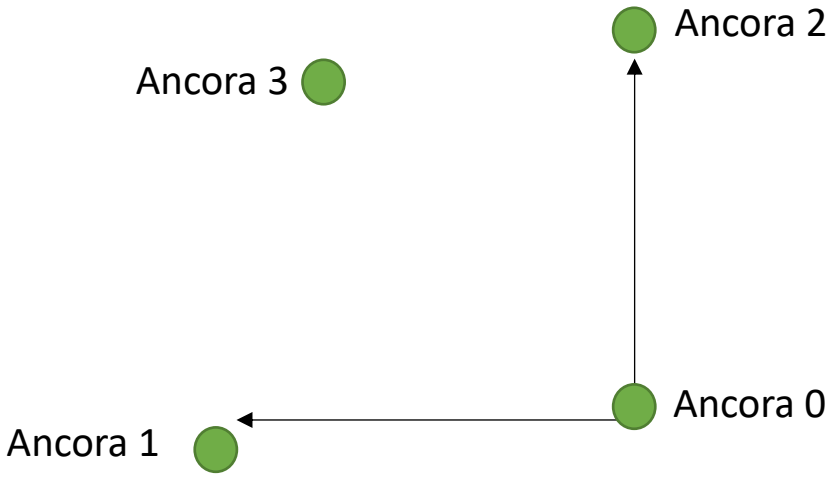
GUIDA ALL'ESPERIMENTO

POSIZIONAMENTO SENSORI

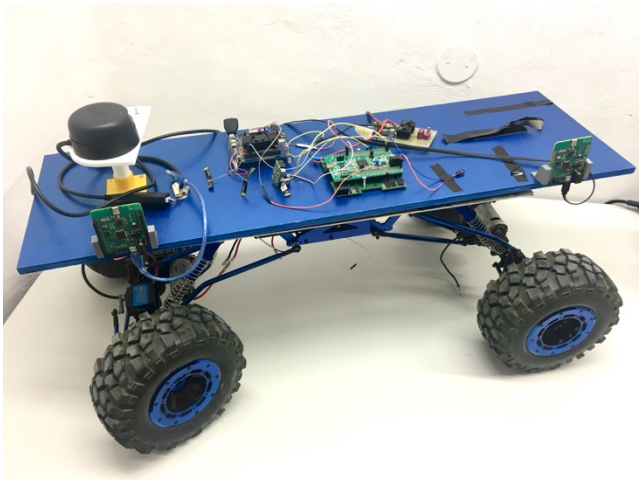
Posizionare le ancore UWB seguendo le seguenti convenzioni:

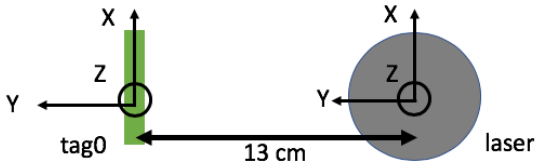
- l'ancora 0 è posta nell'origine del sistema di riferimento;
- l'ancora 1 definisce il verso positivo e la direzione dell'asse y:
- l'ancora 2 definisce il verso positivo e la direzione dell'asse x;
- l'ancora 3 definisce il verso positivo dell'asse z e deve essere posta ad una quota superiore alle altre tre ancore, in modo da avere una z positiva verso l'alto.

Le prime tre ancore devono essere complanari e giacere su un piano parallelo a quello del pavimento. Questa terna costituisce un sistema di riferimento fisso, rispetto al quale saranno espresse le coordinate della macchina.



Posizionare la tag0 (tag con il pettine) nella parte anteriore della macchina sull'apposito sostegno e la tag1 nella parte posteriore.
Posizionare il laser con l'asse x positivo rivolto verso la direzione di marcia della macchina.





COLLEGAMENTO BATTERIE

Collegare le batterie:

- 4S 14.8 V Li-Po 5000mAh per Intel Joule e STM32
- 7.2 V-NiMH 3000mAh per il driver del motore

CALIBRAZIONE UWB

Per eseguire la calibrazione delle ancore, sono possibili due soluzioni: la calibrazione automatica e quella manuale. Utilizzare un Pc con sistema operativo Ubuntu e connesso alla linea rosnet. In ogni finestra del terminale lanciare il comando:

```
>>  ssh -X robot@192.168.20.1
```

e immettere la password: robot
Questo comando serve per collegarsi via ssh all'Intel Joule.

Autocalibrazione

Lanciare da terminale:

```
>>  cd catkin_ws/src/icaro_uwb/src
```

```
>>  gedit get_uwb_info.py
```

Aspettare che si apra il file e modificare il parametro "heigth" riportando l'altezza (espressa in mm) dell'ancora z rispetto al pavimento.

```
>>  source venv/bin/activate
```

Con questo comando si entra in un ambiente virtuale python.

```
>>  cd Desktop/crosato_tesconi/Script\di\base
```

```
>>  gedit autocalibration_ransac.py
```

Aspettare l'apertura del file e settare a "TRUE" il parametro "auto_cal".

```
>>  python autocalibration_ransac.py
```

In questo modo si attiva il processo di autocalibrazione. Al suo termine verrà fatto un print delle coordinate delle ancore nel sistema di riferimento sopra descritto. Verrà chiesto se si vuole ripetere l'autocalibrazione (premere "y" o "n" in base ai risultati ottenuti) e successivamente verrà chiesto se salvare i risultati ottenuti (premere "y" o "n").

```
>>  deactivate
```

Per uscire dal virtual enviroment python.

Calibrazione manuale

```
>>  cd catkin_ws/src/icaro_uwb/src
```

```
>>  gedit get_uwb_info.py
```

Aspettare che si apra il file e modificare il parametro "heigth" riportando l'altezza (espressa in mm) dell'ancora z rispetto al pavimento.

```
>>  cd
```

```
>> cd Desktop/crosato_tesconi/Scrpt di base

>> gedit autocalibration_ransac.py
```

Aspettare l'apertura del file e settare a "FALSE" il parametro "auto_cal" e inserire le distanze relative fra le varie ancore (in mm). La distanza tra l'ancora i e l'ancora j deve essere specificata scrivendo r_{ij}=distanza in mm, ricordando che 0=ORIGINE, 1=Y, 2=X e 3=Z.

ACQUISIZIONE MAPPA

Il sistema di riferimento del laser all'istante iniziale dell'acquisizione della mappa rappresenterà il sistema di riferimento fisso della mappa stessa.
Lanciare i seguenti comandi:

```
>> roslaunch lidar_cam_mapping sensors.launch
```

Si accende il laser. Per controllare il funzionamento effettivo del laser si può mandare il comando "rostopic echo /scan" e verificare che stia pubblicato sul topic.

```
>> roslaunch icaro_uwb uwb.launch
```

Lancia il nodo ros per l'acquisizione delle posizioni delle tag. Nel codice, le tag sono contraddistinte da un id (0 ed 1) a seconda della porta a cui sono associate: la tag con id X è associata alla porta ttyACMX (si vede cercando fra i nomi dati come output digitando il comando "Is/dev"). Di solito la tag che si collega per prima all'Intel Joule ha id=0; la seconda id=1. Se per qualche motivo gli id cambiano, provare a staccare entrambe le tag e ricollegarle. Se il problema sussiste cambiare gli argomenti passati nel launch file. Le posizioni delle tag sono pubblicate rispettivamente su "tag_pose0" e "tag_pose1". Entrambe sono dei geometry_msgs/Point e contengono le misure (x,y,z) espresse in mm. Una volta mandato il launch file vengono stampati a schermo le coordinate della tag0 e della tag1. Se vengono stampati tutti valori nulli o "failed position" interrompere il nodo e rilanciarlo. Nel caso in cui il problema sussista, ricontrollare il posizionamento e l'alimentazione delle ancore e delle tag.

```
>> rosrun amcl tf_tag_map
```

Prima di iniziare a spostare la macchina (e quindi il laser) mandare questo nodo per acquisire la trasformata dalle ancore all'origine del sistema di riferimento della mappa. La trasformata acquisita viene stampata a schermo. I valori di questa devono essere inseriti dall'utente nel nodo "initialpose" come descritto successivamente. In alternativa possono essere misurati a mano.

```
>> roslaunch lidar_cam_mapping hector_mapping_BIPE.launch
```

Questo comando permette di iniziare ad acquisire la mappa. Se durante l'acquisizione non viene utilizzata una Imu, il laser non può essere ruotato durante l'acquisizione.

```
>> cd catkin_ws/src/maps

>> rosrun map_server map_saver -f name
```

Comando per salvare la mappa acquisita. Viene salvato un file .yaml con le caratteristiche della mappa ed un file .pgm, con l'immagine della mappa nella cartella maps in src con il nome "name".

SETTAGGIO DELLA TRASFORMATA TRA TAG E MAP

```
>> cd catkin_ws/src/navigation/amcl/src
>> gedit initalpose
```

Aprire il file e impostare le variabili map_x map_y map_z rispettivamente con le componenti di traslazione e di rotazione che compongono la trasformata fra il sistema di riferimento tag ed il sistema

di riferimento mappa. Queste componenti sono state mandate a schermo durante l'acquisizione della mappa.

```
>>  catkin_make
```

Ricompilare in modo da salvare le modifiche. Nel caso in cui le modifiche non siano state salvate e viene riportato il messaggio "your build may be incompleted", collegarsi ad internet con la Intel Joule, in modo che l'orario riportato sullo schermo sia corretto. Successivamente riconnettersi a rosnet, cancellare la cartella "build" all'interno del catkin_ws e ripetere il comando catkin_make.

LOCALIZZAZIONE AMCL

```
>>  roslaunch lidar_cam_mapping sensors.launch
```

Comando per avviare il laser.

```
>>  cd catkin_ws /src/maps
```

```
>>  rosrun map_server map_server name.yaml
```

Con questo comando viene caricata la mappa acquisita in precedenza.

```
>>  roslaunch icaro_uwb uwb.launch
```

Vengono avviate le tag.

```
>>  rosrun serial_manager Serialmanager
```

Attiva la comunicazione via seriale tra Intel Joule e STM32.

```
>>  rosrun amcl tagff2
```

Nodo per pubblicare la trasformata dal frame odom al frame base_link. Con questo nodo viene pubblicata l'informazione che solitamente si ottiene da un sensore di odometria.

```
>>  roslaunch amcl amcl_ff2.launch
```

Launch per avviare il funzionamento di amcl. Nel file "amcl_ff2.launch" si possono settare i parametri che riguardano amcl. In questo file viene settata anche la trasformata statica dal frame "laser" al frame "base_link_ff2", che è stato centrato nella tag0 con assi paralleli a quelli del frame laser.

```
>>  rosrun amcl initalpose
```

Nodo per settare la posa iniziale del robot, questo può essere fatto anche manualmente su Rviz tramite il comando 2D Pose Estimate.

SETTAGGIO GOAL DA RAGGIUNGERE

```
>>  rosrun amcl amcl2robot_pose
```

Nodo per inviare la posizione della macchina stimata da amcl all'STM. Le coordinate vengono inviate tramite un messaggio di tipo Point il quale ha tre valori: x,y,z. Nel primo viene salvato il valore dell'ascissa della macchinina stimato da amcl nel sistema di riferimento mappa, nel secondo viene salvato il valore dell'ordinata e nel terzo l'orientamento.

```
>>  rosrun amcl navgoal
```

Nodo per poter settare da schermo il waypoint che si vuole far raggiungere alla macchina. Le coordinate di questo punto (ascissa e ordinata) vengono comunicate all'STM.
Usare il comando 2D Nav Goal su Rviz per indicare il punto sulla mappa che si vuole far raggiungere alla macchina oppure pubblicare a mano le coordinate di tale punto con il comando:

```
>>  rostopic pub /move_base_simple_goal
```

```
>>  rostopic pub /star_stop
```

Pubblicare il valore 1 su questo topic se si intende far partire i motori, 0 quando si vuole far fermare la macchina (comando utile per far fermare la macchina, nel caso in cui qualcosa non dovesse funzionare).

REGISTRAZIONE FILE CON ROSBAG

Per poter registrare un bag file, prima deve essere montata una pennetta usb sulla quale salvare il file registrato.

```
>>  mount /dev/sda1
```

```
>>  cd USB
```

Comando per visualizzare i file contenuti nella pennetta. Questo è utile per verificare che la pennetta sia stata montata correttamente.

```
>>  rosbag record -a
```

-a (sta per all) per registrare tutti i topic, per registrarne uno, specificarne il nome.

```
>>  control + C
```

Per interrompere la registrazione.

UTILIZZO DELLA STM TRAMITE MATLAB

All'interno della cartella "BIPE MATLAB" è presente il file Matlab utilizzato all'interno della STM. Aprire su matlab 2014a:

```
>> ICARO3_orientation_controllo_BIPE
```

Il file della telemetria è utilizzato per vedere in tempo reale i dati ed eventualmente i loro grafici. Aprire su matlab 2014a:

```
>> ICAROIII_debug_telemetry
```

COMANDI UTILI

- Nel caso in cui si voglia spengere la macchina da ssh, digitare:

  ```
  >>  sudo shutdown
  ```

  mettere la password ed attendere che la macchina si spenga.

- Per poter avviare una registrazione utilizzare il comando:

  ```
  >>  rosbag play record name.bag -clock
  ```

- Nel caso in cui si voglia utilizzare amcl in simulazione, settare questo parametro a true:

  ```
  >>  rosparam set use_sime_time true
  ```

- Per configurare l'Rplidar A3 al posto dell' A2:

  ```
  >>  cd catkin_ws/src/lidar_cam_mapping/launch/sensors
  ```

  ```
  >>  gedit sensors.launch
  ```

  Impostare il parametro della porta seriale con `value="/dev/rplidarA3"` al posto di `value="/dev/rplidarA2"/>`.

Impostare il baudrate a `value="256000"` al posto di `value="115200"`.

```
>>  cd

>>  cd catkin_ws/src/rplidar_ros/src

>>  gedit node.cpp
```

Impostare il parametro “`serial_baudrate`” a `256000` , al posto di `115200` .

```
>>  catkin_make
```

Per salvare le modifiche.