POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Tesi di Laurea Magistrale in
Computer Science and Engineering
Ingegneria Informatica

Title of the thesis

Author:
Name Surname

Student ID:
XXXXXX

Advisor:
Prof. Name Surname

Academic Year:
20xx-xx

Dedicated to my family.

Abstract

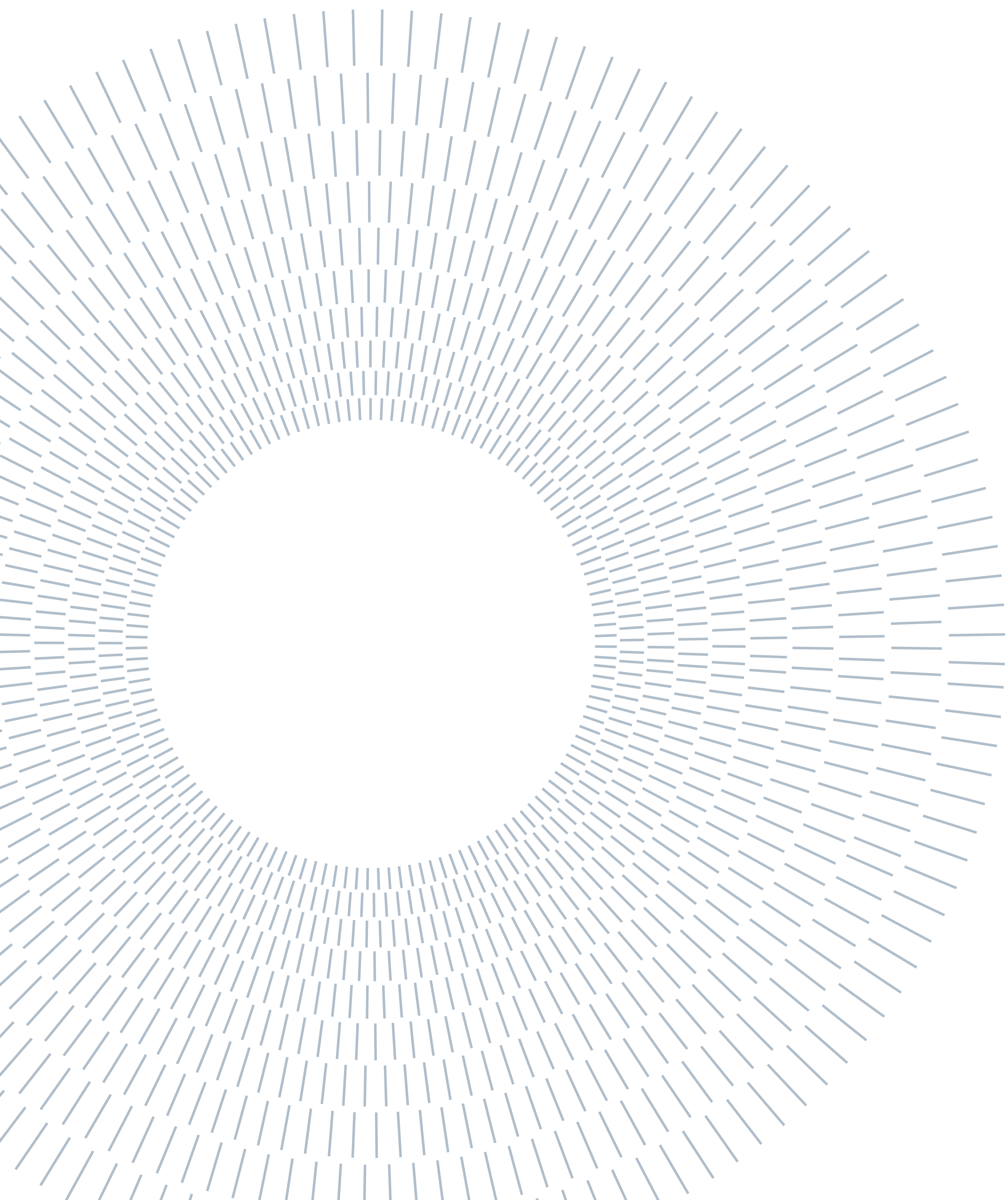
Here goes the abstract.

Keywords: key, words, go, here

Abstract in lingua italiana

Qui va inserito l'abstract in italiano.

Parole chiave: qui, vanno, le, parole, chiave

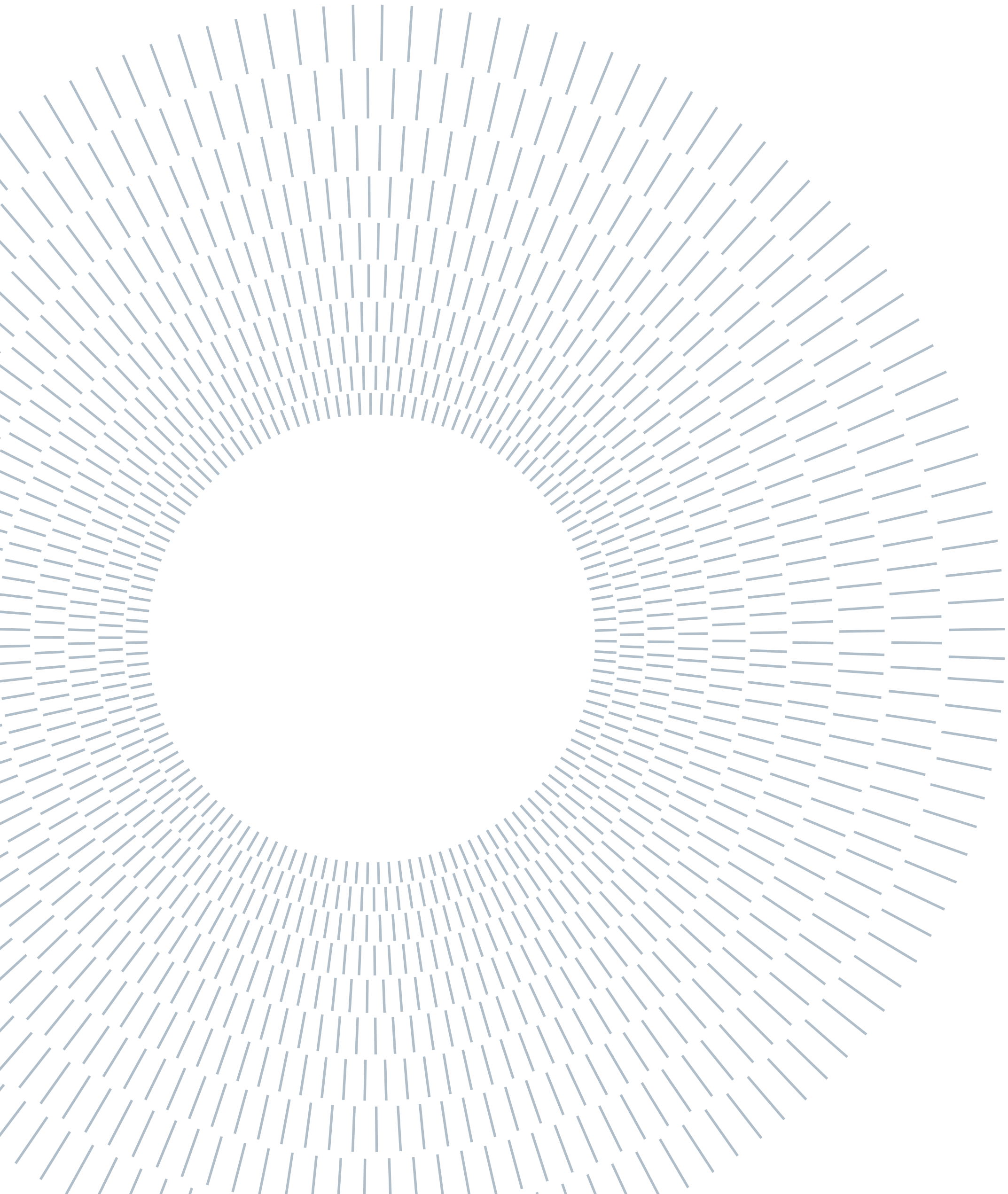


Contents

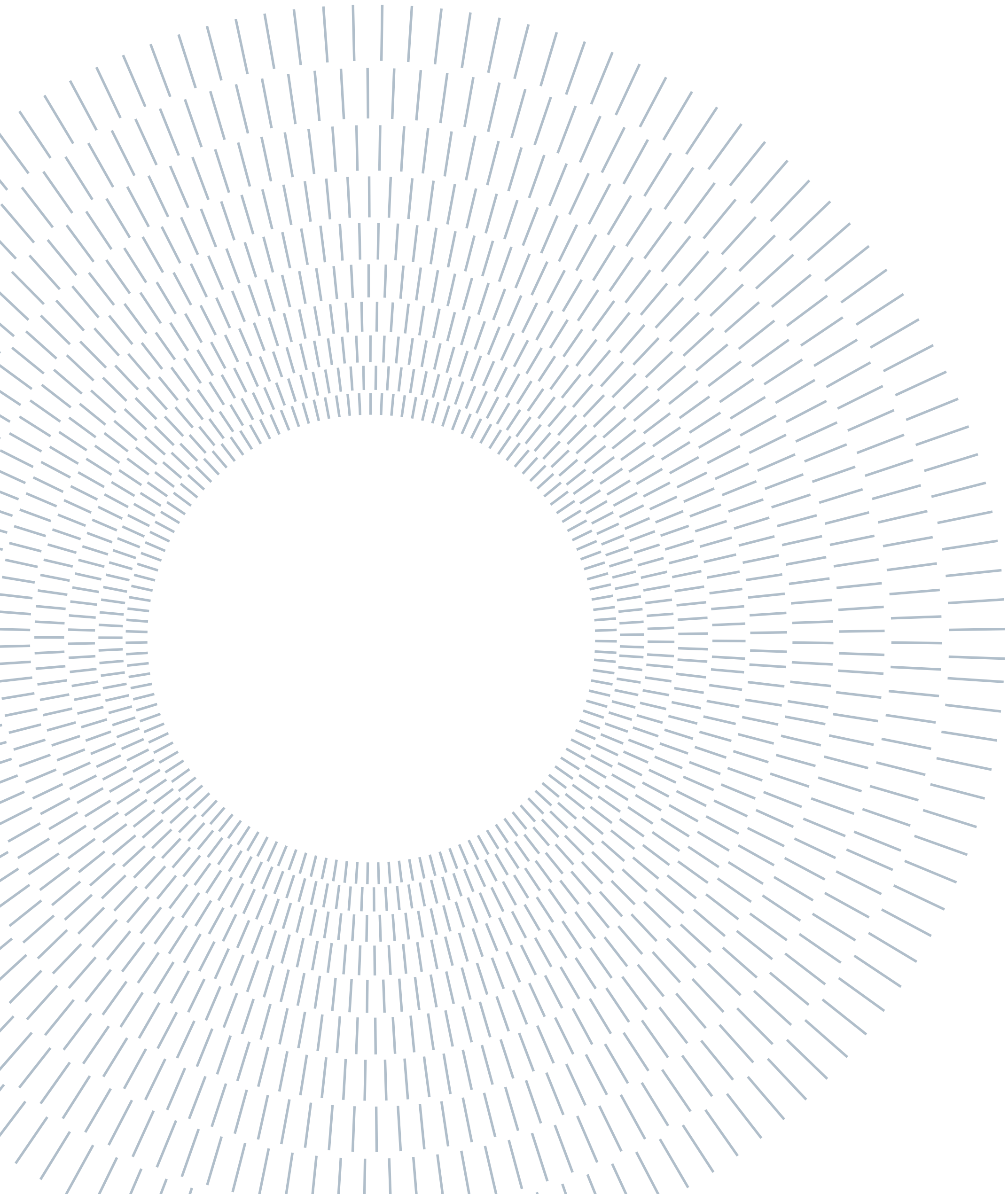
Pre Tino goal

SLAM, process and RD

General sensor stuff (speaker, mic, VR

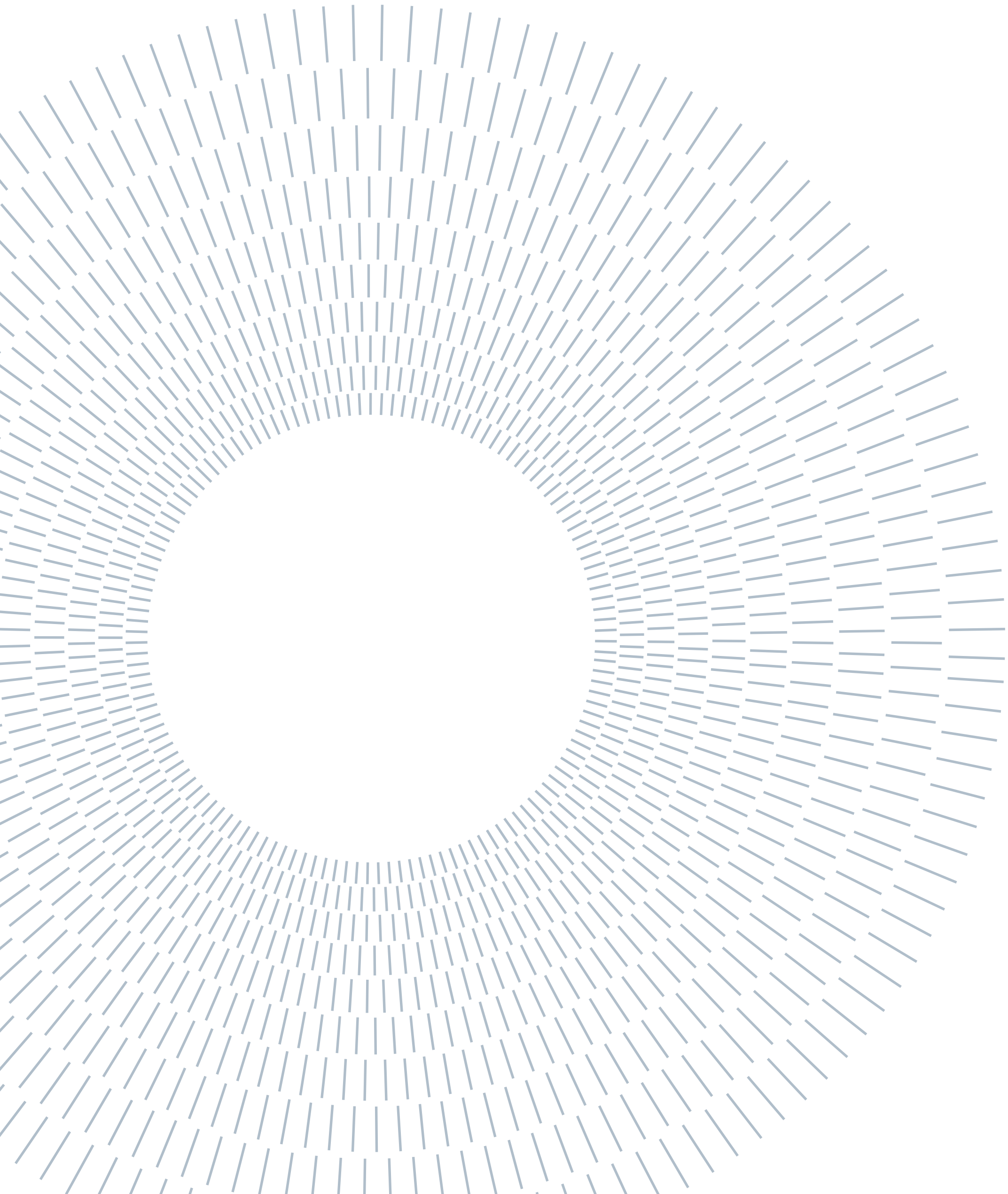


Legacy to ROS2 migration ROS arquitecture



Physical changes

Testing



1 | Temporal R&D

Localization Technologies

Onboard Sensing

Technology	**Pros**	**Cons**	**Key Papers & Resources**
Visual Odometry (VO)	Could use existing camera; no hardware mods	Narrow FOV; tilt disrupts SLAM	ORB-SLAM3 (Campos et al., 2021) – Robust monocular/Stereo SLAM. SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems
IMU + Wheel Encoders	Low cost; integrates motion data	Drift over time; Stewart tilt issues	Sensor Fusion for Mobile Robot Localization (Huang et al., 2018) – Kalman filtering. EKF Sensor Fusion in Practice for Mobile Robot Localization
UWB-IR	Small footprint; could work with fabric	Requires external anchors	A UWB-IR based Localization System for Indoor Robot Navigation.

External Sensing

Technology	Pros	Cons	Key Papers & Resources
UWB Anchors	High accuracy; no line-of-sight	Setup/calibration required	Robot vision ultra-wideband wireless sensor in non-cooperative industrial environments
AprilTags	Low cost; precise	Line-of-sight; limited area	AprilTag: A Robust Fiducial System (Olson, 2011).
MoCap Systems	Sub-mm accuracy	Expensive; fixed environment	OptiTrack for Robotics – Industrial use cases.

Orientation Technologies

- **Sensor Fusion**: A Review of Sensor Fusion Techniques filters for combining UWB, IMU, and encoders. (Waiting for access request)
- **UWBOri**: enabling accurate orientation estimation with ultra-wideband signals
- **NLOS Mitigation**: UWB System for Indoor Positioning and Tracking With Arbitrary Target Orientation, Optimal Anchor Location, and Adaptive NLOS Mitigation
- **RPO**: Measurement of Relative Position and Orientation using UWB

Human Detection Technologies

Onboard Sensing

Technology	**Pros**	**Cons**	**Key Papers & Resources**
Thermal Cameras	Works in darkness; fabric-friendly?	No depth; limited range	Thermal Human Detection – CNN-based approaches.
Ultrasonic Array	Low cost; proximity detection	No human distinction	Ultrasonic Human Tracking (In Korean).
Upgraded Camera	Wider FOV; ML-compatible	Fabric obstruction; compute-heavy	YOLOv7 (Wang et al., 2022) – Real-time object detection.

External Sensing

Technology	**Pros**	**Cons**	**Key Papers & Resources**
RGB-D Cameras	Depth data; multi-human tracking	Fixed installation	OpenPose: Real-Time Human Pose (Cao et al., 2019). Azure Kinect for Robotics – Performance Analysis of Body Tracking.
LiDAR	High-resolution 3D mapping	Expensive; compute-heavy	LiDAR-based Human detection (Zhi Yan et al., 2019).
WiFi/Radar	Privacy-friendly; fabric-penetrating	Lower resolution	RF-Sensing: A New Way to Observe Surroundings

Technologies for Tino Robot Implementation

Localization Technologies

Visual Odometry (VO)

- **Variants:**
  - *ORB-SLAM3* (supports RGB-D): GitHub
    + Synergy with human detection via depth data
    + Robust feature matching for dynamic environments
    − Higher computational cost (requires GPU optimization)
  - *SVO* (Semi-direct Visual Odometry):GitHub
    + Works with fisheye/catadioptric cameras (wide FOV)
    + Lower computational footprint
    − Less accurate in textureless environments
- **Shared Advantage:** Dual-purpose for localization & human detection

UWB-IR Localization

+ Centimeter-level accuracy (theoretical)
+ Low power consumption
− Requires external infrastructure (anchors)
− Fabric penetration uncertainty (needs RF testing)
− No native orientation data ⇒ Requires:
  - IMU sensor fusion (Kalman filtering)
  - RPO/UWBOri techniques (experimental)
  - NLOS mitigation strategies

Wheel Encoders + IMU

+ Low-cost solution
− Unsuitable for impulse-based movement (slippage errors)
− IMU drift accumulates over time
− Poor performance on uneven surfaces

Human Detection Technologies

RGB-D Camera (e.g., Intel RealSense)

+ Simultaneous color + depth data
+ Enables skeleton tracking (OpenPose, MediaPipe)

  - – Requires careful physical integration (size/visibility)
  - – Limited range (typically $<$5m)

Thermal Imaging

  - + Potential fabric penetration capability
  - + Works in low-light conditions
  - – No depth sensing $\Rightarrow$ Requires fusion with VO
  - – Limited contextual information (heat-only data)

ML-Enhanced 2D Camera

  - + Lower profile than RGB-D
  - + Modern architectures (YOLOv8, EfficientNet) enable real-time detection
  - – Requires depth estimation via:
    - – Monocular depth networks (MiDaS, LeReS)
    - – Sensor fusion with other localization data

Lidar

  - – Impractical due to Tino's soft structure (vibration issues)
  - – High cost-to-benefit ratio
  - – Overkill for indoor social robot ranges

Recommended Hybrid Approach

- **Localization:** ORB-SLAM3 with RGB-D camera (despite computational cost) + optional UWB for absolute positioning
- **Human Detection:** Thermal camera + RGB-D fusion (if concealable) or ML 2D camera with monocular depth estimation
- **Backup:** SVO with fisheye lens as fallback if RGB-D integration fails

Week 18 Mar RD on different techs

Week 25 Mar Task: Work on Orin nano testing cameras ZED 2, and Orb slam 3 with webcam and Realsense T265 Result: The Zed camera that was available was not functioning properly, orbslam had a lot of bugs in terms of compilation given is an old library that is not being maintained.

Week 1 Apr Task: Work on Orin Nano and Relsense T265 to try and make SLAM atlas creation and load Result: The T265 was deprecated so I had to install an old version of librealsense (2.53) in order to make the camera be detected, even after camera detection I was able to run the camera with orbslam but the accuracy was very low, my initial tought is that it was because of poor calibration. Orbslam had some issues with the camera, in stereo inertial was the best mode that it worked but it needed some acceleration in order to start outputing some video, also it took a long time to actually grap into something (features) so to actually start creating a map, in only stereo it crashed, same as in Mono

Week 08 Apr Task: keep working on Realsense T265 and most important save and load atlas Result: even tho it had a lot of issues the first thing that was tested is calibrating the camera to see if the detection improved, it didnt, then i tried saving the atlas but given the old library it always ended in a crash. After looking on the web I found a git fork that "fixed" this atlas save and load, testing the library i found that it managed to save but it always crashed when trying to open the atlas back, either that or it starts creating a new map from scrathc. Given all of the issues that the orbslam3 had i tried using SVO, it had again a lot of issues given its an old not maintained library, mainly in the compilation part as i am working in arm64 so i had to fix a lot of flags in order to make it compile in arm64. even after all of the work trying to make it build in a container i had a same result that with orbslam, loaded the map, mapped something (not that accurrate) and they did not have any atlas/map management so I scraped that work. Given that with 2 systems i had simmilar issues i tought it could be related to the Realsense T265, so I requested if a D435I was available (given that the video examples used in orbslam3 are with that camera), in the end that camera was not available but I was provided with a oak-d pro, after testing the basic functionallity with the depthAi library I tried checking for slam approaches, they had a community fork of orbslam3 using that camera and a guide to try and build it with lxc, but after trying a lot I was not able to pass the camera to the container in a way that it was detected as a bootable device. One of the other options Luxonics mentrioned was using RtabMap, so thats what I was set to try

Week 15 Apr Task: Try to set the Oak-D pro to work with Rtabmap Result: The first thing I had to do was try to build the library, it had a lot of dependencies and with that a lot of issues to be fixed, the first time I tried to build the standalone version, this didnt work at first because the depthAI library was not detected. Then I tried to build the Ros version of rtab but this one had not a proper implementation between depthAI and the ros wrapper

Then I tied again with the standalone version and this time I was able to make it link with the depthAI, and it worked amaizing, I did some tests with the camera over Tino moving around, this showed that Rtab was working really well creating a map and most importantly it was able to save a map and then relocalize itself in that map. Now the next step was how to get the data out of the standalone version, how to get the position and orientation.

After some trial an error managed to install and run it with ros2 using the depthai$_{r}osandthertabmap$

Managed to load the correct map, I refactored the old Tino source code to work with Ros2, created the respective topics and the needed structure. Created respective launch files for mapping and localization modes Added human detection, this system works by subscribing to the same camera topic and run it with yolo11 in tensorRt format. This provides all of the information needed for the human detection getting all of the skeleton pose joints, getting the depth (using the stereo camera info) and position in relation to the robot.

Also by creating this ros version I created a node for handling the VR connection in the future.

Apr 22 Next task was adding audion in/out to the system. I was provided with a omnidirectional mic iTalk-01, and a pair of speakers. Impelemented a system that gets the data and publishes it to the vr, and also receives from the vr and publishes to the speakers.

At this point most of the internals where ready We bought a display port dummy in order to have good performance when connected via vnc because the Orin Nano does not run headless by default

One of the head supports broke so we had to print a new one with more internal support

Next steps is hardware related. we need to: Build the new kinematic base that can support tino weight Fix the head supports Add the power supply needed to support the Orin Nano

Apr 29 Started by dissasembling the robot completely into the main 4 parts Fabric head Servo Head Body Kinematic base

First I modified the Servo head by adding a trypod that can hold the camera, this was done with simple brackets to make the support fixed and steady, specially because the old camera mount (that was for a pi camera) was very very flexible and moved a lot Then I tested the power supply, we got a powerful and stable 12v to 19v DC DC step up converter Oumefar, using this proved and testing the Orin Nano at max power, so with all of tino system actives (SLAM, audio, ROS) it reached a max of 2A of consumption, this from a 12v battery They are 5200 mAh 80c 11.1v 57.72Wh gave a approximate time of 1.37 hours during max consumption, but this really is not accurate as the jetson usually works between 1.3 to 1.4 A https://chatgpt.com/c/68125c25-216c-8000-a956-52b2702d04b8

Given that this will fix the power supply issue I modified the cable harness to remove the old USBA and USBC that powered the Raspberry from a powerbank, and replaced it with the 12V input and the 19V DC jack the Orin needs Also we added a 12v to 5V converter connected to the same 12v battery to power the Onboard router and the Oak-D camera the camera can be powered by the orin but we wanted to leave the option to power it directly if we wanted in the future to add the machine learning algorithms inside the camera. Also doing this change helped us remove the powerbank that was dedicated to the router, helping the total process of turning tino on and reducing from 4 batteries to 3 batteries

(couldn't do more because the week had thursday and Friday as holiday)

May 6 This week started on upgrading the kinamtic base. given that tino had an omniwheel base and tino is almost 20KG the wheels that it have where breaking apart and getting stuck, the rollers of the wheels where getting squared out. this also because given the movement, the back wheel of the triksta base was most of the time being dragged, as tino only moves forward and turn side to side

given this issues we decided to remove the omnidirectional triksta base as this movement is not needed, we decided a simple but reliable differential drive system, using 2 wheels at the front and a caster wheel in the back. To start this process we decided to just modify the base instead of changing it, given that it already had most of the things we needed. We removed the 3 motors, replaced them with 2 more powerful motors, given this new motors whe changed the old 2 motor drivers with a new and more powerful mdd10a.

We built the T structure using Aluminium profiles, item, this allowed us to have a dynamic and regulable system where we can extend out the wheels to try and get a proper balance. One of our main issues where the wheels we started by using plastic wheels that had a rubber neumatic, this worked first but then when tino was built it created an issue

Once the new base was rebuilt I had to modify the code for it to work, the original base used the VirHas library (custom internal library of the airlab to manage and control the triksta bases) so given this used a differential drive I had to implement my own PID movement controller (Proportional–integral–derivative controller) but keeping the commands the same in order to keep the original tino movement

Once the base was ready I started rebuilding tino, removing things that where not needed and adding the new things and new cable harness. I also added the speakers in the servo head because it had enough space and the microphone was passed through the fabric on the head

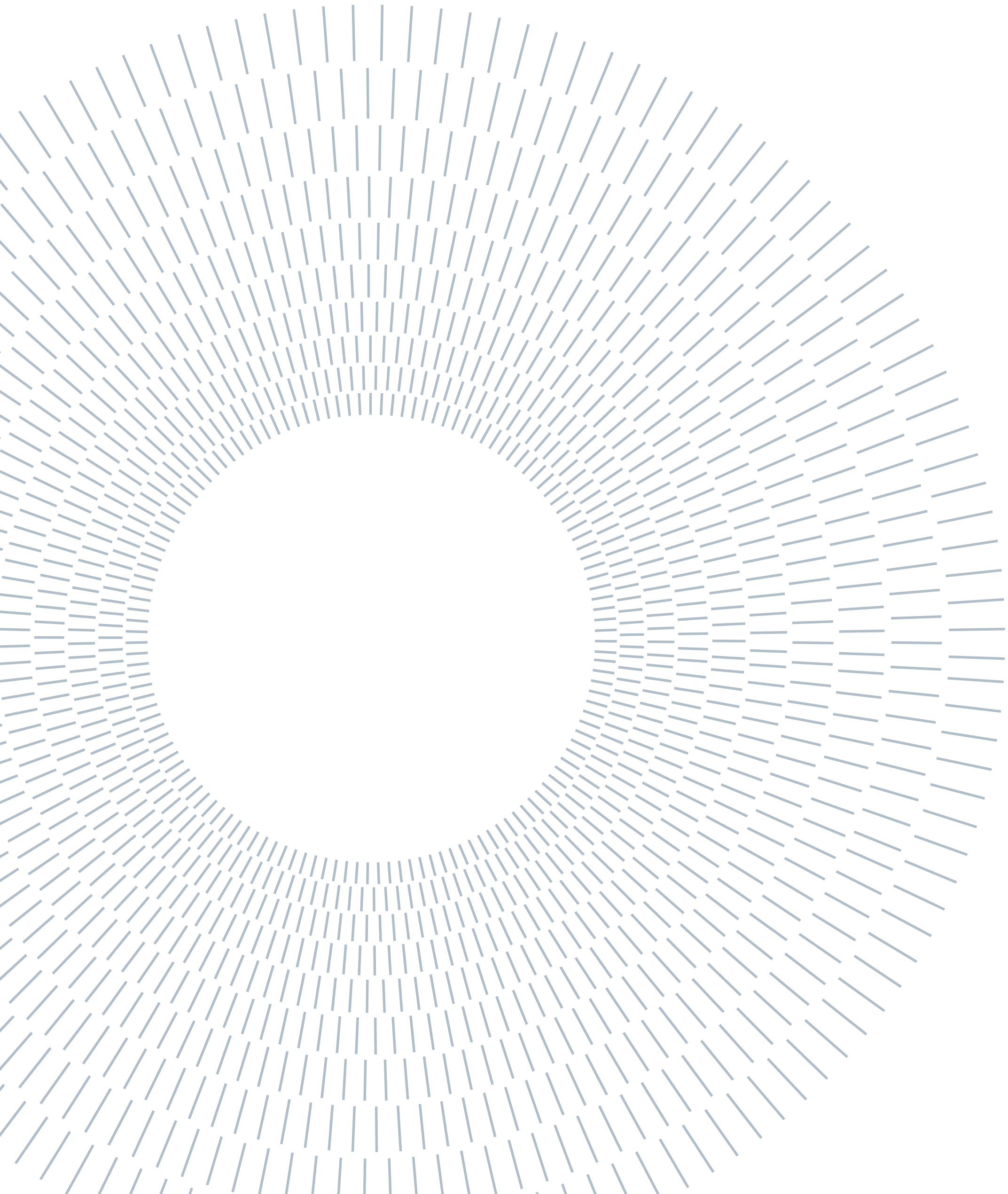
Then, once built, I had to test the connection from the arduinos to the Jetson, this proved to have some issues, the head Arduiono was a az-delivery arduino mega clone, but the jetson does not had the needed drivers (CH340) the only solution was to rebuilt the kernel of the jetpack system so to include it, given that this would be time consuming I decided to change that arduino mega with a Arduino elego uno r3. this one properly linked and connected to the jetson, the change did not create any issue as the head only used 3 PWM pins for the servos. I also setup a symlink using the serial of the devices so that when connected they always be in the same route /dev/ttyHEAD /dev/ttyBASE /dev/ttyLEG

once all systems where working again and some fine tunning had to be done to the gamepad (we changed from D input to X input because the jetson did not had the drivers to manage the D input) tino was working once again. The next step was going to test the wheels, the wheels we had put, given the weight of the robot the tire Partially de-beaded. consulting this issue we had 3 approaches to take next week 1. Fill the current wheels with hotglue, easy but could cause issues with the traction of the tire 2. use some hard plastic wheels but is demanding in labor because this wheels do not have the 6mm axis needed to connect to the motor axis, so we will need to modify them a lot in order to connect properly 3. buy a new pair of wheels that can support this weight better

We also encountered some issues with the fabric enrolling over the wheels so we may need to add a type of "bumper" in order to avoid this

Also we need to find a way to make the camera avoid the fabric, or better said, the fabric to avoid moving over the camera FOV I tried sticking the fabric to a foam external shell I put over the camera but this velcro was not sticking to the fabric given the camera is behind the leg, and this side of the robot moves the fabric a lot. Also using this foam to make the shell was not ideal because it was absorbing the camera heat and not letting the camera cooldown.

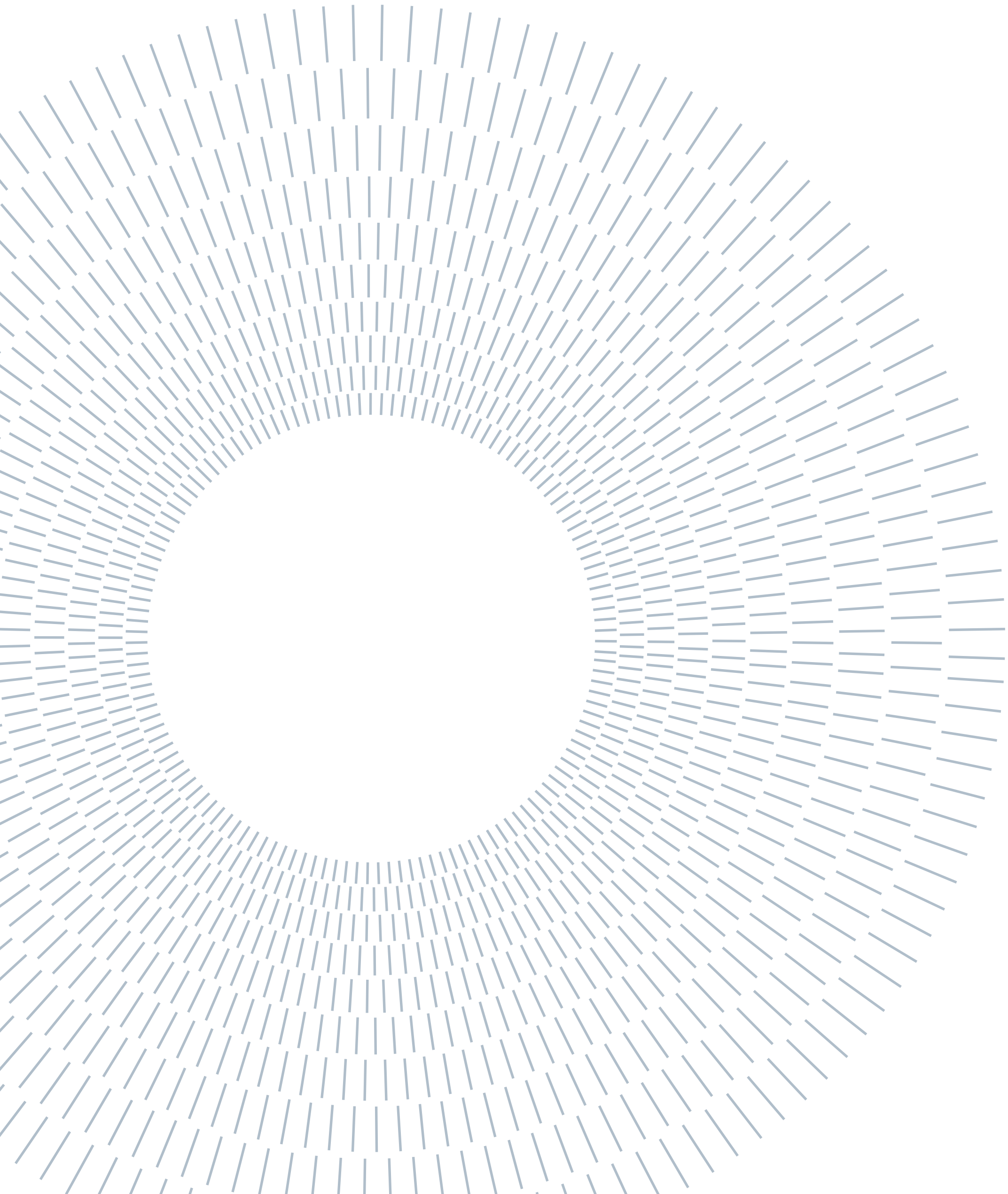
this are the issues to solve by next week



Bibliography

List of Figures

List of Tables



Acknowledgements

Here you may want to acknowledge someone.