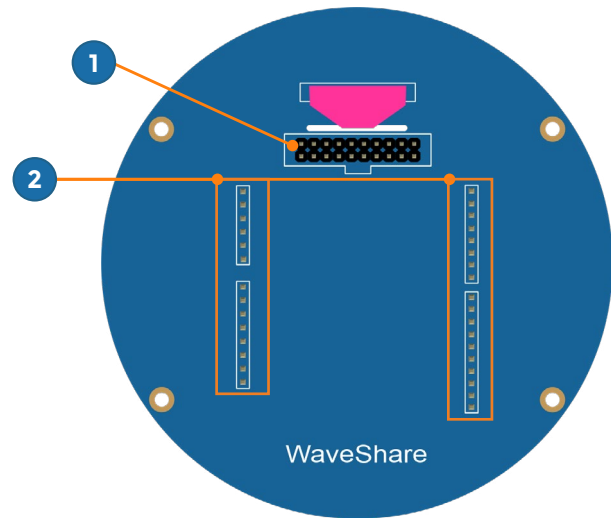
Fonte : [AlphaBot2-Ar - Waveshare Wiki](#)

- 1 • Interfaccia di controllo dell'AlphaBot2 (femmina)
- 2 • Interfaccia del modulo ultrasonico
- 3 • Indicatori di aggiramento degli ostacoli
- 4 • Sensore fotoelettrico a infrarossi riflettente, per evitare gli ostacoli
- 5 • Tasso di riduzione del micromotore a ingranaggi 1:30, 6V/600RPM
- 6 • Portapile per pile ricaricabili Li-ion 14500. *Attenzione, l'orientamento di questo contenitore può essere invertito secondo le revisioni del telaio.*
- 7 • Indicatore di alimentazione
- 8 • Porta USB 5V per la ricarica della batteria
- 9 • Indicatore di carica della batteria

- 1 • Ruota omnidirezionale
- 2 • Sensore fotoelettrico a infrarossi riflettente, per il tracciamento di linea
- 3 • Potenzimetri per regolare la sensibilità del sistema anticollisioni. *Dovrai regolarli con un cacciavite in modo che i LED anticollisione (figura 1, punto 4) funzionino.*
- 4 • Ruote in gomma diametro 42mm, larghezza 19mm
- 5 • Interruttore di alimentazione
- 6 • Interruttore di ripristino
- 7 • LEDs RGB indirizzabili.

• **Figura 3 : Base superiore, lato anteriore**

Fonte : [AlphaBot2-Ar - Waveshare Wiki](#)

• **Figura 4 : Base superiore, retro**

Fonte : [AlphaBot2-Ar - Waveshare Wiki](#)

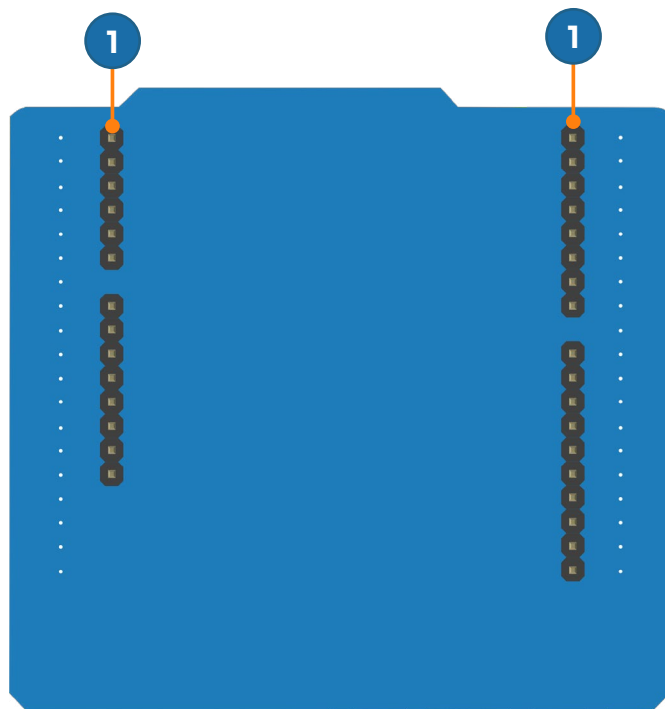
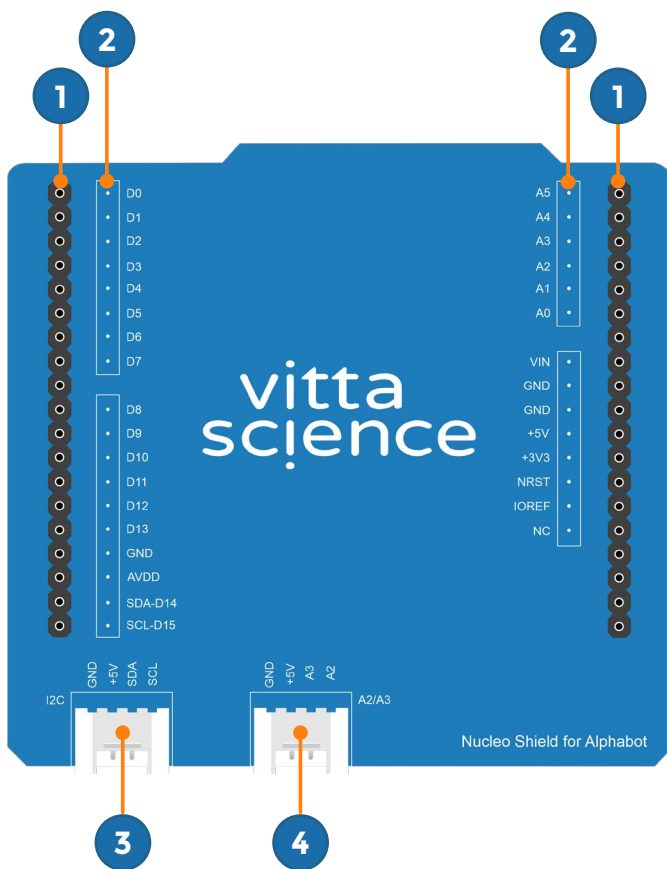
- 1 • Schermo OLED 128x64
- 2 • Cialino
- 3 • Connettori per periferiche Arduino
- 4 • Joystick
- 5 • Connettore di estensione Arduino (femmina)
- 6 • Connettore Xbee (femmina), non utilizzato nel nostro kit.
- 7 • Ricevitore IR (per il telecomando)

- 1 • Interfaccia di controllo AlphaBot2 (maschio)
- 2 • Connettore di estensione Arduino (maschio)

Panoramica del Nucleo Shield per Alphabot

• **Figura 5 : Nucleo Shield per Alphabot, lato anteriore**

• **Figura 6 : Nucleo Shield per Alphabot, retro**



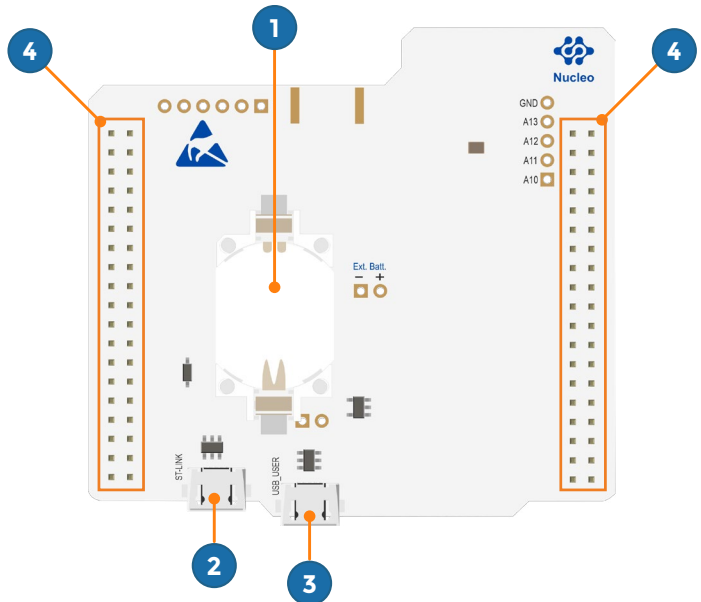
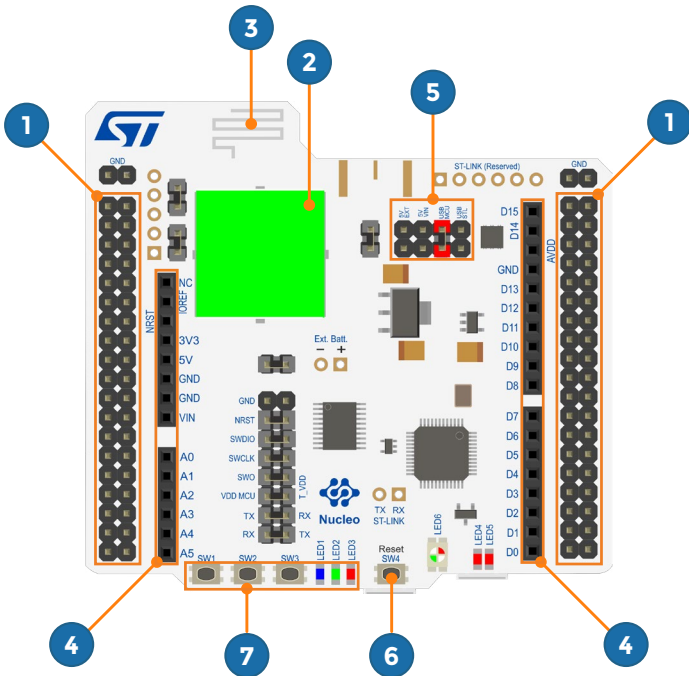
- 1 • Connettore femmina per Nucleo
- 2 • Etichette delle estensioni Arduino
- 3 • Connettore Grove al supporto I2C
- 4 • Connettore Grove all'ingresso analogico

- 1 • Connettore di estensione Arduino (maschio)

Panoramica della scheda Nucleo-WB55

• Figura 7 : Nucleo-WB55 lato anteriore

• Figura 8 : Nucleo-WB55 retro

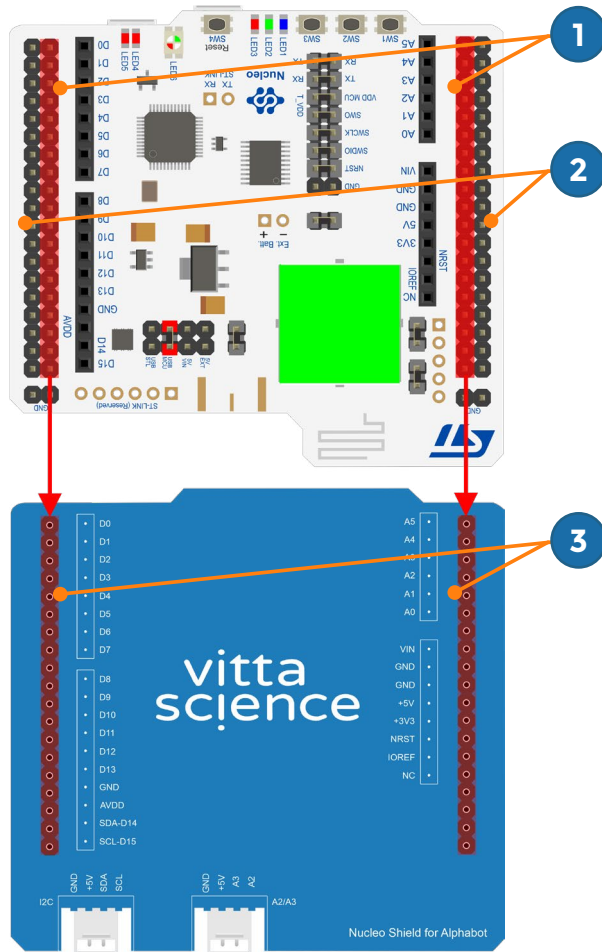


- 1 • Connettore per estensione ST-Morpho (maschio)
- 2 • Chip STM32 WB55 integrante la radio Bluetooth a bassa energia
- 3 • Antenna Bluetooth a bassa energia
- 4 • Connettore di estensione Arduino (femmina)
- 5 • Connettore di alimentazione e di programmazione (maschio)
- 6 • Interruttore di ripristino
- 7 • Interruttori utente (3) e LED utente (3)

- 1 • Presa per pila CR2023 (a seconda della versione)
- 2 • Connettore Micro-USB a ST-LINK (programmazione del firmware)
- 3 • Connettore Micro-USB a USB _ USER (collegamento a MicroPython REPL)
- 4 • Connettore ST-MORPHO (maschio)

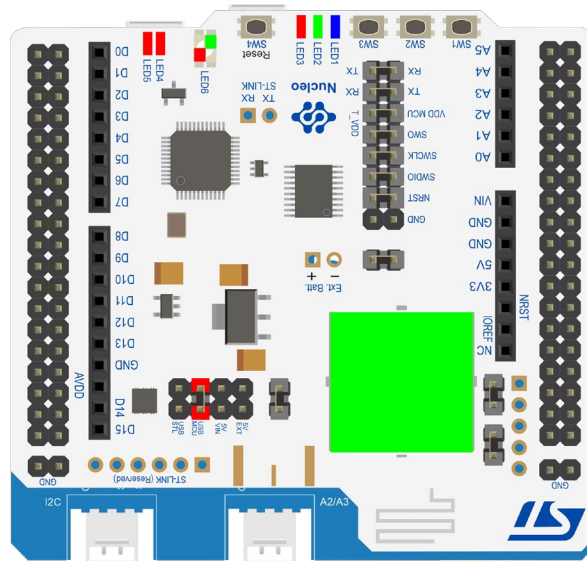
Istruzioni per il montaggio 🔒 A discrezione del supervisore Collegamento del Nucleo-WB55 allo Shield Nucleo per Alphabet

• Figura 9 : Nucleo-WB55 e shield Nucleo pour il collegamento dei connettori Alphabet



- 1 • Colonne interne del connettore di estensione ST-Morpho
- 2 • Colonne esterne del connettore di espansione ST-Morpho
- 3 • Nucleo - to - Arduino connettore

• **Figura 10 : Scheda Nucleo-WB55 collegato allo shield Nucleo-to-AlphaBot2 (vista dall'alto)**



• **Innanzitutto, collegare la scheda Nucleo-WB55 sulla parte superiore del Nucleo Shield per Alfabot.**

A tal fine, le due **colonne interne** del connettore di estensione ST-Morpho Nucleo-WB55 devono inserirsi nel connettore del ponte Nucleo-to-Arduino. Come illustrato nella figura 9.

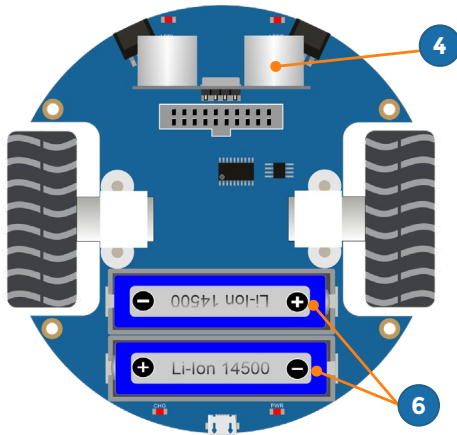
E così facendo :

- **Prestare attenzione** al corretto orientamento della Nucleo-WB55 ;
- **Assicurarsi** che i connettori siano allineati correttamente e che tutti i pin siano completamente premuti ;
- **Si noti** che una volta completata l'operazione, i pin delle colonne esterne dei connettori ST-Morpho devono galleggiare parallelamente ai ponti Nucleo - to - Arduino dello shield.

Se hai eseguito correttamente questa operazione, il tuo sistema dovrebbe assomigliare alla Figura 10.

Installazione della base Alphabot

• **Figura 11 : Installazione della base inferiore di Alphabot (vista dall'alto)**



- 1 • Sensore a ultrasuoni collegato
 - 2 • Batterie ricaricabili Li-ion 14500 collegate.
- Attenzione, le polarità potrebbero essere invertite sul tuo robot!

• Prepara la base inferiore di Alphabot

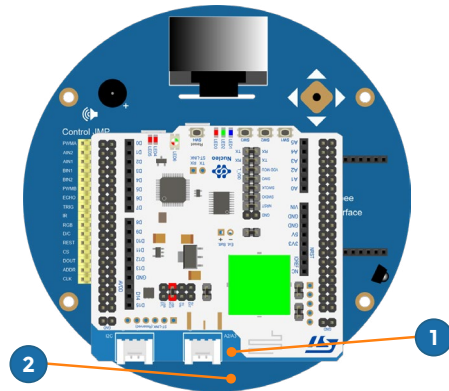
Per fare ciò, collegare le due batterie ricaricabili Li-ion 14500 al supporto delle batterie e il sensore ad ultrasuoni nel suo connettore. Come illustrato sulla figura 11.

E così facendo :

- **Assicurarsi** di utilizzare il tipo di pile corretto, ovvero le pile ricaricabili Li-ion 14500.

In particolare, anche se si adattano perfettamente agli alloggiamenti del telaio, le pile AA NON CONSENTONO al robot di funzionare.

• **Figura 12 : Installazione della base superiore di Alphabot (vista dall'alto)**



- 1 • Shield Nucleo per Alphabot collegato al connettore di estensione Arduino, sulla parte alta della base superiore di Alphabot.
- 2 • Base superiore di Alphabot (vista dal basso)

• Prepara la base superiore di Alphabot

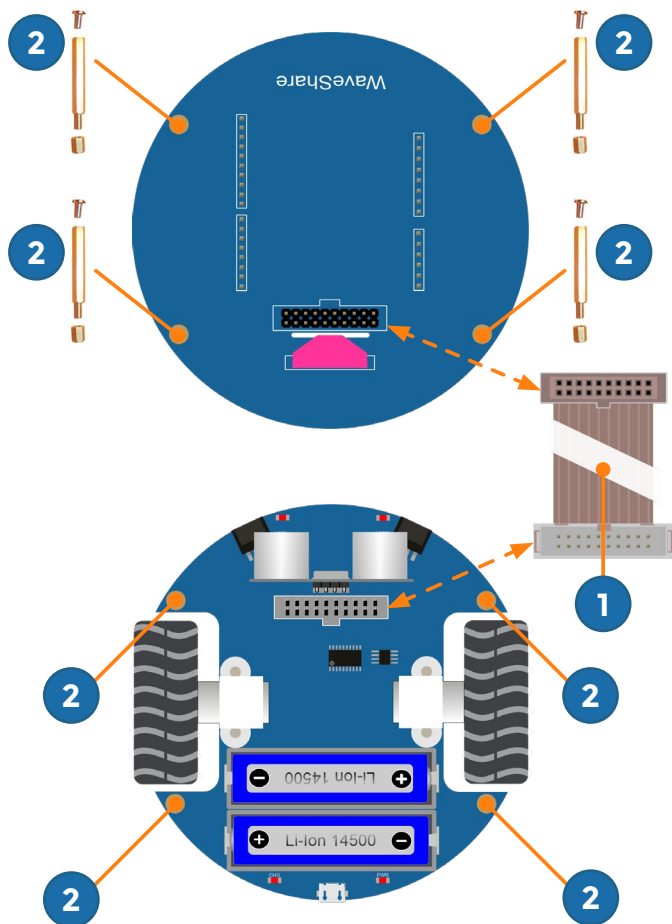
Per fare ciò, collegare i connettori Arduino femmine sul retro dello shield Nucleo per Alphabot al connettore di estensione Arduino posto sulla parte anteriore della base superiore di Alphabot.

E così facendo :

- **Assicurarsi** che i connettori siano correttamente allineati. Se hai eseguito correttamente questa operazione, il risultato, visto dall'alto, dovrebbe assomigliare alla Figura 12.

Collegamento della base inferiore alla base superiore di Alhabot

• **Figura 13 : Collegare la base inferiore alla base superiore**



- 1 • Scheda dei connettori flessibile.
- 2 • Fori e viti di distanziamento
- 3 • Fori per le viti di distanziamento

• **Collegare i livelli di Alhabot**

Per finire,

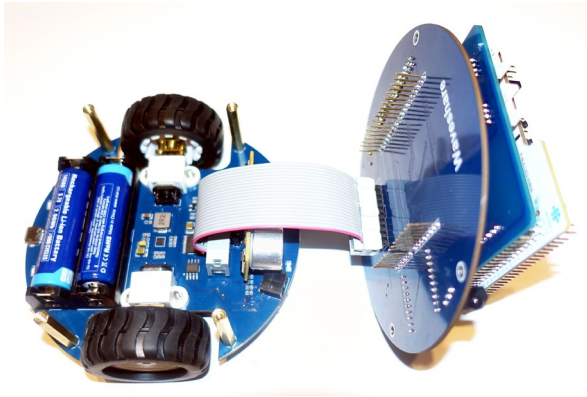
- 1 • Collegare l'estremità femmina della scheda di connessione flessibile all'interfaccia di controllo di AlphaBot2, alla parte posteriore della base superiore.
- 2 • Collegare l'estremità maschio della scheda di connessione flessibile all'interfaccia di controllo di AlphaBot2 sul lato anteriore della base inferiore.
- 3 • Completare l'assemblaggio delle basi superiore e inferiore posizionando le 4 viti di distanziamento nei fori corrispondenti.

E così facendo :

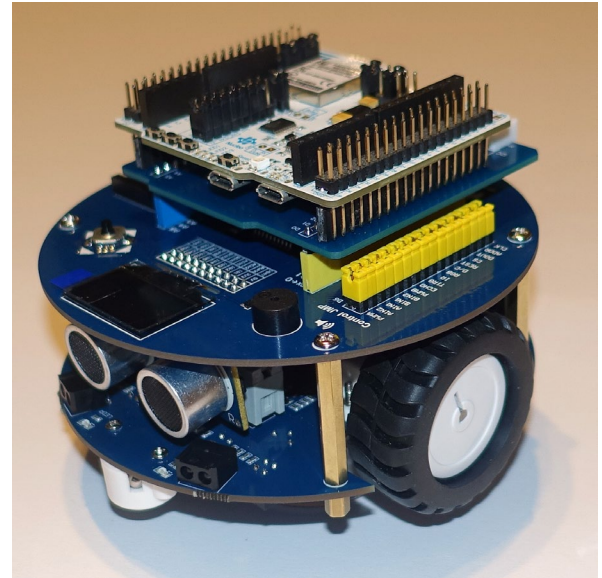
- **Assicurarsi** di tirare fermamente le estremità della scheda flessibile all'interno dei connettori delle basi superiore e inferiore.

La figura 13 illustra questa fase.

Foto del robot marziano assemblato



• Foto 1 : Collegamento della base inferiore di Alphabot alla base superiore.



• Foto 2 : Robot marziano assemblato.

• Fotos del robot assemblato

Le foto 1 e 2 ti daranno la conferma visiva che non hai commesso errori seguendo le precedenti istruzioni di montaggio precedenti.

Ora è il momento di condividere gli ultimi punti tecnici per utilizzare correttamente il robot!

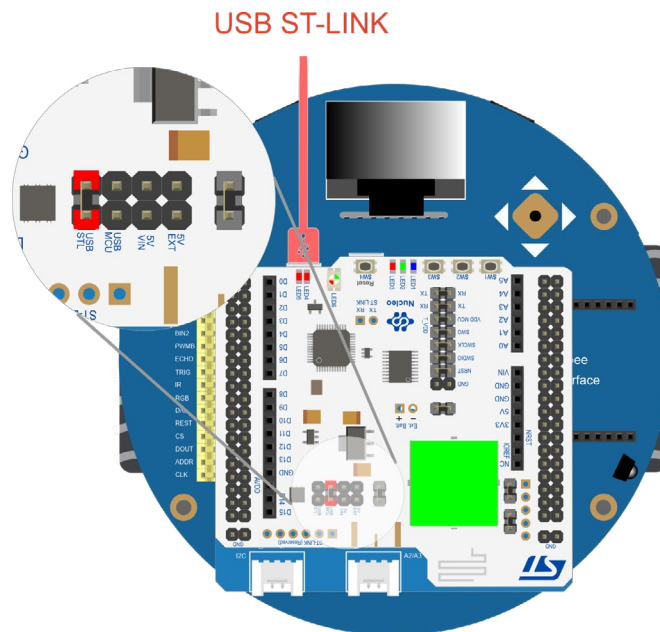
Programmazione del robot marziano

🕒 A discrezione del supervisore

Configurazione dell'aggiornamento del firmware di MicroPython

- 1 • Posizionare l'interruttore di accensione situato sotto il robot su «OFF».
- 2 • Posizionare il ponticello indicato in rosso nella figura 14 in posizione «USB ST-LINK».
- 3 • Collegare il Nucleo-WB55 al computer tramite il connettore USB _ STL.
- 4 • Dal computer, trascinare il nuovo firmware nel disco USB virtuale associato alla Nucleo-WB55.
- 5 • Attendere fino al completamento dell'operazione di copia.

Affinché il robot funzioni correttamente, è necessario che abbia il firmware in [una versione 1.17 o successiva](#). Si raccomanda di scaricare l'ultima versione convalidata del firmware [esclusivamente dal sito di Vittascience](#).

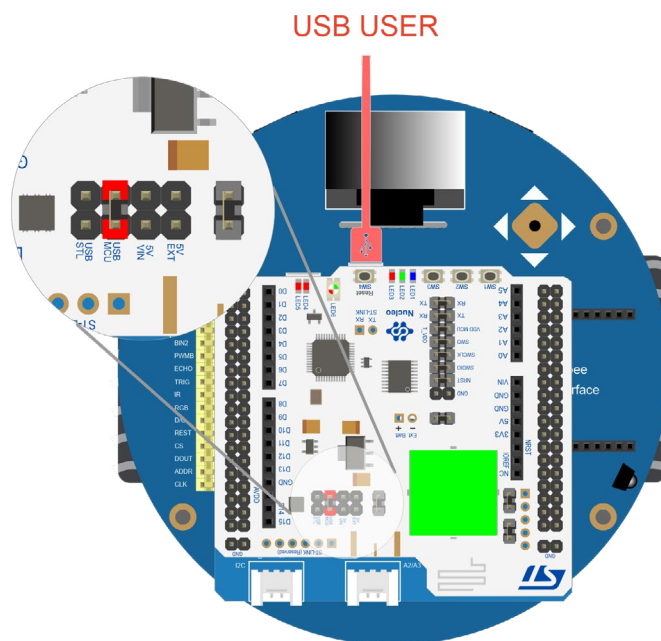


Configurazione per la programmazione in MicroPython

- 1 • Posizionare l'interruttore sotto il robot su «OFF»
- 2 • Posizionare il ponticello (in rosso) sulla figura sopra sulla posizione «USB MCU».
- 3 • Collegare la scheda al computer utilizzando il connettore «USB USER» della scheda Nucleo-WB55.
- 4 • Collegare il cavo USB al computer e avviare l'interfaccia di programmazione di Vittascienze:

vittascienze.com/stm32

Ora puoi connetterti all'interfaccia Vittascienze e scaricare i tuoi programmi nel Nucleo-WB55 che guida il robot.

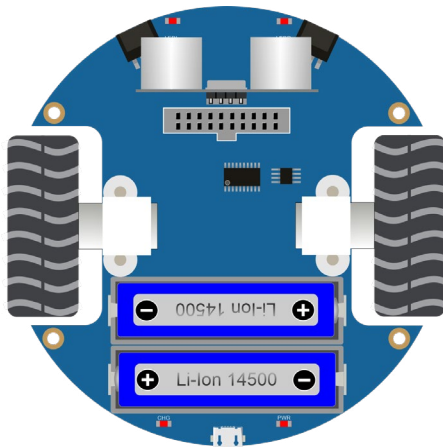


Fare funzionare il Mars Rover

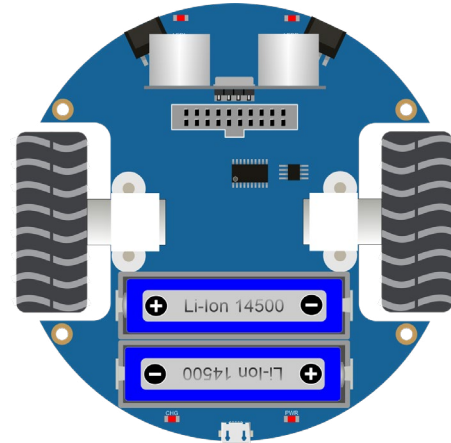
🕒 A discrezione del supervisore

Caricare le batterie del robot

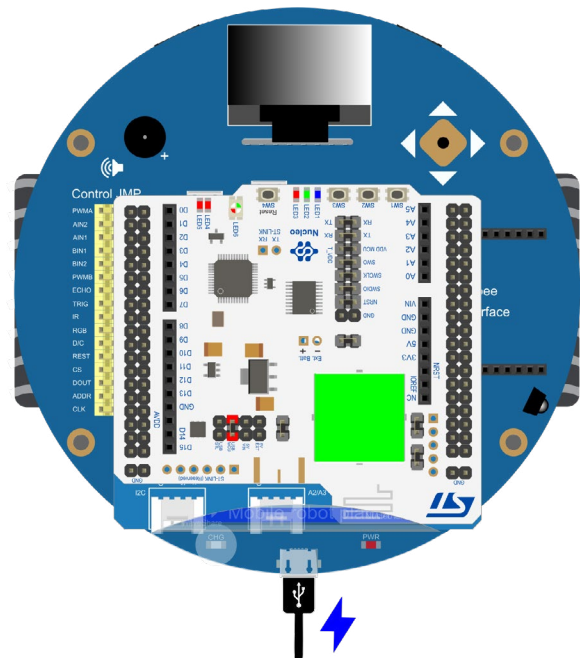
- 1 • Se l'orientamento delle batterie del vostro robot corrisponde alla figura 16-a, posizionare l'interruttore di alimentazione sotto il robot su "OFF".
- 2 • Se l'orientamento delle batterie del vostro robot corrisponde alla figura 16-b, posizionare l'interruttore di alimentazione sotto il robot su "ON".
- 3 • Collegare un singolo cavo USB del robot alla porta USB 5V sulla base dell'AlphaBot2.
- 4 • Il LED CHG dovrebbe rimanere acceso fino a quando le batterie sono completamente cariche. Se il LED lampeggia rapidamente, controlla che il cavo USB e le batterie siano collegati correttamente.
- 5 • Quando le batterie sono piene, la luce CHG è spenta. Il tuo robot è pronto a avviare, basta girare l'interruttore di alimentazione sotto il robot su "ON".



• Figura 16-a



• Figura 16-b



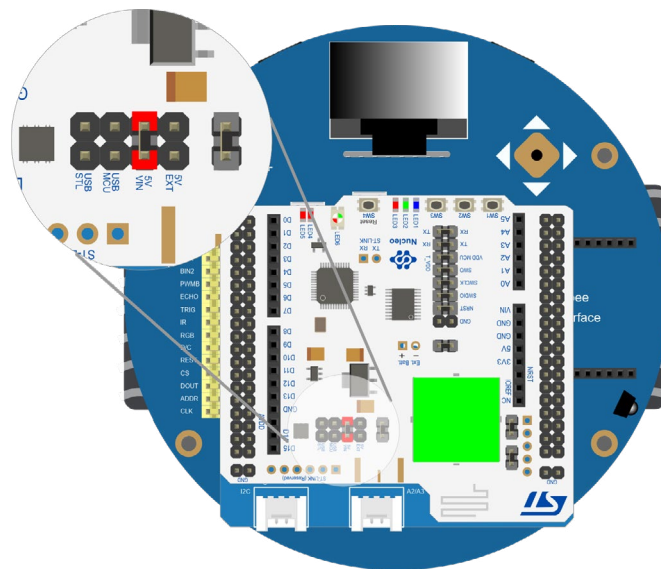
• Figura 17

Avviare il robot

- 1 • Scollegate tutti i cavi USB collegati al vostro robot.
- 2 • Impostare il ponticello (in rosso) nella figura sopra sulla posizione "5V VIN". Ora la scheda Nucleo-WB55 trarrà la sua potenza dalle due batterie della base Alphabot.
- 3 • Girare l'interruttore di alimentazione sul fondo del robot su "ON", il LED PWR sulla sua base dovrebbe illuminarsi.

Se le batterie di Alphabot sono sufficientemente cariche, il LED PWR sulla base del robot si illuminerà e il robot inizierà a eseguire l'ultimo programma che avete caricato sul Nucleo-WB55 utilizzando l'interfaccia di programmazione Vittascience.

Se il comportamento del vostro robot non è quello che vi aspettavate, controllate il vostro programma e assicuratevi che le batterie siano sufficientemente cariche.



Missione • 1 🕒 a discrezione del supervisore

Scoperta e uso del materiale

Scoperta della scheda del controllore del futuro mini rover (missione molto semplice per fare conoscere appieno il trasferimento di programma e l'interfaccia della piattaforma Vittascience).

• Missione 1-1 : Visualizzare un testo sullo schermo OLED

Per iniziare questa prima missione, visualizzeremo il testo "Marte" sullo schermo OLED del robot.



Il vostro programma di lavoro sarà molto impegnativo per farvi possedere pienamente tutte le tecnologie utili a questo progetto. Dovrete fare prova d'iniziativa per sviluppare nuove competenze che non erano previste nella vostra formazione iniziale per venire su Marte. Cominciamo con qualcosa di semplice. Vi troverete ad affrontare alcuni piccoli esercizi di programmazione per familiarizzarvi con le procedure di utilizzo della scheda microcontrollore presente in tutti i robot radioguidati della superficie.



Mamma mia ! Dobbiamo maneggiare queste cosette ?? Ma le schiaccerò con le mie ditaccia !

ROCK

Ma dai ! Sono più di 30 anni che usiamo dei microcontrollori nei nostri rover e funziona ancora! Immagina, ce n'erano già su Perseveranza! È roba robusta !

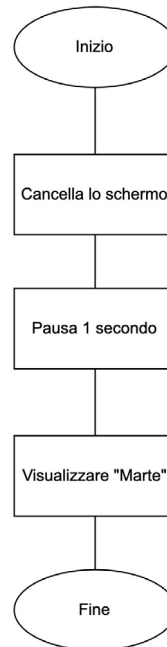


CHLOE



Procedura in corso di spedizione. Passo e chiudo.

Ecco nell'ordine l'algoritmo, poi il codice di programmazione a blocchi e infine il codice Python.



```

1 import machine
2 from stm32_ssd1306 import SSD1306, SSD1306_I2C
3 import utime
4
5 oled = SSD1306_I2C(128, 64, machine.I2C(1))
6
7 while True:
8     oled.text('Mars ', 0, 0)
9     oled.show()
10    utime.sleep(1)
11    oled.fill(0)
12    oled.show()
  
```

Nell'interfaccia di programmazione di Vittascience è possibile scegliere se programmare in Python o in blocco. In un primo momento e a seconda del livello, si consiglia di utilizzare il codice a blocchi che è più semplice per iniziare.

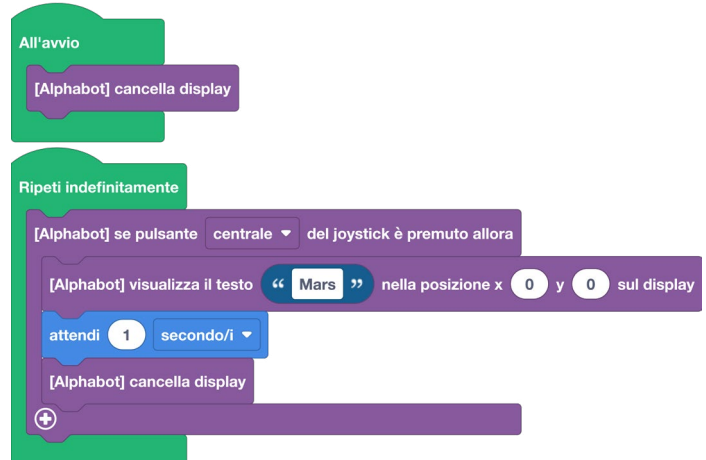
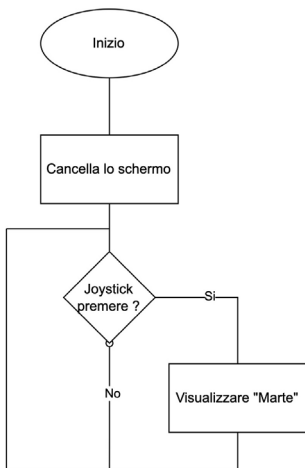
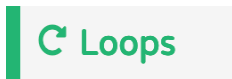
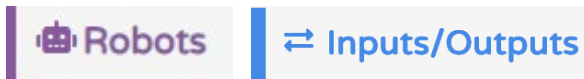
I blocchi giusti sono nel menù a sinistra, per iniziare solo i due seguenti saranno utili.



• Missione 1-2 : Uso del joystick

Ora che abbiamo capito come visualizzare un testo con la scheda, interagirremo con lo schermo usando in un primo momento il joystick.

Manterremo la visualizzazione di "Marte" come codice di base, ma ora verrà visualizzata solo dopo aver premuto il joystick del robot. Per questo avremo bisogno di nuovi blocchi che si trovano nei seguenti menu:



```

1 import machine
2 from stm32_ssd1306 import SSD1306, SSD1306_I2C
3 from stm32_alphabot_v2 import AlphaBot_v2
4 import utime
5
6 oled = SSD1306_I2C(128, 64, machine.I2C(1))
7 alphabot = AlphaBot_v2()
8
9 oled.fill(0)
10 oled.show()
11
12 while True:
13     joystickButton = alphabot.getJoystickValue()
14     if joystickButton == "center":
15         oled.text('Mars ', 0, 0)
16         oled.show()
17         utime.sleep(1)
18         oled.fill(0)
19         oled.show()
  
```


• Missione 1-3 : Uso dei sensori ad infrarossi frontali

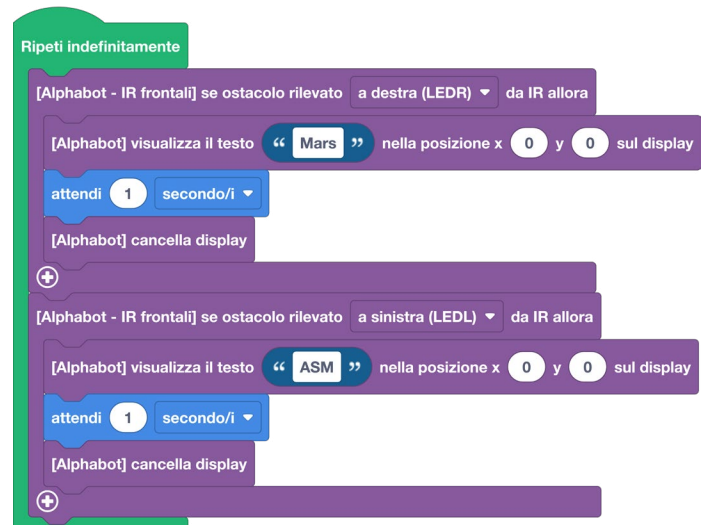
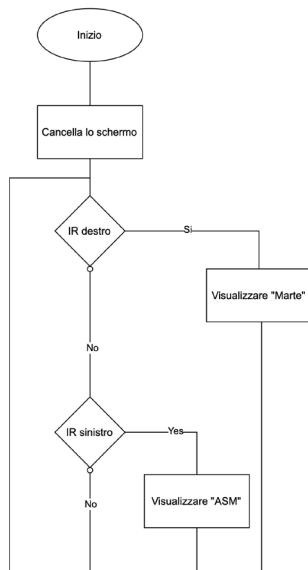
Al posto del joystick, che è solo un sensore di pressione basico (premere o non premere) useremo i sensori ad infrarossi frontali del robot.

Per questa missione, completeremo il codice precedente con la seguente funzione se passo la mano davanti al sensore a destra, il display OLED mostra Marte per 1 secondo. Se passo la mano davanti al sensore a destra, allora il display OLED mostra "Marte" per 1 secondo.

Se passo la mano davanti al sensore di sinistra, allora visualizza "ASM". » (Di nuovo l'informazione ricevuta sarà basica «capito o no»).

Ecco nell'ordine l'algorigramma, poi il codice di programmazione a blocchi e infine il codice Python.

Nel codice, possiamo vedere che c'è un blocco «cancella schermo» nel ciclo all'avvio, questo permette di assicurarsi che lo schermo sia effettivamente vuoto all'avvio del robot.



```

1 import machine
2 from stm32_ssd1306 import SSD1306, SSD1306_I2C
3 from stm32_alphabot_v2 import AlphaBot_v2
4 import utime
5
6 oled = SSD1306_I2C(128, 64, machine.I2C(1))
7 alphabot = AlphaBot_v2()
8
9 oled.fill(0)
10 oled.show()
11
12 while True:
13     detection = alphabot.readInfrared()
14     if detection == alphabot.RIGHT_OBSTACLE:
15         oled.text('Mars ', 0, 0)
16         oled.show()
17         utime.sleep(1)
18         oled.fill(0)
19         oled.show()
20     detection = alphabot.readInfrared()
21     if detection == alphabot.LEFT_OBSTACLE:
22         oled.text('ASM', 0, 0)
23         oled.show()
24         utime.sleep(1)
25         oled.fill(0)
26         oled.show()
  
```

Missione • 2 🔧 a discrezione del supervisore

Pilotaggio dei motori

• Missione 2-1 : Pilotare i motori

È finalmente giunto il momento di far muovere il nostro piccolo robot. Per ora ci accontenteremo di semplici movimenti avanti/girare.



Qui ASM. I robot costruttori di superficie sono attualmente fermi e i robot sono in posizioni sconosciute. Ora imparerete a spostare i robot e a riposizionarli in modo che possano essere rimpatriati per una revisione.



Figo, dov'è il joystick ?

Secondo me non è come con i tuoi trapani che premi solo un pulsante !!



```

Ripeti indefinitamente
  Commento Etichetta 1
  [Alphabot] controlla il motore destra direzione velocità 100 (%)
  [Alphabot] controlla il motore sinistra direzione velocità 100 (%)
  Commento Etichetta 2
  [Alphabot] controlla il robot vai avanti velocità 100 (%)
  [Alphabot] controlla il robot vai indietro velocità 100 (%)
  Commento Etichetta 3
  [Alphabot] arresta il motore destra
  [Alphabot] arresta il motore sinistra
  [Alphabot] arresta il motore destro & sinistro
  Commento Etichetta 4
  [Alphabot] gira a destra velocità 50 (%)
  [Alphabot] gira a sinistra velocità 50 (%)
  
```

Blocco 1 : Permette di controllare ogni motore destro o sinistro in modo indipendente, di controllare il senso di rotazione e infine di controllare la velocità di rotazione.

Blocco 2 : Consente di controllare contemporaneamente entrambi i motori destro e sinistro tanto per il senso di rotazione quanto per la velocità di rotazione.

Blocco 3 : Consente di spegnere i motori in modo indipendente o entrambi allo stesso tempo.

Blocco 4 : Abilita il motore a girare a destra e a sinistra e a controllare la potenza del motore.

Ci sono diversi modi per controllare le ruote del robot in base ai blocchi di codice scelti che sono tre :

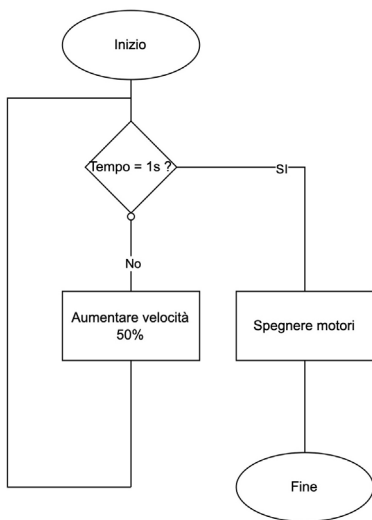
Per il nostro primo programma ci accontenteremo di far avanzare il robot in avanti per 1 secondo.

Per questo abbiamo bisogno di blocchi che si trovano in :

Robots

Inputs/Outputs

Ecco nell'ordine l'algoritmo, poi il codice di programmazione a blocchi e infine il codice Python.



```

1 import machine
2 from stm32_alphabot_v2 import AlphaBot_v2
3 import utime
4
5 alphabot = AlphaBot_v2()
6
7 alphabot.moveForward(50)
8 utime.sleep(1)
9 alphabot.stop()
10
11 while True:
12     pass
  
```

• Missione 2-2 : Imparare a girare

Un robot che va dritto, è bello, ma è pericoloso, dobbiamo insegnargli a girare. Per questo useremo il controllo indipendente di ogni motore.

Abbiamo due scelte per girare: il metodo carro armato o il metodo automobile. Il carro armato fa ruotare i cingoli in senso opposto da ogni lato, permettendo così di ruotare sul posto. L'auto con il sistema differenziale di solito ha una ruota che gira più velocemente rispetto all'altra, il che permette di girare.

Ogni tecnica ha i suoi vantaggi e svantaggi a seconda della situazione. La tecnica con direzioni opposte permette di girare più velocemente ma a scapito della velocità globale di spostamento. E naturalmente per l'altro metodo, è il contrario.

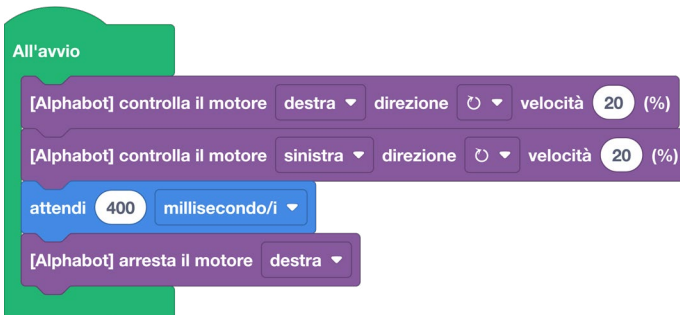
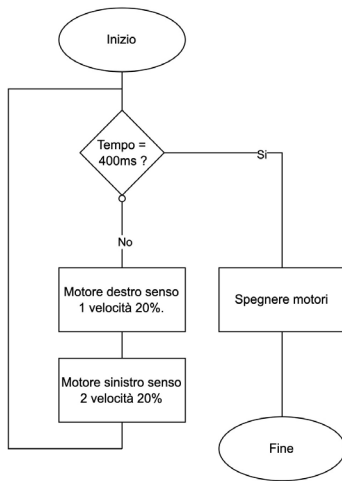


Nel nostro caso ci muoveremo in un ambiente ostile (Marte) dove quindi la prudenza e la manovrabilità sono più importanti quindi inizieremo con il metodo carro armato con una curva a 90°.

Per questo avremo bisogno di un blocco che si trova in :



Ecco nell'ordine l'algorigramma, poi il codice di programmazione a blocchi e infine il codice Python.



```

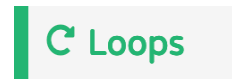
1 import machine
2 from stm32_alphabot_v2 import AlphaBot_v2
3 import utime
4
5 alphabot = AlphaBot_v2()
6
7 alphabot.setMotors(right=20)
8 alphabot.setMotors(left=-20)
9 utime.sleep_ms(400)
10 alphabot.stop()
11
12 while True:
13     pass
  
```

Il programma non sarà perfetto, occorrerà regolare il tempo al fine di ottenere un angolo di 90°. È possibile farlo per tentativi ed errori o utilizzare una tabella di proporzionalità. Non esitate a ridurre la velocità per controllare meglio la direzione. Ora controlliamo perfettamente il nostro robot e possiamo muoverci dove vogliamo.

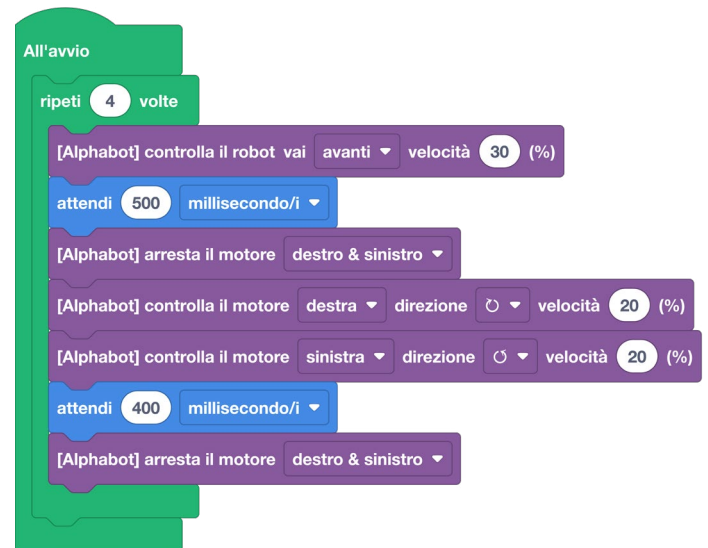
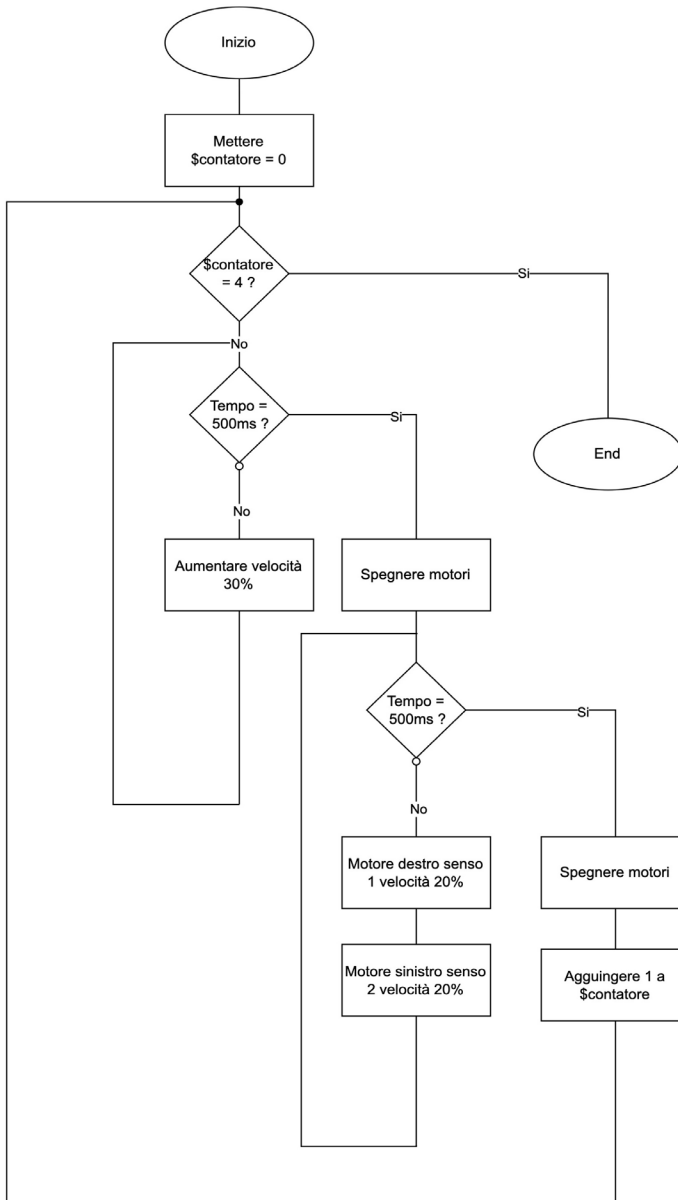
• Missione 2-3 : Seguendo un percorso

Sappiamo avanzare e indietreggiare, ma anche girare a 90°. Mostra la tua abilità eseguendo un quadrato.

Per questo abbiamo bisogno di blocchi che si trovano in :



! Attenzione : L'algorigramma diventa più complicato in scrittura a causa del ciclo che conta quante volte si è eseguita una serie di azioni. Si vedrà quindi apparire una variabile contatore.



```

1 import machine
2 from stm32_alphabot_v2 import AlphaBot_v2
3 import utime
4
5 alphabot = AlphaBot_v2()
6
7 for count in range(4):
8     alphabot.moveForward(30)
9     utime.sleep_ms(500)
10    alphabot.stop()
11    alphabot.setMotors(right=20)
12    alphabot.setMotors(left=-20)
13    utime.sleep_ms(400)
14    alphabot.stop()
15
16 while True:
17     pass
  
```

Missione • 3 🔓 a discrezione del supervisore

Prova del rivelatore di ostacolo

• Missione 3-1 : Scoperta del sensore ad ultrasuoni

ASM, abbiamo un problema! Uno dei robot sembra bloccato.

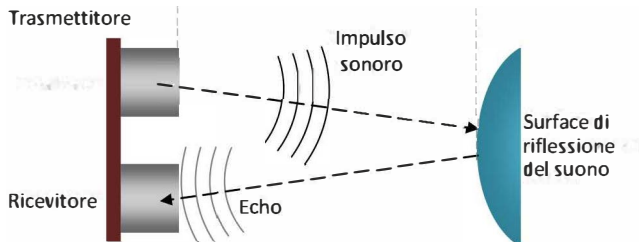
CHLOÉ

Dev'essere di sicuro un problema del rilevatore di ostacoli. Secondo me, è incastrato contro una roccia.

ROCK

Ricevuto Arès I. Ecco il protocollo di utilizzo del sensore ad Ultrasuoni e del sensore TOF.

Il sensore ad ultrasuoni permette di rilevare un ostacolo davanti a sé e persino di determinare una distanza.



Il sensore ha un trasmettitore e un ricevitore. Il trasmettitore invia un impulso sonoro. Quando questo colpisce un ostacolo, è parzialmente rinvio, in eco, al ricevitore. Un calcolatore "conta" poi il tempo impiegato per questo viaggio di andata e ritorno. Conoscendo la velocità di movimento del suono nell'aria si può dedurre la distanza ($V=d/t$).

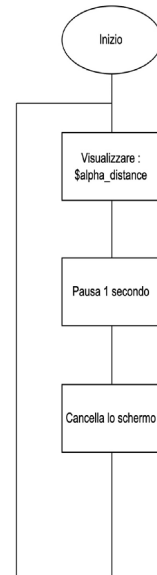
Per questa missione mostreremo la distanza di un oggetto in modo permanente sullo schermo OLED.

Per questo abbiamo bisogno di blocchi che si trovano in :

Robots

Inputs/Outputs

Loops



```

1 import machine
2 from stm32_ssd1306 import SSD1306, SSD1306_I2C
3 from stm32_alphabot_v2 import AlphaBot_v2
4 import utime
5
6 oled = SSD1306_I2C(128, 64, machine.I2C(1))
7 alphabot = AlphaBot_v2()
8
9 oled.fill(0)
10 oled.show()
11
12 while True:
13     oled.text(str(alphabot.readUltrasonicDistance()), 0, 0)
14     oled.show()
15     utime.sleep(1)
16     oled.fill(0)
17     oled.show()
    
```

Ripeti indefinitamente

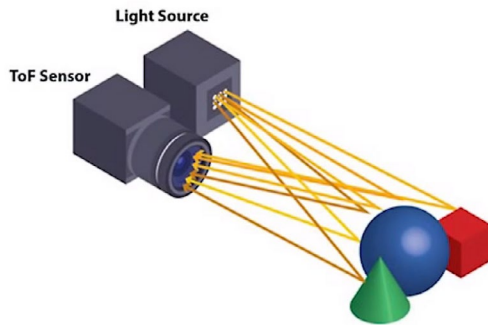
[Alphabot] visualizza il testo [Alphabot - Sensore a ultrasuoni] distanza (cm) nella posizione x 0 y 0 sul display

attendi 1 secondo/i

[Alphabot] cancella display

• Missione 3-2 : Scoperta del Time of Flight (ToF)

Il sensore Time of Flight è un sensore che permette di rilevare un ostacolo davanti a sé e persino di determinare una distanza.



Il sensore ha un trasmettitore e un ricevitore. Il trasmettitore invia un impulso laser. Quando questo colpisce un ostacolo, è parzialmente rinvio, in eco, al ricevitore. Un calcolatore "conta" poi il tempo impiegato per questo viaggio di andata e ritorno. Conoscendo la velocità di movimento del suono nell'aria si può dedurre la distanza ($V=d/t$).

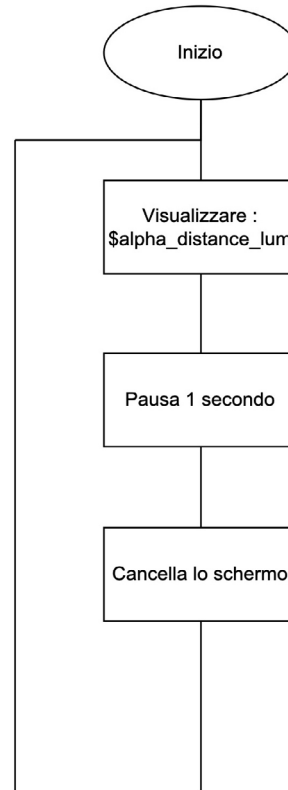
Per questa missione mostreremo la distanza di un oggetto in modo permanente sullo schermo OLED.

Per questo abbiamo bisogno di blocchi che si trovano in :

🤖 Robots

↔ Inputs/Outputs

🔄 Loops



• Missione 3-3 : sensori di distanza e motori

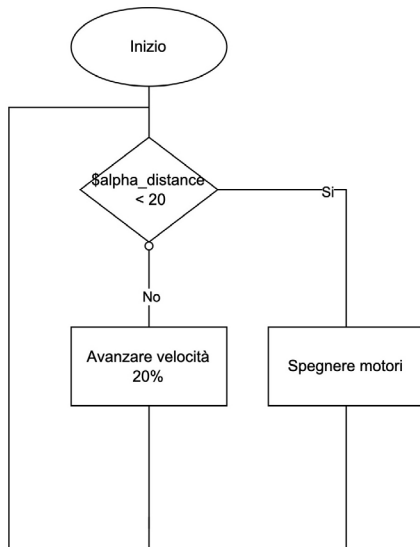
Sfrutteremo subito questo sensore ad ultrasuoni per lo spostamento del robot. Questo dovrà spostarsi fino a che la distanza tra il robot e un muro sia quella predeterminata, per esempio 30 cm.

Per questo abbiamo bisogno di blocchi che si trovano in :

🤖 Robots
🔧 Logic

🔄 Loops

Useremo in questa missione i comparatori logici, che permettono di verificare se un'informazione è vera o falsa. Una volta effettuata questa verifica, ciò consente di attivare una azione ad un'altra. Nel nostro caso, la domanda sarà: la distanza tra il mio robot e il muro è inferiore a 20 cm ?



```

1 import machine
2 from stm32_ssd1306 import SSD1306, SSD1306_I2C
3 from stm32_alphabot_v2 import AlphaBot_v2
4
5 oled = SSD1306_I2C(128, 64, machine.I2C(1))
6 alphabot = AlphaBot_v2()
7
8 oled.fill(0)
9 oled.show()
10
11 while True:
12     while not alphabot.readUltrasonicDistance() < 20:
13         alphabot.moveForward(20)
14     alphabot.stop()
  
```

Misurando la distanza ottenuta dopo lo spostamento del robot, potrebbe esserci una leggera differenza. Questo è dovuto a due parametri :


- Il primo è un errore umano. Si dimentica che non si deve misurare a livello delle ruote, ma piuttosto a livello del sensore di distanza che effettua la misurazione.
- Il secondo parametro è determinato dall'inerzia del robot e dalla sua velocità di calcolo. Il robot non può fermarsi istantaneamente e più elevata è la velocità più grande è l'errore dovuto alla frenata. Analogamente, il tempo di calcolo della carta, in generale trascurabile, può talvolta provocare una leggera deviazione di alcuni millimetri se la velocità è elevata.


È assolutamente possibile utilizzare il sensore Tof al posto del sensore ad ultrasuoni, il programma sarà identico tranne che per il nome del sensore utilizzato.

Missione • 4 a discrezione del supervisore

Pilotaggio in tempo reale


ASM, qui Arès I. Confermiamo il problema: il robot bloccato è troppo vicino ad un ostacolo.





Magari potremmo metterci le nostre tute per andare sbloccarlo ?

Non se ne parla nemmeno Chloe ! È troppo pericoloso per voi, userete un telecomando per robot per disincastarlo. Ecco una procedura per imparare a comunicare con i robot.



Per pilotare il piccolo robot, useremo il telecomando dedicato.

Il principio è che il robot riceve un'informazione e reagisce in base all'informazione ricevuta.

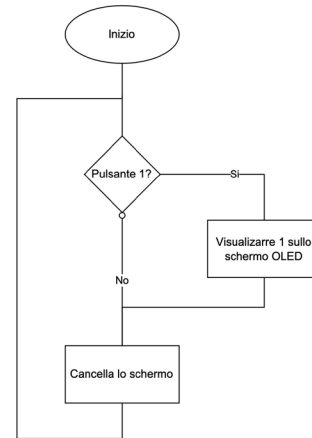
• Missione 4-1 : Il telecomando

In una prima missione, visualizzeremo sulla scheda un'informazione a seconda del tasto del telecomando. Ad esempio, se si preme il tasto 1 lo schermo dovrebbe visualizzare 1.

Per questo occorreranno i blocchi che si trovano in :

 Robots

 Loops



```

1 import machine
2 from stm32_ssd1306 import SSD1306, SSD1306_I2C
3 from stm32_alphabot_v2 import AlphaBot_v2
4 import utime
5 from stm32_nec import NEC_8, NEC_16
6 import gc
7
8 oled = SSD1306_I2C(128, 64, machine.I2C(1))
9 alphabot = AlphaBot_v2()
10 ir_current_remote_code = None
11
12 def remoteNEC_basicBlack_getButton(hexCode):
13     if hexCode == 0x0c: return "1"
14     elif hexCode == 0x18: return "2"
15     elif hexCode == 0x5e: return "3"
16     elif hexCode == 0x08: return "4"
17     elif hexCode == 0x1c: return "5"
18     elif hexCode == 0x5a: return "6"
19     elif hexCode == 0x42: return "7"
20     elif hexCode == 0x52: return "8"
21     elif hexCode == 0x4a: return "9"
22     elif hexCode == 0x16: return "0"
23     elif hexCode == 0x40: return "up"
24     elif hexCode == 0x19: return "down"
25     elif hexCode == 0x07: return "left"
26     elif hexCode == 0x09: return "right"
27     elif hexCode == 0x15: return "enter_save"
28     elif hexCode == 0x0d: return "back"
29     elif hexCode == 0x45: return "volMinus"
30     elif hexCode == 0x47: return "volPlus"
  
```

```

31 elif hexCode == 0x46: return "play_pause"
32 elif hexCode == 0x44: return "setup"
33 elif hexCode == 0x43: return "stop_mode"
34 else: return "NEC remote code error"
35
36 def remoteNEC_callback(data, addr, ctrl):
37     global tr_current_remote_code
38     if data < 0: # NEC protocol sends repeat codes.
39         print('Repeat code.')
40     else:
41         print('Data {:02x} Addr {:04x} Ctrl {:02x}'.format(data, addr, ctrl))
42         tr_current_remote_code = remoteNEC_basicBlack_getButton(data)
43
44 classes = (NEC_8, NEC_16)
45 tr_remote = classes[0](alphanot.pin_IR, remoteNEC_callback)
46
47 oled.fill(0)
48 oled.show()
49
50 while True:
51     utime.sleep_ms(150)
52     gc.collect()
53     if tr_current_remote_code == "1":
54         oled.text('1', 0, 0)
55         oled.show()
56     utime.sleep(1)
57     oled.fill(0)
58     oled.show()

```

Ripeti indefinitamente

[Alphanot] se il comando **tasto1** è ricevuto dal telecomando NEC nero allora

[Alphanot] visualizza il testo **" 1 "** nella posizione x **0** y **0** sul display

attendi **1** secondo/i

[Alphanot] cancella display

ASM siamo riusciti a comunicare con il robot. Ma abbiamo un dubbio sulla procedura per farlo muovere.

CHLOE

Potremmo servirci del robot dell' hangar per stabilire il collegamento!

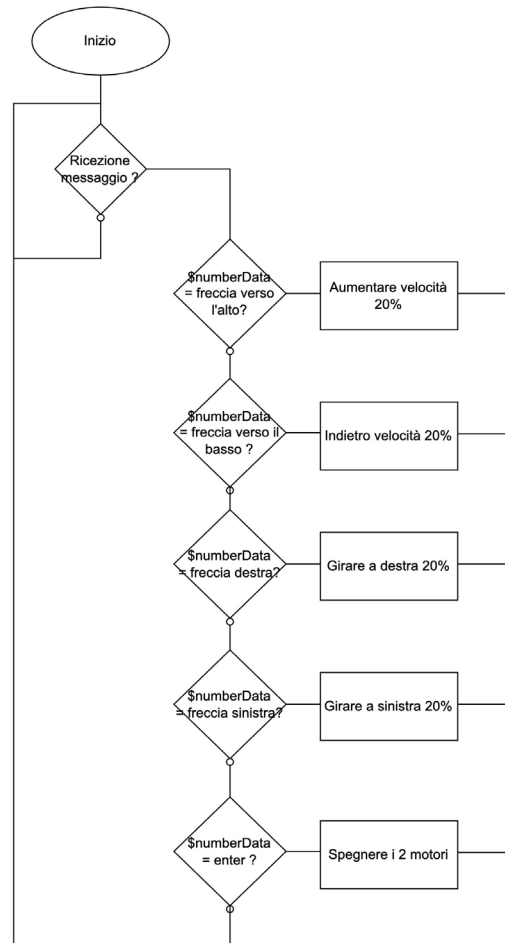
ROCK

Tranquilli! Vi inviamo una procedura. E d'accordo per il robot dell' hangar.

ASM

• Missione 4-2 Test in condizioni "Marziana" reale

Adesso piloteremo il nostro piccolo robot nell'hangar. Per questo useremo i programmi realizzati precedentemente. L'obiettivo sarà quello di far avanzare il robot con la freccia in alto, ruotare con le frecce destra e sinistra, indietro con la freccia in basso e fermarsi con il pulsante invio (Enter sul telecomando).



Ripeti indefinitamente

[Alphabot] se il comando **alto (PREV)** è ricevuto dal telecomando NEC nero allora

[Alphabot] controlla il robot vai **avanti** velocità **20** (%)

[Alphabot] se il comando **basso (NEXT)** è ricevuto dal telecomando NEC nero allora

[Alphabot] controlla il robot vai **indietro** velocità **20** (%)

[Alphabot] se il comando **sinistra (CH-)** è ricevuto dal telecomando NEC nero allora

[Alphabot] gira a **sinistra** velocità **20** (%)

[Alphabot] se il comando **destra (CH+)** è ricevuto dal telecomando NEC nero allora

[Alphabot] gira a **destra** velocità **20** (%)

[Alphabot] se il comando **ENTER/SAVE** è ricevuto dal telecomando NEC nero allora

[Alphabot] arresta il motore **destra & sinistro**

```

10 while True:
11     utime.sleep_ms(150)
12     gc.collect()
13     if ir_current_remote_code == "up":
14         alphabot.moveForward(20)
15     utime.sleep_ms(150)
16     gc.collect()
17     if ir_current_remote_code == "down":
18         alphabot.moveBackward(20)
19     utime.sleep_ms(150)
20     gc.collect()
21     if ir_current_remote_code == "left":
22         alphabot.turnLeft(20)
23     utime.sleep_ms(150)
24     gc.collect()
25     if ir_current_remote_code == "right":
26         alphabot.turnRight(20)
27     utime.sleep_ms(150)
28     gc.collect()
29     if ir_current_remote_code == "enter_save":
30         alphabot.stop()
31

```

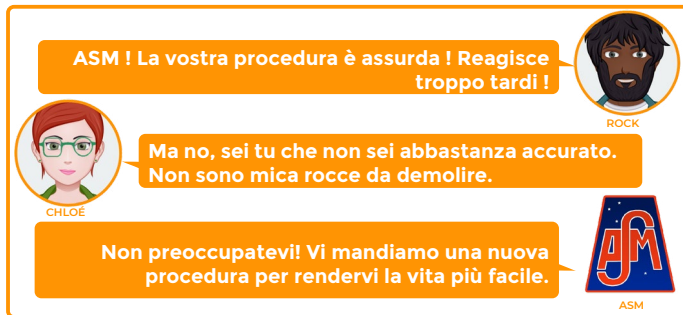
Il codice python è volutamente incompleto, l'intero inizio del programma è identico al programma precedente e qui sono presenti solo le chiamate delle funzioni.

Dopo questa simulazione, ci si rende conto che pilotare il robot con il telecomando è molto complesso, una soluzione più efficace è programmare il robot affinché decida in autonomia secondo l'ambiente dove evolve.

Con un semplice ordine, il robot dovrà adattarsi per svolgere al meglio il suo compito. Per esempio, "Vai a 100 metri al Nord!" E con la sua programmazione dovrà andare al punto d'incontro che si trova 100 metri verso nord.

Missione • 5 a discrezione del supervisore

Spostamento in modalità casella



• Missione 5-1 : Spostamento in modalità casella

Per questa missione, inizieremo a programmare in modalità casella. Il robot farà piccoli movimenti come se fosse su una scacchiera da scacchi o da dama. Questo semplificherà la programmazione iniziale e ci permetterà di dire al robot, ad esempio, di spostarsi di 3 caselle in avanti e di 2 a destra.

Per prima cosa metteremo a profitto le missioni precedenti con lo spostamento a 90° che si adatta perfettamente per uno spostamento di questo tipo e poi misureremo quanto tempo occorre per avanzare di una casella andando dritto.

Fisseremo la dimensione della casella a 10 cm questo permette di avere uno schema di 4x3 caselle su un foglio A3. Il codice di missione che useremo è quello della 2-1 e della 2-2. E anche quello della missione 4-2.

Eviteremo di riscrivere un nuovo codice : utilizzeremo il codice già fatto migliorandolo, come si fa spesso in informatica. Per questo, utilizzeremo le funzioni :

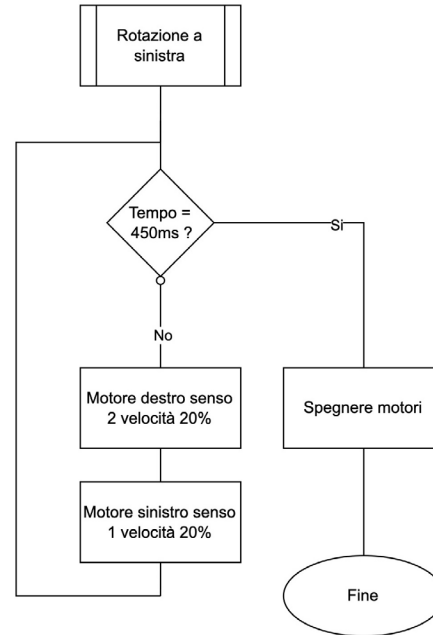
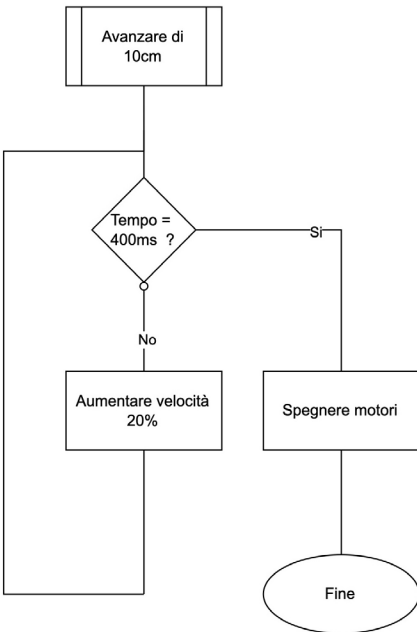
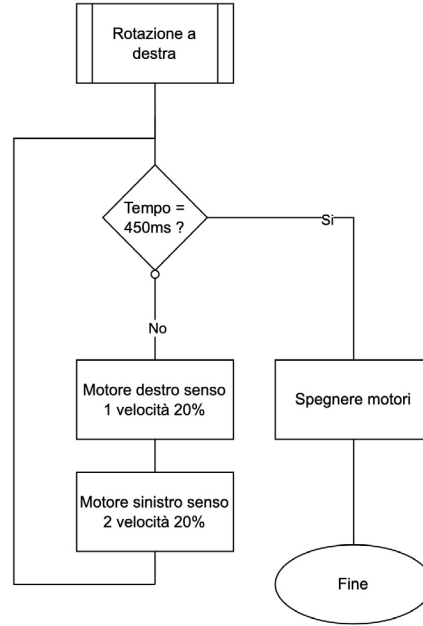
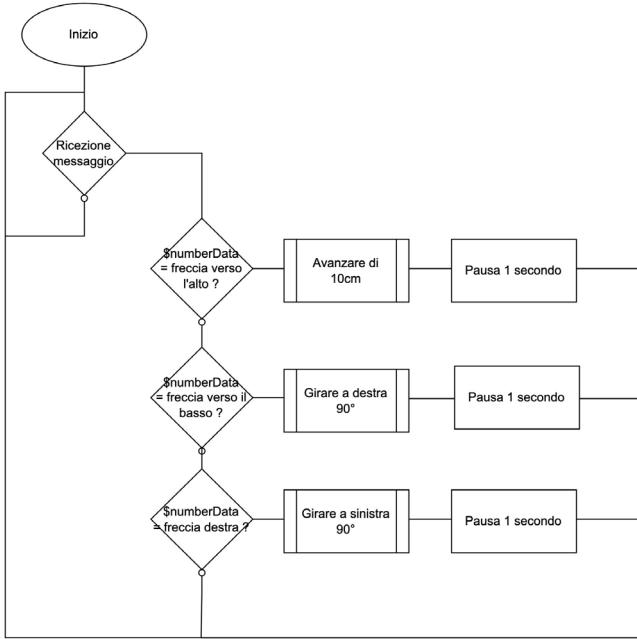


Per fare questo, creeremo già tre funzioni: una per avanzare, una per girare a destra e una per girare a sinistra.

Non ci sono cambiamenti rispetto alla missione 2-2 tranne che invece di mettere i blocchi nel ciclo "Ripeti a tempo indeterminato" o "All'avvio", li metteremo in un blocco funzione a cui daremo un nome riconoscibile in modo da non perderci nel codice.

Aggiungeremo un blocco di condizioni in modo che il robot esegua la funzione quando si preme il pulsante giusto del telecomando, poi si aggiunge un secondo di latenza per avere il tempo di premere un tasto del telecomando in zona morta (tasto non operativo).

Questo dà gli algorigrammi presentati nelle seguenti pagine :



Ripeti indefinitamente

[Alphabot] se il comando alto (PREV) è ricevuto dal telecomando NEC nero allora

avanti di 10 cm

attendi 1 secondo/i

[Alphabot] se il comando destra (CH+) è ricevuto dal telecomando NEC nero allora

gira a destra 90°

attendi 1 secondo/i

[Alphabot] se il comando sinistra (CH-) è ricevuto dal telecomando NEC nero allora

gira a sinistra 90°

attendi 1 secondo/i

definisci gira a destra 90°

[Alphabot] controlla il motore destra direzione velocità 20 (%)

[Alphabot] controlla il motore sinistra direzione velocità 20 (%)

attendi 450 millisecondo/i

[Alphabot] arresta il motore destro & sinistro

definisci avanti di 10 cm

[Alphabot] controlla il robot vai avanti velocità 100 (%)

attendi 400 millisecondo/i

[Alphabot] arresta il motore destro & sinistro

definisci gira a sinistra 90°

[Alphabot] controlla il motore sinistra direzione velocità 20 (%)

[Alphabot] controlla il motore destra direzione velocità 20 (%)

attendi 450 millisecondo/i

[Alphabot] arresta il motore destro & sinistro

```

45 def tourner_gauche_90_C2_B0():
46     alphabot.setMotors(left=20)
47     alphabot.setMotors(right=-20)
48     utime.sleep_ms(450)
49     alphabot.stop()
50
51 def tourner__droite_90_C2_B0():
52     alphabot.setMotors(right=20)
53     alphabot.setMotors(left=-20)
54     utime.sleep_ms(450)
55     alphabot.stop()
56
57 def avancer_10cm():
58     alphabot.moveForward(20)
59     utime.sleep_ms(400)
60     alphabot.stop()
61

```

```

62 while True:
63     utime.sleep_ms(150)
64     gc.collect()
65     if ir_current_remote_code == "up":
66         avancer_10cm()
67         utime.sleep(1)
68     utime.sleep_ms(150)
69     gc.collect()
70     if ir_current_remote_code == "right":
71         tourner__droite_90_C2_B0()
72         utime.sleep(1)
73     utime.sleep_ms(150)
74     gc.collect()
75     if ir_current_remote_code == "left":
76         tourner_gauche_90_C2_B0()
77         utime.sleep(1)
78

```

Ora che abbiamo le nostre funzioni per programmare, diventa più semplice, basta chiamare la funzione nel programma principale per eseguire l'intera funzione.

Come si può vedere, questo semplifica notevolmente il programma, non occorre più scrivere più linee e linee di codice con il principio delle funzioni.

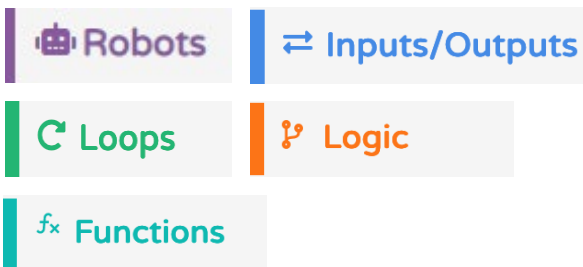
• Missione 5-2 : Individuazione di ostacoli prima dello spostamento

In questa missione, aggiungeremo gli elementi della missione 3. Infatti, muoversi senza guardare dove si va rischia di porre rapidamente dei problemi. Nella vita di tutti i giorni e per gli umani, gli occhi sono i principali organi di rilevamento di ostacoli. Infatti, la nostra vista tridimensionale ci permette di individuare gli ostacoli e di valutare la loro distanza. Nel nostro caso, sarà il sensore ad ultrasuoni o Time of Flight a svolgere questo ruolo.

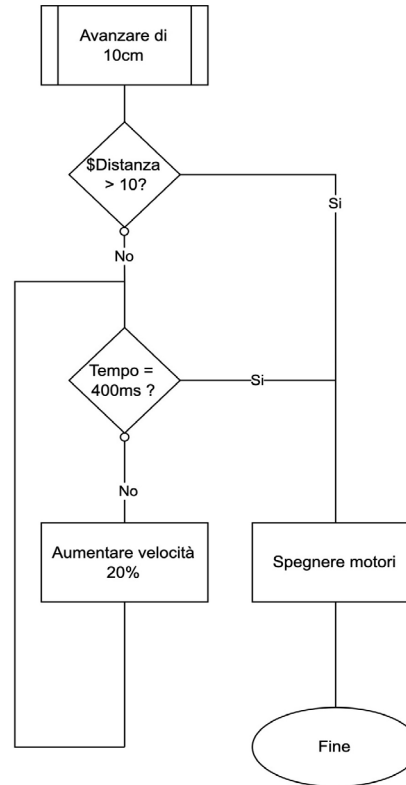
L'obiettivo di questa missione sarà di spostarsi sulla scacchiera controllando prima di ogni spostamento se la casella in cui ci si deve muovere è libera. E se non lo è, non andare sulla casella occupata e segnare una croce sulla mappa.

Per questo modificheremo solo la funzione avanzare in modo che possa rilevare gli ostacoli.

Avremo bisogno dei blocchi che si trovano in :



Qui vengono presentati solo i cambiamenti nella funzione «Avanzare di 10 cm».




```


45 def avancer_10_cm():
46     if alphabot.readUltrasonicDistance() > 10:
47         alphabot.moveForward(20)
48         utime.sleep_ms(400)
49         alphabot.stop()
50     else:
51         alphabot.stop()

```


Nel codice Python, la funzione relativa al sensore ad ultrasuoni non è aggiunta qui. Se necessario, questa è disponibile nella missione 3.

Abbiamo già fatto molti progressi, il nostro robot è ora in grado di muoversi da solo evitando gli ostacoli o piuttosto ignorandoli!

ASM ! Super procedura, funziona quasi bene. Ma il problema è quando si entra nei colori della base. In questo caso, il sensore impazzisce e il robot diventa impossibile da pilotare.




ROCK



CHLOÉ

Non si potrebbero seguire le linee a terra ?

Ben detto Chloé, è l'idea. Vi mandiamo le procedure per risolvere i problemi.



ASM

Missione • 6 🕒 a discrezione del supervisore

Tracker di linea

• Missione 6-1 : Seguire una linea

Il tracker di linea integrato nel robot Alhabot è composto da 5 sensori ad infrarossi situati sotto il telaio del robot.

Il principio di funzionamento è il seguente: Il tracker di linea è composto da 5 sensori ad infrarossi situati sotto il telaio del robot. Il LED emette un segnale e il foto-transistor lo rileva. Il nero assorbe la radiazione infrarossa, il ricevitore non rileva il segnale emesso quando il sensore è sopra la linea, quindi questo restituisce 0. In caso contrario, il segnale viene riflesso dal bianco, il sensore restituisce 1.

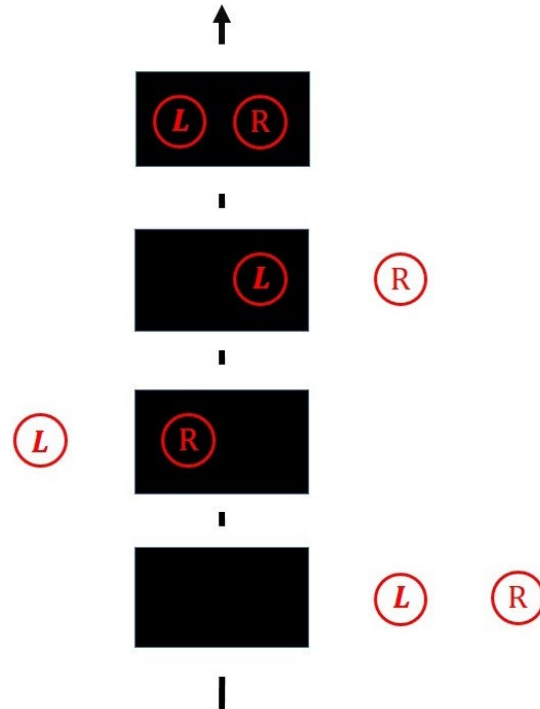
Inizialmente useremo solo due sensori per tracciare la linea.

- **Caso 1:** I 2 sensori sono sopra la linea.
-> Ritorno 0,0

- **Caso 2:** La linea-L è sopra la linea e la linea-R è all'esterno.
-> Ritorno 0,1

- **Caso 3:** La linea-L è fuori e la linea-R è sopra la linea.
-> Ritorno 1,0

- **Caso 4:** I 2 sensori sono fuori dalla linea nera.
-> Ritorno 1,1



Qui possiamo vedere il funzionamento dei sensori e le informazioni che otteniamo sulla scheda microcontrollore.

Grazie a queste informazioni, saremo in grado di riflettere sul comportamento di ogni motore in base alle informazioni dei sensori.

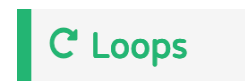
Per facilitare il compito, useremo una tabella logica in cui avremo due colonne per i sensori e due colonne per i motori. Useremo il codice 0 per motore spento o colore bianco e 1 per motore attivo e colore nero.

Sensore sinistro	Sensore destro	Motore sinistro	Motore destro
0 / bianco	0 / bianco	?	?
0 / bianco	1 / nero	1 / attivare	0 / spento
1 / nero	0 / bianco	0 / spento	1 / attivare
1 / nero	1 / nero	1 / attivare	1 / attivare

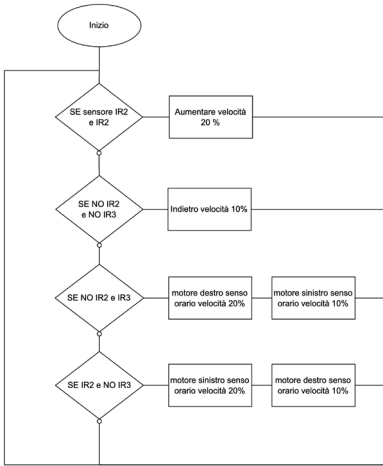
Ci si rende conto che nel primo caso non c'è più una linea e che occorre dunque attuare una strategia di ricerca di linea. Come per esempio «avanzare di 5 cm poi girare in cerchio fino a rilevamento linea» o semplicemente «arretrare» perché si è stati troppo veloci e si è persa la linea.

Proviamo a farlo funzionare.

Per questo avremo bisogno dei blocchi seguenti :



Nel programma si vede inizialmente nel blocco "All'avvio" un "spegnere il motore". Si tratta di un mini inizializzazione, che permette di essere sicuri che si avvia un motore spento. In questa inizializzazione, si possono aggiungere molte cose per essere sicuri di iniziare il programma in buone condizioni; ad esempio un «cancella schermo» per essere sicuri che nulla rimane sullo schermo. Si può aggiungere una pausa per avere il tempo di posizionare bene il robot prima che si avvii.



```

1 while True:
2   if isSensorAboveLine(alphabot, sensor='IR2') and isSensorAboveLine(alphabot, sensor='IR3'):
3     alphabot.moveForward(20)
4   if not isSensorAboveLine(alphabot, sensor='IR2') and not isSensorAboveLine(alphabot, sensor='IR3'):
5     alphabot.moveBackward(15)
6   if not isSensorAboveLine(alphabot, sensor='IR2') and isSensorAboveLine(alphabot, sensor='IR3'):
7     alphabot.setMotors(right=0)
8     alphabot.setMotors(left=20)
9   if isSensorAboveLine(alphabot, sensor='IR2') and not isSensorAboveLine(alphabot, sensor='IR3'):
10    alphabot.setMotors(right=20)
11    alphabot.setMotors(left=0)
  
```

Ripeti indefinitamente

• Missione 6-2 : Seguire una linea con 3 sensori

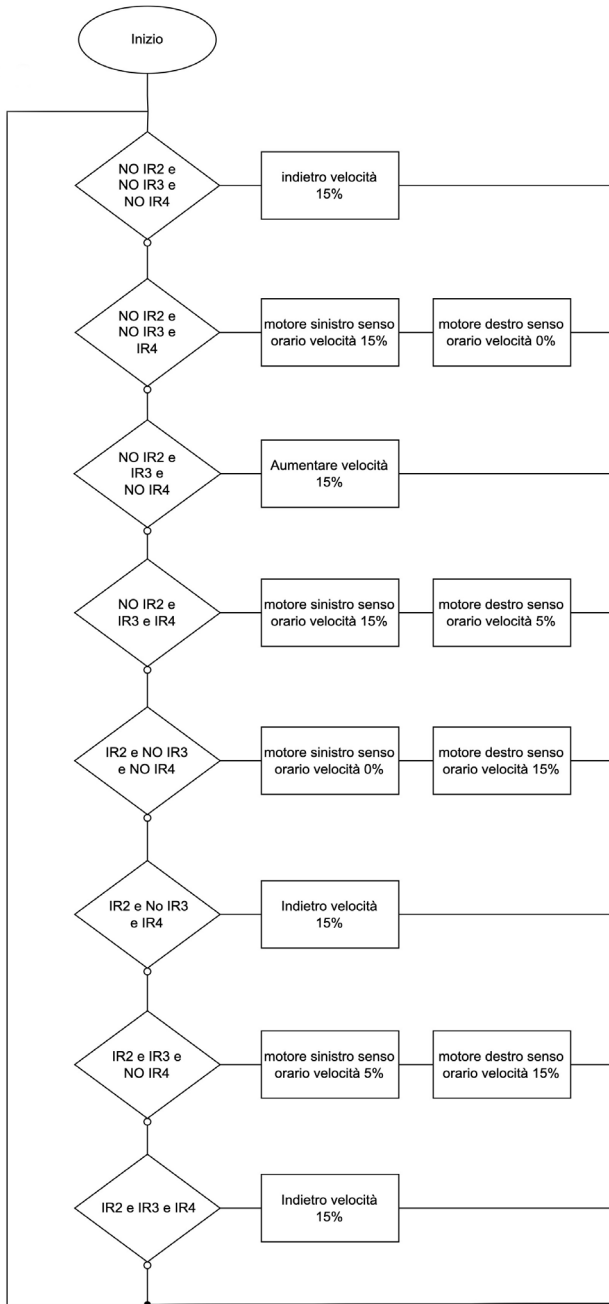
Ora cerchiamo di migliorare il nostro programma utilizzando tre sensori di linea: a sinistra, al centro e a destra del robot. Utilizzeremo il codice seguente : 0 per bianco e 1 per linea nera.

Sensore sinistro	Sensore centrale	Sensore destro	Motore sinistro	Motore destro
0	0	0	-15%	-15%
0	0	1	15%	0
0	1	0	15%	15%
0	1	1	15%	5%
1	0	0	0	15%
1	0	1	-15%	-15%
1	1	0	5%	15%
1	1	1	-15%	-15%

```

33 while True:
34     if not alphanbot.TRSensors_readLine(2) <= 300 and not alphanbot.TRSensors_readLine(3) <= 300 and not alphanbot.TRSensors_readLine(4) <= 300:
35         alphanbot.moveBackward(15)
36     if not alphanbot.TRSensors_readLine(2) <= 300 and not alphanbot.TRSensors_readLine(3) <= 300 and alphanbot.TRSensors_readLine(4) <= 300:
37         alphanbot.setMotors(left=15)
38         alphanbot.setMotors(right=0)
39     if not alphanbot.TRSensors_readLine(2) <= 300 and alphanbot.TRSensors_readLine(3) <= 300 and not alphanbot.TRSensors_readLine(4) <= 300:
40         alphanbot.moveForward(15)
41     if not alphanbot.TRSensors_readLine(2) <= 300 and alphanbot.TRSensors_readLine(3) <= 300 and alphanbot.TRSensors_readLine(4) <= 300:
42         alphanbot.setMotors(left=15)
43         alphanbot.setMotors(right=5)
44     if alphanbot.TRSensors_readLine(2) <= 300 and not alphanbot.TRSensors_readLine(3) <= 300 and not alphanbot.TRSensors_readLine(4) <= 300:
45         alphanbot.setMotors(left=0)
46         alphanbot.setMotors(right=15)
47     if alphanbot.TRSensors_readLine(2) <= 300 and not alphanbot.TRSensors_readLine(3) <= 300 and alphanbot.TRSensors_readLine(4) <= 300:
48         alphanbot.moveBackward(15)
49     if alphanbot.TRSensors_readLine(2) <= 300 and alphanbot.TRSensors_readLine(3) <= 300 and not alphanbot.TRSensors_readLine(4) <= 300:
50         alphanbot.setMotors(left=5)
51         alphanbot.setMotors(right=15)
52     if alphanbot.TRSensors_readLine(2) <= 300 and alphanbot.TRSensors_readLine(3) <= 300 and alphanbot.TRSensors_readLine(4) <= 300:
53         alphanbot.moveBackward(15)
54     oled.fill(0)
55     oled.show()

```



All'avvio

imposta soglia di rilevamento su 300

Ripeti indefinitamente

se non [Alphabot] sensore IR2 > sopra la linea nera < valore limite soglia di rilevamento e non [Alphabot] sensore IR3 > sopra la linea nera < valore limite soglia di rilevamento e non [Alphabot] sensore IR4 > sopra la linea nera < valore limite soglia di rilevamento allora

[Alphabot] controlla il robot vai indietro < velocità 15 (%)

se non [Alphabot] sensore IR2 > sopra la linea nera < valore limite soglia di rilevamento e non [Alphabot] sensore IR3 > sopra la linea nera < valore limite soglia di rilevamento e [Alphabot] sensore IR4 > sopra la linea nera < valore limite soglia di rilevamento allora

[Alphabot] controlla il motore sinistra > direzione < velocità 15 (%)

[Alphabot] controlla il motore destra > direzione < velocità 5 (%)

se non [Alphabot] sensore IR2 > sopra la linea nera < valore limite soglia di rilevamento e [Alphabot] sensore IR3 > sopra la linea nera < valore limite soglia di rilevamento e non [Alphabot] sensore IR4 > sopra la linea nera < valore limite soglia di rilevamento allora

[Alphabot] controlla il motore sinistra > direzione < velocità 15 (%)

[Alphabot] controlla il motore destra > direzione < velocità 15 (%)

se non [Alphabot] sensore IR2 > sopra la linea nera < valore limite soglia di rilevamento e [Alphabot] sensore IR3 > sopra la linea nera < valore limite soglia di rilevamento e [Alphabot] sensore IR4 > sopra la linea nera < valore limite soglia di rilevamento allora

[Alphabot] controlla il motore sinistra > direzione < velocità 15 (%)

[Alphabot] controlla il motore destra > direzione < velocità 5 (%)

se [Alphabot] sensore IR2 > sopra la linea nera < valore limite soglia di rilevamento e non [Alphabot] sensore IR3 > sopra la linea nera < valore limite soglia di rilevamento e non [Alphabot] sensore IR4 > sopra la linea nera < valore limite soglia di rilevamento allora

[Alphabot] controlla il motore sinistra > direzione < velocità 0 (%)

[Alphabot] controlla il motore destra > direzione < velocità 15 (%)

se [Alphabot] sensore IR2 > sopra la linea nera < valore limite soglia di rilevamento e non [Alphabot] sensore IR3 > sopra la linea nera < valore limite soglia di rilevamento e [Alphabot] sensore IR4 > sopra la linea nera < valore limite soglia di rilevamento allora

[Alphabot] controlla il robot vai indietro < velocità 15 (%)

se [Alphabot] sensore IR2 > sopra la linea nera < valore limite soglia di rilevamento e [Alphabot] sensore IR3 > sopra la linea nera < valore limite soglia di rilevamento e non [Alphabot] sensore IR4 > sopra la linea nera < valore limite soglia di rilevamento allora

[Alphabot] controlla il motore sinistra > direzione < velocità 5 (%)

[Alphabot] controlla il motore sinistra > direzione < velocità 15 (%)

se [Alphabot] sensore IR2 > sopra la linea nera < valore limite soglia di rilevamento e [Alphabot] sensore IR3 > sopra la linea nera < valore limite soglia di rilevamento e [Alphabot] sensore IR4 > sopra la linea nera < valore limite soglia di rilevamento allora

[Alphabot] controlla il robot vai indietro < velocità 15 (%)

• Missione 6-3 : Seguire una linea ed evitare gli ostacoli

Per questa missione, immaginiamo che il robot rilevi un oggetto mentre segue la sua linea. In questo caso, deve smettere di seguire la linea, aggirare l'oggetto e poi ritrovare la linea dall'altro lato dell'ostacolo.

Nel nostro caso, limiteremo la dimensione dell'oggetto fastidioso a 10 cm di diametro.

Useremo il lavoro precedente della missione 5 dove abbiamo usato le funzioni. Si mantengono le funzioni "Avanzare di 10 cm" e "Girare a destra" o a sinistra di 90°. Aggiungiamo solo una funzione di aggiramento di ostacolo e miglioreremo la nostra inizializzazione con una pausa.

Proviamo a farlo funzionare.

Per questo avremo bisogno dei blocchi seguenti :

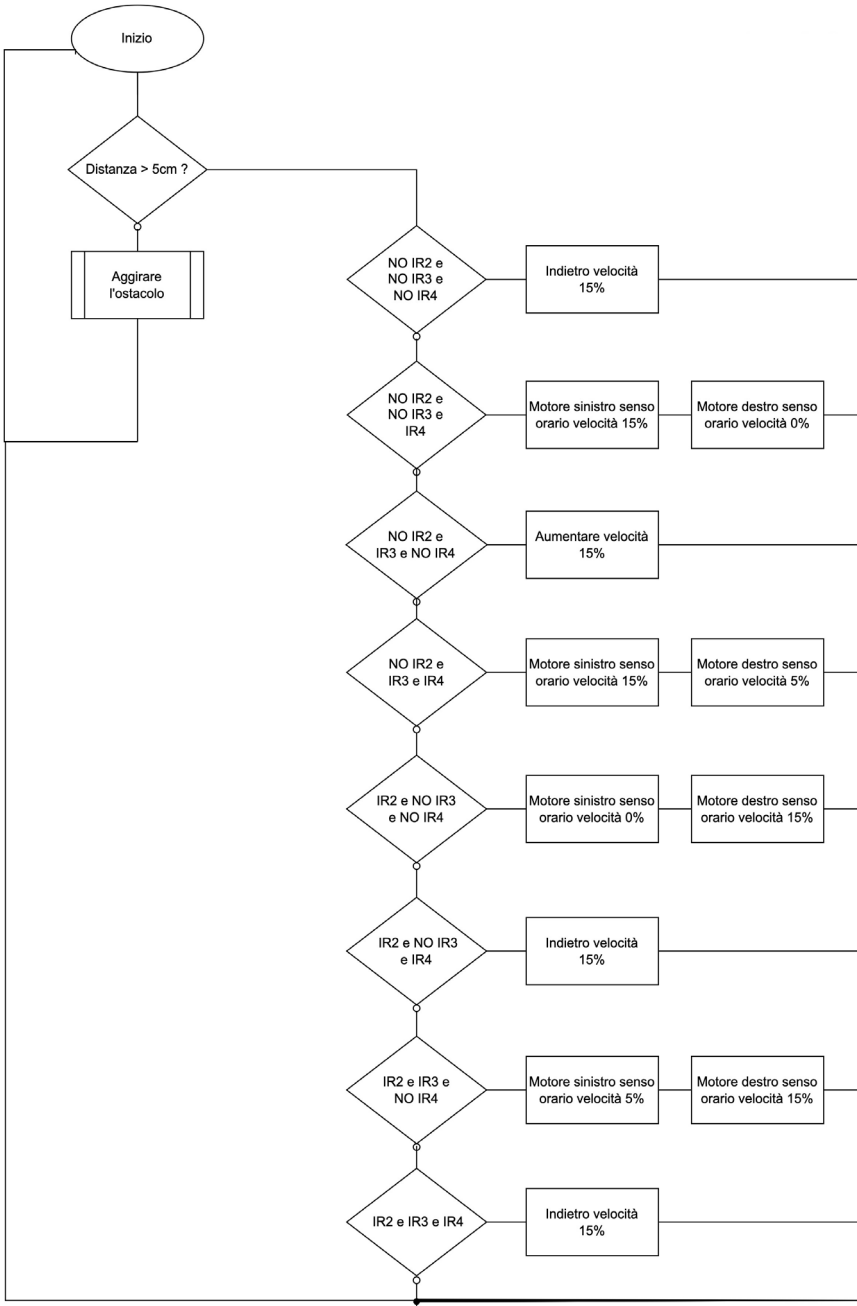


Gli algoritmi e il programma diventano più ragguardevoli, ma si ritrovano molte parti di precedenti programmi. Lo stesso vale per gli algoritmi: pochissimi cambiamenti solamente dei complementi. Gli algoritmi non sono ripresi integralmente, sono state riprodotte solo le parti importanti.

```

33 while True:
34     if alphabot.readUltrasonicDistance() <= 5:
35         tourner_gauche_90_C2_B0()
36         avancer_10_cm()
37         tourner_droite_90_C2_B0()
38         avancer_10_cm()
39         tourner_droite_90_C2_B0()
40         avancer_10_cm()
41         tourner_gauche_90_C2_B0()
42     else:
43         if not alphabot.TRSensors_readLine(2) <= 300 and not alphabot.TRSensors_readLine(3) <= 300 and not alphabot.TRSensors_readLine(4) <= 300:
44             alphabot.moveBackward(15)
45         if not alphabot.TRSensors_readLine(2) <= 300 and not alphabot.TRSensors_readLine(3) <= 300 and alphabot.TRSensors_readLine(4) <= 300:
46             alphabot.setMotors(left=15)
47             alphabot.setMotors(right=0)
48         if not alphabot.TRSensors_readLine(2) <= 300 and alphabot.TRSensors_readLine(3) <= 300 and not alphabot.TRSensors_readLine(4) <= 300:

```

Ripeti indefinitamente

se [Alphabot - Sensore a ultrasuoni] distanza (cm) <= 5 allora

- gira a sinistra 90°
- avanti di 10 cm
- gira a destra 90°
- avanti di 10 cm
- gira a destra 90°
- avanti di 10 cm
- gira a sinistra 90°

altrimenti

se non [Alphabot] sensore IR2 > sopra la linea nera > valore limite > soglia di rilevamento > e non [Alphabot] sensore IR3 > sopra la linea nera > valore limite > soglia di rilevamento > e non [Alphabot] sensore IR4 > sopra la linea nera > valore limite > soglia di rilevamento > allora

[Alphabot] controlla il robot vai indietro > velocità 15 (%)

se non [Alphabot] sensore IR2 > sopra la linea nera > valore limite > soglia di rilevamento > e non [Alphabot] sensore IR3 > sopra la linea nera > valore limite > soglia di rilevamento > e non [Alphabot] sensore IR4 > sopra la linea nera > valore limite > soglia di rilevamento > allora

Missione • 7 a discrezione del supervisore


Finalmente la routine !

Collaudo in situazione complessa: Il robot segue il suo percorso in modo continuo a si verifica un brusco calo della temperatura. Il robot interrompe tutte le attività e deve mettersi al riparo. Per questo si passa al comando manuale e si pianifica un ritorno al centro Ares I o alle coordinate indicate. Gli spostamenti saranno protetti utilizzando i sensori ad ultrasuoni o il sensore Tof per verificare se uno spostamento è valido. Il controllo a distanza sarà effettuato con il telecomando. Occorrerà prevedere spostamenti di vari generi.

Attenzione i pulsanti del telecomando sono in numero limitato e bisogna prevedere di lasciare un pulsante libero per simulare un problema e non dimenticarne un altro per rilanciare il robot in funzione percorritore di linea.


Attenzione : Qui il codice completo è presentato. Benché non sia troppo complesso, diventa quindi di ragguardevoli dimensioni!

ASM ! Tutti i robot sono tornati! Possiamo tornare alle nostre missioni iniziali.




ROCK

Quasi Rock. Ora bisogna riavviare la procedura automatica. E soprattutto monitorarlo in caso di problemi. Dovrete fare i turni per recuperare il tempo perso.

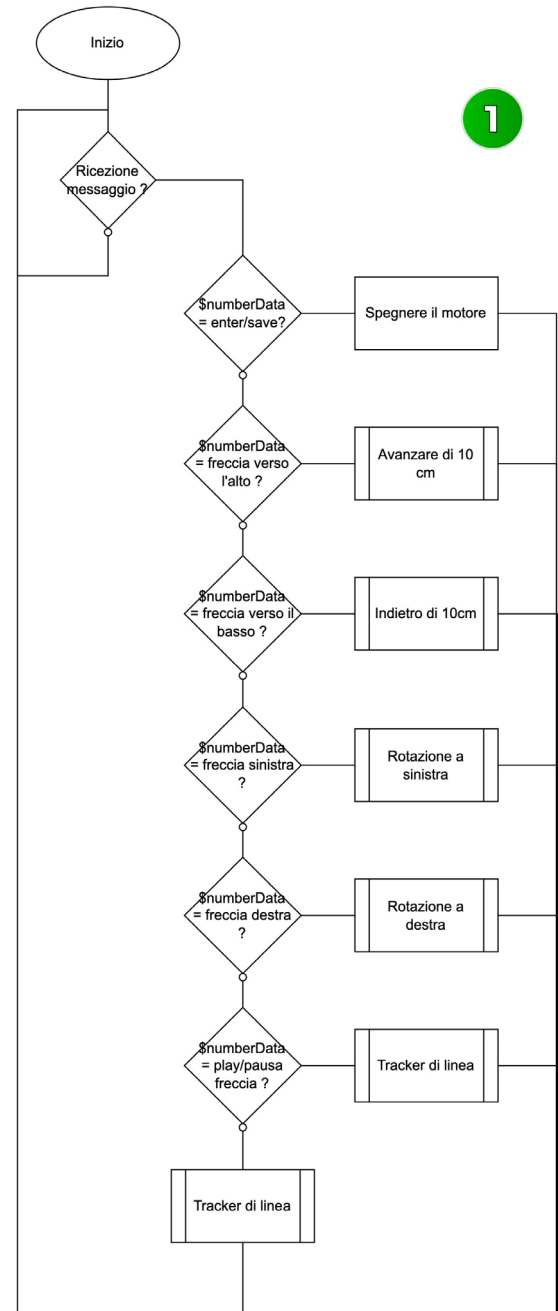


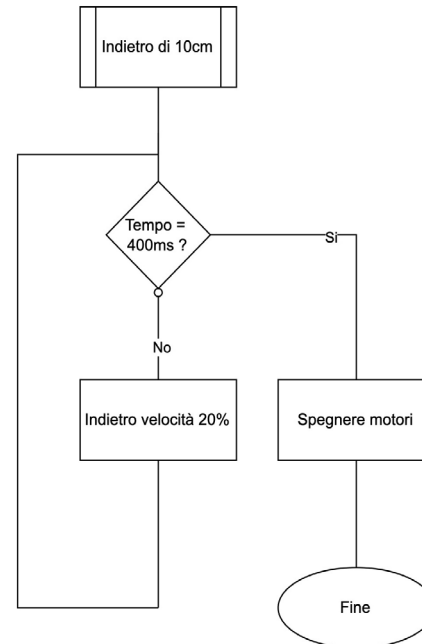
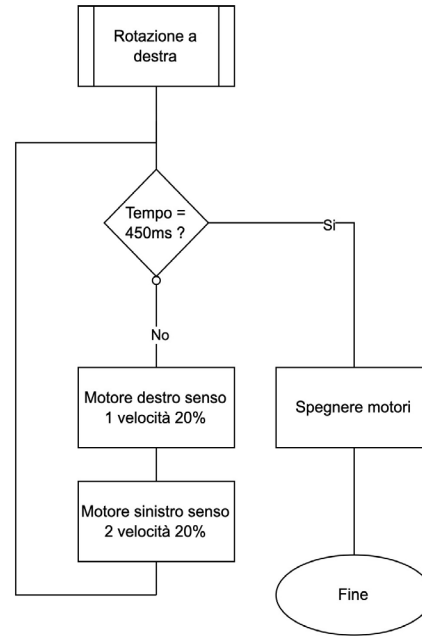
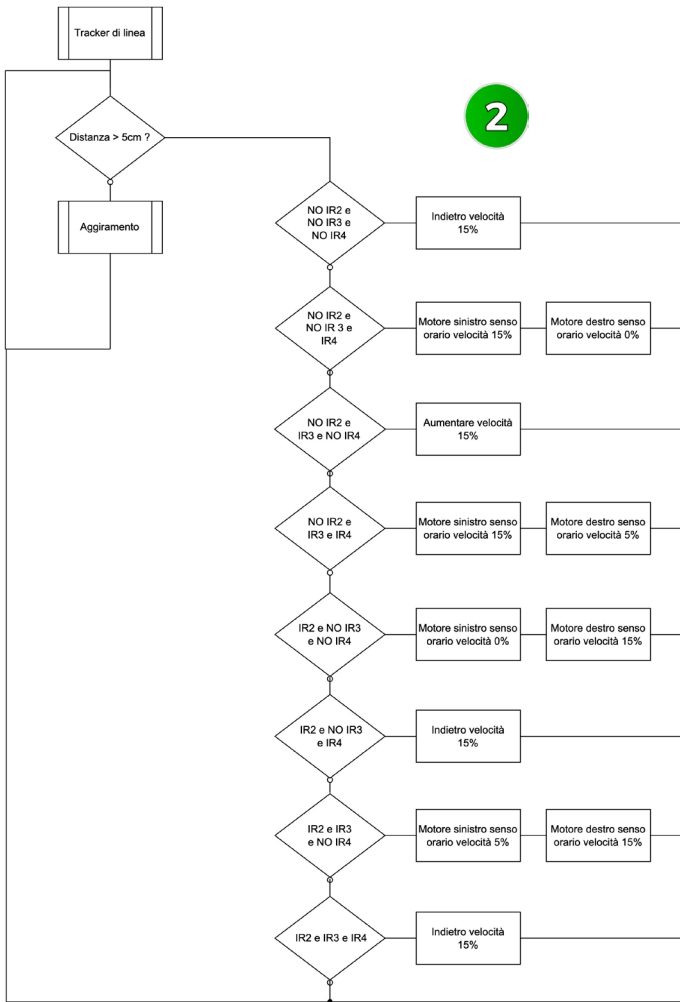
ASM

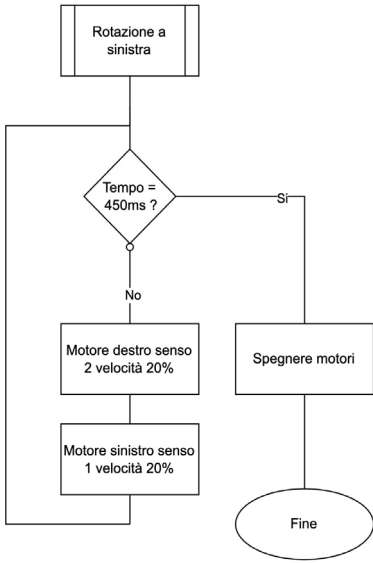
Ricevuto ASM!



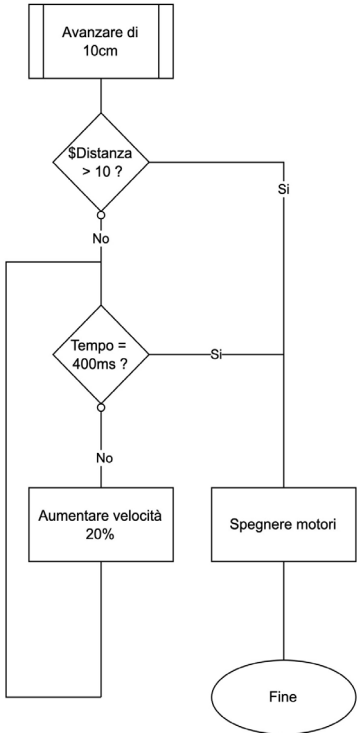
CHLOE



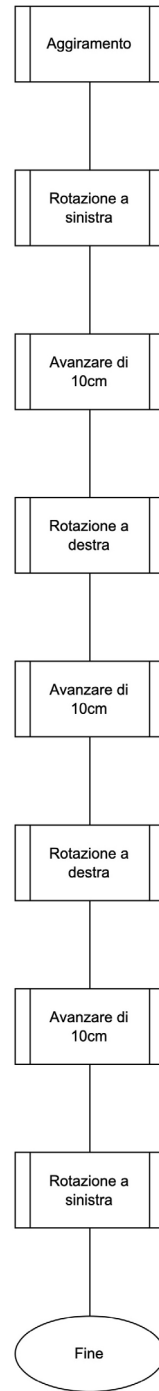




5



6



7

```

All'avvio
attendi 1 secondo/i
[Alphabot] calibra l'inseguitore di linea
imposta soglia di rilevamento su 300
    
```

```

definisci Aggiramento
  gira a sinistra 90°
  avanti di 10 cm
  gira a destra 90°
  avanti di 10 cm
  gira a destra 90°
  avanti di 10 cm
  gira a sinistra 90°
    
```

```

definisci indietro di 10 cm
  [Alphabot] controlla il robot vai indietro velocità 20 (%)
  attendi 400 millisecondo/i
  [Alphabot] arresta il motore destro & sinistro
    
```

```

Ripeti indefinitamente
  [Alphabot] se il comando ENTER/SAVE è ricevuto dal telecomando NEC nero allora
    [Alphabot] arresta il motore destro & sinistro
  altrimenti se alto (PREV) è ricevuto allora
    avanti di 10 cm
  altrimenti se basso (NEXT) è ricevuto allora
    indietro di 10 cm
  altrimenti se sinistra (CH-) è ricevuto allora
    gira a sinistra 90°
  altrimenti se destra (CH+) è ricevuto allora
    gira a destra 90°
  altrimenti se tasto1 è ricevuto allora
    suiveur de ligne
  altrimenti
    suiveur de ligne
    
```

```

definisci gira a sinistra 90°
  [Alphabot] controlla il motore sinistra direzione velocità 20 (%)
  [Alphabot] controlla il motore destra direzione velocità 20 (%)
  attendi 450 millisecondo/i
  [Alphabot] arresta il motore destro & sinistro
    
```

definisci **gira a destra 90°** 

[Alphabot] controlla il motore **destra** ▾ direzione  velocità **20** (%)

[Alphabot] controlla il motore **sinistra** ▾ direzione  velocità **20** (%)

attendi **450** millisecondo/i ▾

[Alphabot] arresta il motore **destra & sinistro** ▾

definisci **avanti di 10 cm** 

[Alphabot] controlla il robot vai **avanti** ▾ velocità **20** (%)

attendi **400** millisecondo/i ▾

[Alphabot] arresta il motore **destra & sinistro** ▾

definisci **sollevare da ligne** 

se [Alphabot - Sensore a ultrasuoni] distanza (cm) <= **5** allora

Aggrinamento

altrimenti

se non [Alphabot] sensore IR2 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ non [Alphabot] sensore IR3 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ non [Alphabot] sensore IR4 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ allora

[Alphabot] controlla il robot vai **indietro** ▾ velocità **15** (%)

se non [Alphabot] sensore IR2 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ non [Alphabot] sensore IR3 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ [Alphabot] sensore IR4 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ allora

[Alphabot] controlla il motore **sinistra** ▾ direzione ▾ velocità **15** (%)

[Alphabot] controlla il motore **destra** ▾ direzione ▾ velocità **0** (%)

se non [Alphabot] sensore IR2 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ [Alphabot] sensore IR3 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ non [Alphabot] sensore IR4 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ allora

[Alphabot] controlla il motore **sinistra** ▾ direzione ▾ velocità **15** (%)

[Alphabot] controlla il motore **destra** ▾ direzione ▾ velocità **15** (%)

se non [Alphabot] sensore IR2 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ [Alphabot] sensore IR3 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ [Alphabot] sensore IR4 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ allora

[Alphabot] controlla il motore **sinistra** ▾ direzione ▾ velocità **15** (%)

[Alphabot] controlla il motore **destra** ▾ direzione ▾ velocità **5** (%)

se [Alphabot] sensore IR2 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ non [Alphabot] sensore IR3 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ non [Alphabot] sensore IR4 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ allora

[Alphabot] controlla il motore **sinistra** ▾ direzione ▾ velocità **0** (%)

[Alphabot] controlla il motore **destra** ▾ direzione ▾ velocità **15** (%)

se [Alphabot] sensore IR2 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ non [Alphabot] sensore IR3 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ [Alphabot] sensore IR4 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ allora

[Alphabot] controlla il robot vai **indietro** ▾ velocità **15** (%)

se [Alphabot] sensore IR2 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ [Alphabot] sensore IR3 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ non [Alphabot] sensore IR4 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ allora

[Alphabot] controlla il motore **sinistra** ▾ direzione ▾ velocità **5** (%)

[Alphabot] controlla il motore **sinistra** ▾ direzione ▾ velocità **15** (%)

se [Alphabot] sensore IR2 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ [Alphabot] sensore IR3 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ e ▾ [Alphabot] sensore IR4 ▾ sopra la linea nera ▾ valore limite **soglia di rilevamento** ▾ allora

[Alphabot] controlla il robot vai **indietro** ▾ velocità **15** (%)

```
1 import machine
2 from stm32_alphabot_v2 import AlphaBot_v2
3 import utime
4 from stm32_nec import NEC_8, NEC_16
5 import gc
6 from stm32_ssd1306 import SSD1306, SSD1306_I2C
7 |
8 alphabot = AlphaBot_v2()
9 ir_current_remote_code = None
10 oled = SSD1306_I2C(128, 64, machine.I2C(1))
11
12 def remoteNEC_basicBlack_getButton(hexCode):
13     if hexCode == 0x0c: return "1"
14     elif hexCode == 0x18: return "2"
15     elif hexCode == 0x5e: return "3"
16     elif hexCode == 0x08: return "4"
17     elif hexCode == 0x1c: return "5"
18     elif hexCode == 0x5a: return "6"
19     elif hexCode == 0x42: return "7"
20     elif hexCode == 0x52: return "8"
21     elif hexCode == 0x4a: return "9"
22     elif hexCode == 0x16: return "0"
23     elif hexCode == 0x40: return "up"
24     elif hexCode == 0x19: return "down"
25     elif hexCode == 0x07: return "left"
26     elif hexCode == 0x09: return "right"
27     elif hexCode == 0x15: return "enter_save"
28     elif hexCode == 0x0d: return "back"
29     elif hexCode == 0x45: return "volMinus"
30     elif hexCode == 0x47: return "volPlus"
31     elif hexCode == 0x46: return "play_pause"
32     elif hexCode == 0x44: return "setup"
33     elif hexCode == 0x43: return "stop_mode"
```



```

32 elif hexCode == 0x44: return "setup"
33 elif hexCode == 0x43: return "stop_mode"
34 else: return "NEC remote code error"
35
36 def remoteNEC_callback(data, addr, ctrl):
37     global ir_current_remote_code
38     if data < 0: # NEC protocol sends repeat codes.
39         print('Repeat code.')
40     else:
41         print('Data {:02x} Addr {:04x} Ctrl {:02x}'.format(data, addr, ctrl))
42         ir_current_remote_code = remoteNEC_basicBlack_getButton(data)
43
44 classes = (NEC_8, NEC_16)
45 ir_remote = classes[0](alphanot.pin_IR, remoteNEC_callback)
46 alphanot.calibrateLineFinder()
47
48 def suivre_de_ligne():
49     if alphanot.readUltrasonicDistance() <= 5:
50         Contournement()
51     else:
52         if not alphanot.TRSensors_readLine(2) <= 300 and not alphanot.TRSensors_readLine(3) <= 300 and not alphanot.TRSensors_readLine(4) <= 300:
53             alphanot.moveBackward(15)
54         if not alphanot.TRSensors_readLine(2) <= 300 and not alphanot.TRSensors_readLine(3) <= 300 and alphanot.TRSensors_readLine(4) <= 300:
55             alphanot.setMotors(left=15)
56             alphanot.setMotors(right=0)
57         if not alphanot.TRSensors_readLine(2) <= 300 and alphanot.TRSensors_readLine(3) <= 300 and not alphanot.TRSensors_readLine(4) <= 300:
58             alphanot.moveForward(15)
59         if not alphanot.TRSensors_readLine(2) <= 300 and alphanot.TRSensors_readLine(3) <= 300 and alphanot.TRSensors_readLine(4) <= 300:
60             alphanot.setMotors(left=15)
61             alphanot.setMotors(right=5)
62         if alphanot.TRSensors_readLine(2) <= 300 and not alphanot.TRSensors_readLine(3) <= 300 and not alphanot.TRSensors_readLine(4) <= 300:
63             alphanot.setMotors(left=0)
64             alphanot.setMotors(right=15)
65         if alphanot.TRSensors_readLine(2) <= 300 and not alphanot.TRSensors_readLine(3) <= 300 and alphanot.TRSensors_readLine(4) <= 300:
66             alphanot.moveBackward(15)
67         if alphanot.TRSensors_readLine(2) <= 300 and alphanot.TRSensors_readLine(3) <= 300 and not alphanot.TRSensors_readLine(4) <= 300:
68             alphanot.setMotors(left=5)
69             alphanot.setMotors(right=15)
70         if alphanot.TRSensors_readLine(2) <= 300 and alphanot.TRSensors_readLine(3) <= 300 and alphanot.TRSensors_readLine(4) <= 300:
71             alphanot.moveBackward(15)
72         oled.fill(0)
73         oled.show()
74
75 def Contournement():
76     tourner_gauche_90_C2_B0()
77     avancer_10_cm()
78     tourner_droite_90_C2_B0()
79     avancer_10_cm()
80     tourner_droite_90_C2_B0()
81     avancer_10_cm()
82     tourner_gauche_90_C2_B0()
83

```

```
84 def tourner_gauche_90_C2_B0():
85     alphabot.setMotors(left=20)
86     alphabot.setMotors(right=-20)
87     utime.sleep_ms(450)
88     alphabot.stop()
89
90 def tourner_droite_90_C2_B0():
91     alphabot.setMotors(right=20)
92     alphabot.setMotors(left=-20)
93     utime.sleep_ms(450)
94     alphabot.stop()
95
96 def avancer_10_cm():
97     if alphabot.readUltrasonicDistance() > 10:
98         alphabot.moveForward(20)
99         utime.sleep_ms(400)
100        alphabot.stop()
101     else:
102         alphabot.stop()
103
104 def reculer_10():
105     alphabot.moveBackward(20)
106     utime.sleep_ms(400)
107     alphabot.stop()
108
109 utime.sleep(3)
110
111 while True:
112     utime.sleep_ms(150)
113     gc.collect()
114     if ir_current_remote_code == "enter_save":
115         alphabot.stop()
116     elif ir_current_remote_code == "up":
117         avancer_10_cm()
118     elif ir_current_remote_code == "down":
119         reculer_10()
120     elif ir_current_remote_code == "left":
121         tourner_gauche_90_C2_B0()
122     elif ir_current_remote_code == "right":
123         tourner_droite_90_C2_B0()
124     elif ir_current_remote_code == "play_pause":
125         suiveur_de_ligne()
126     else:
127         suiveur_de_ligne()
```

Per andare oltre

Per completare la programmazione di quest'opuscolo, è possibile aggiungere le seguenti attività nel corso dell'anno o su altri livelli in parallelo.

Realizzazione di oggetti in 3D:

- Mini cratere che si adatta al percorso
- Roccia
- Base Marziana
- Pannello

Un progetto più ambizioso potrebbe prevedere una base per accogliere il robot. Si può partire dall'idea di un garage con una porta automatica, questo permetterà di utilizzare il sensore ad ultrasuoni per rilevare la presenza del robot e comandare l'apertura o la chiusura del portello di entrata della base. Per questo progetto occorre programmare, ma anche progettare e fabbricare con macchinari, stampante 3D o taglio laser.

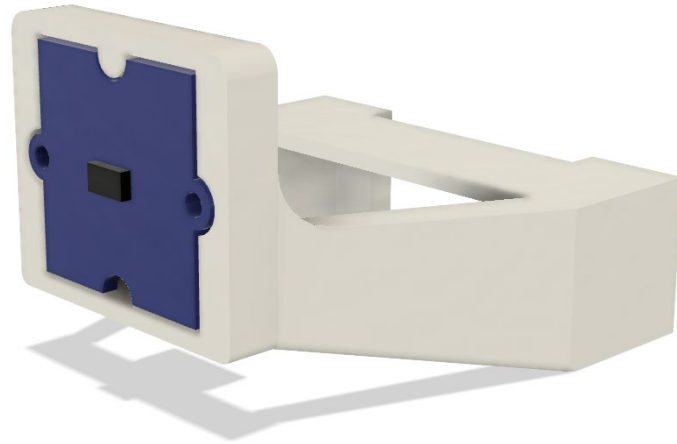
Un esempio di roccia marziana stampata in 3D: un ostacolo ideale per il sensore di distanza !



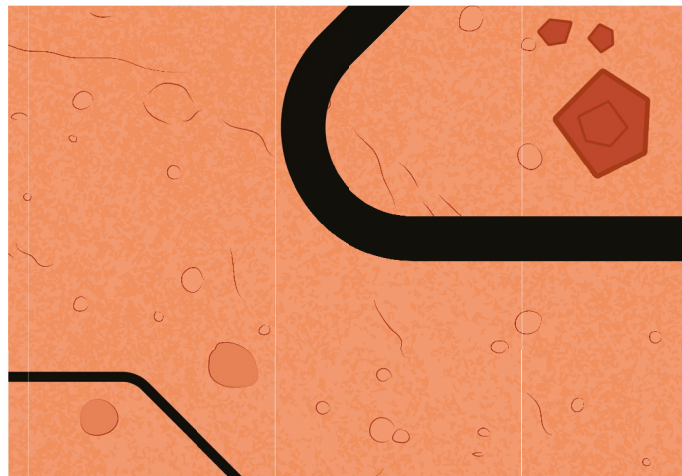
An example of a 3D printed Martian rock, which makes for an ideal obstacle for the distance sensor!

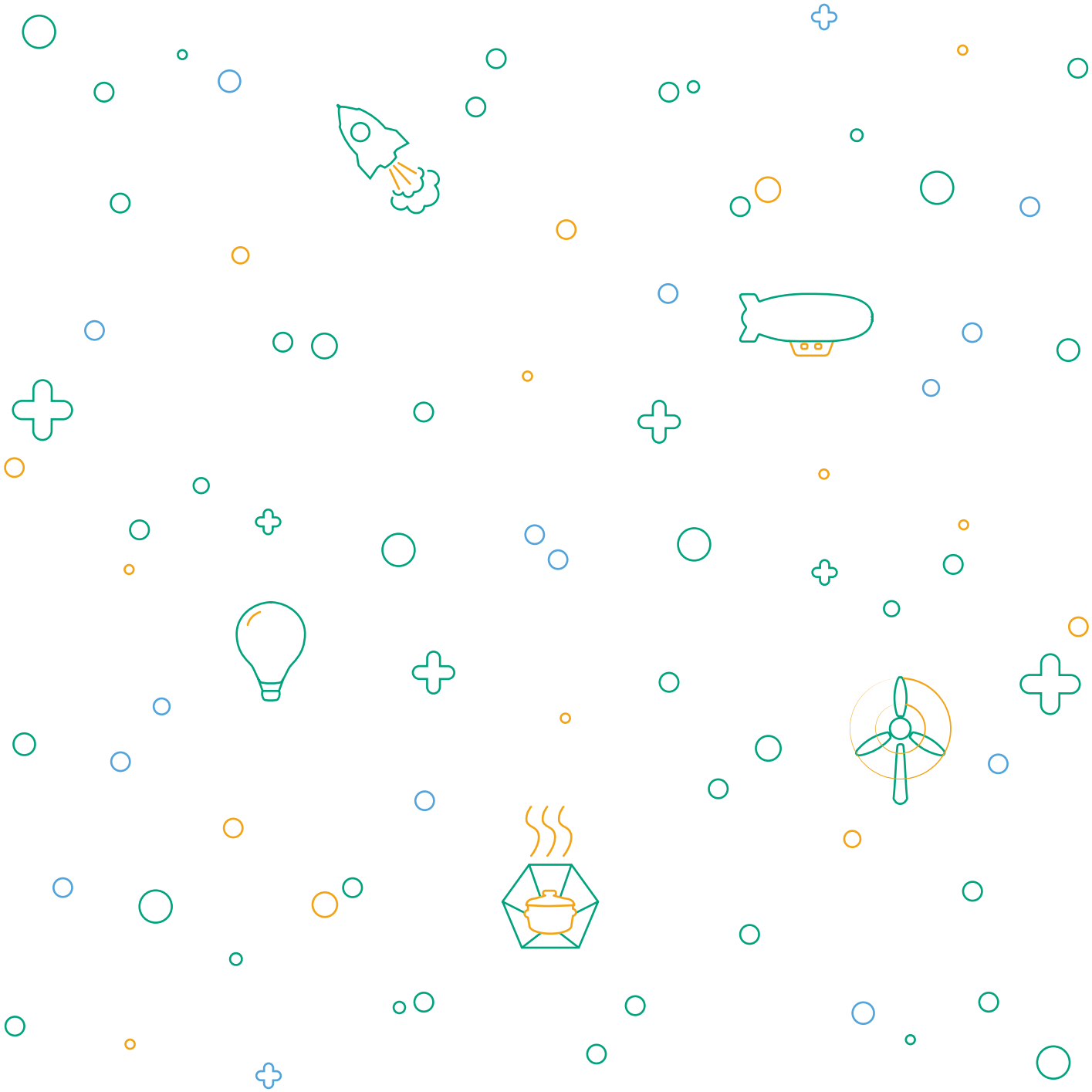
Pour aller plus loin

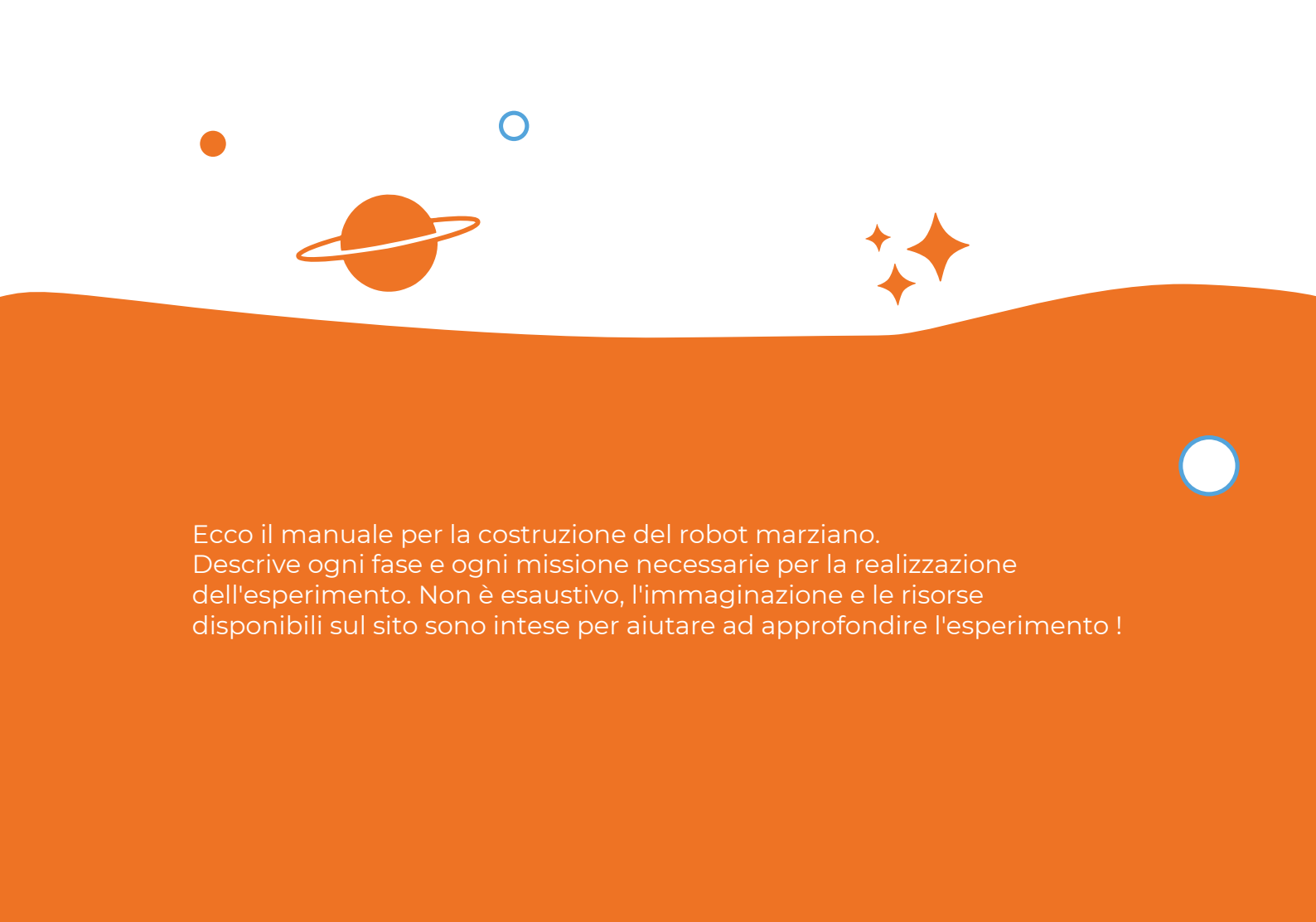
Se si desidera sostituire il sensore ad ultrasuoni con il sensore "Time of Flight", ecco un supporto da stampare in 3D. Trovalo sul sito vittascience.com/learn.



Per chiudere la traccia del robot marziano, stampa questo foglio in formato A4 e posizionalo sulla traccia. La trovi sul sito vittascience.com/learn







Ecco il manuale per la costruzione del robot marziano.
Descrive ogni fase e ogni missione necessarie per la realizzazione
dell'esperimento. Non è esaustivo, l'immaginazione e le risorse
disponibili sul sito sono intese per aiutare ad approfondire l'esperimento !

vitta
science



life.augmented



STEM your way
Innovation depends on you

 **IPCEI**
on Microelectronics