



POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Tino V2, the revenge of the maze (Temporal Title)

TESI DI LAUREA MAGISTRALE IN
COMPUTER SCIENCE AND ENGINEERING
INGEGNERIA INFORMATICA

Author: **Sebastian Enrique Perea Lopez**

Student ID: 10986638

Advisor: Prof. Andrea Bonarini

Co-advisors: Federico Espositi

Academic Year: 2024-25

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

Abstract	i
Abstract in lingua italiana	iii
Contents	v
1 Introduction	1
1.1 Social Robotics and Human-Robot Interaction	1
1.2 The Tino Robot Project	1
1.3 Project Objectives	2
1.3.1 Technical Objectives	2
1.3.2 Research Objectives	3
1.4 Thesis Structure	4
2 Background	5
2.1 Simultaneous Localization and Mapping (SLAM) Technologies	5
2.1.1 Feature-Based SLAM Systems	5
2.1.2 Direct and Semi-Direct SLAM Systems	6
2.1.3 Graph-Based SLAM Systems	6
2.1.4 Ultra-Wideband Positioning Technology	7
2.1.5 LiDAR-Based SLAM	7
2.1.6 Sensor Fusion Approaches	8
2.2 Human Detection and Pose Estimation Technologies	8
2.2.1 Computer Vision-Based Human Detection	8
2.2.2 Alternative Sensing Modalities	9
2.3 Social Robotics and Human-Robot Interaction Foundations	10
2.3.1 Non-Verbal Communication in Robotics	10
2.3.2 Telepresence and Mediated Interaction	11
2.4 VR Integration in Robotics	11

2.4.1	Immersive Teleoperation Systems	11
2.4.2	Challenges in VR-Robot Integration	11
2.5	Legacy Tino System Analysis	12
2.5.1	Original System Architecture	12
2.5.2	Software Architecture Limitations	12
2.5.3	VR Integration Requirements	13
2.5.4	Identified Enhancement Requirements	13
3	Conceptual Work	15
3.1	Technology Research and Selection Methodology	15
3.2	System Architecture Design	15
3.3	Sensor Fusion Strategy	16
3.4	Atomic Movement Control Architecture	17
3.5	Real-time Performance Optimization	18
4	Implementation	19
4.1	Hardware Platform Migration and Integration	19
4.2	ROS2 Software Architecture Implementation	19
4.3	SLAM and Sensor Fusion Implementation	20
4.4	Human Detection and Pose Estimation Implementation	21
4.5	VR Integration and Atomic Movement System Implementation	21
4.6	System Integration and Testing Infrastructure	22
5	Evaluation	25
5.1	Localization System Performance Analysis	25
5.2	Human Detection and Pose Estimation Performance	26
5.3	VR Integration and User Experience Analysis	26
5.4	System Reliability and Performance Metrics	27
5.5	Comparative Analysis and Lessons Learned	28
6	Conclusions	29
6.1	Summary of Achievements	29
6.2	Technical Contributions and Innovation	30
6.3	Research Impact and Broader Implications	30
6.4	Limitations and Challenges	31
6.5	Future Work and Research Directions	32
6.6	Final Reflections and Conclusions	32
7	Temporal R&D	35

Bibliography	53
List of Figures	59
List of Tables	61
Acknowledgements	63

1 | Introduction

1.1. Social Robotics and Human-Robot Interaction

The field of social robotics has emerged from the fundamental human need for meaningful interaction and connection. As society increasingly integrates technology into daily life, the motivation for developing robots capable of effective social interaction becomes paramount. These robots must transcend mere functional utility to engage humans in ways that feel natural, empathetic, and purposeful.

Effective human-robot interaction encompasses several critical dimensions. Empathy represents the robot's ability to recognize, understand, and appropriately respond to human emotions and social cues. Trust emerges from consistent, predictable, and reliable robot behavior that aligns with human expectations and social norms. Accessibility ensures that robots can interact meaningfully with users of varying abilities, ages, and technological backgrounds.

The need for such effective interaction stems from robotics applications in healthcare, education, elderly care, and therapeutic settings, where the quality of human-robot relationships directly impacts outcomes. These applications demand robots that can navigate complex social dynamics while maintaining their functional objectives, often relying on non-verbal communication through movement, gesture, and spatial behavior.

1.2. The Tino Robot Project

The Tino robot project at Politecnico di Milano investigates novel approaches to mobile robot social interaction through an interdisciplinary approach combining robotics engineering, human-computer interaction, and artificial intelligence research. Within the AIRLab robotics laboratory, the project focuses on natural movement, responsive behavior, and immersive control paradigms that bridge the gap between virtual and physical interaction spaces.

Tino is part of a larger research initiative where it acts as a digital medium for interaction

between humans. In the envisioned scenario, one human interacts directly with the robot, while another controls the robot's movements remotely through a virtual reality interface. The two humans are unaware of each other's presence, with the robot serving as the communication medium. This setup enables the exploration of telepresence, empathy transfer, and mediated social interaction through robotic embodiment. The VR interface and virtual environment are being developed as part of a parallel thesis project, with this work focusing on the robot-side implementation that provides localization, orientation, and human pose data to the custom-built virtual space.

The robot's design philosophy emphasizes non-verbal and non-anthropomorphic features to build meaningful communication, convey emotions, and foster connections. By avoiding anthropomorphic design, Tino challenges conventional expectations of robotic form and demonstrates how purely physical movements can evoke empathy and emotional responses in human subjects. This approach enables exploration of movement as a communicative tool, independent of association with human anatomy.

The development of Tino V2 arose from the specific requirements of VR-based remote control and the limitations identified in the previous robot iteration. The legacy Tino system, while successful in demonstrating basic social interaction capabilities through direct local control, faced significant constraints when extended to real-time VR teleoperation. The original Raspberry Pi-based architecture with a Triskar omnidirectional base lacked the computational power necessary for real-time processing of VR commands, computer vision algorithms, and sophisticated sensor fusion required for remote operation.

The need for VR integration drove comprehensive system redesign focusing on enhanced computational capabilities, improved sensor fusion, and low-latency communication systems. The transition to more powerful hardware platforms enables real-time artificial intelligence processing while maintaining the responsive, expressive movement capabilities essential for effective mediated human interaction.

1.3. Project Objectives

1.3.1. Technical Objectives

The Tino V2 project establishes several key technical objectives that address the limitations of the previous system while enabling advanced VR-mediated interaction capabilities. The primary goal is the development of a computational platform capable of real-time processing for VR teleoperation, enabling responsive and low-latency control essential for natural human-robot interaction through virtual reality interfaces.

Enhanced localization and navigation capabilities represent another critical objective, addressing the need for precise robot positioning and spatial awareness to provide accurate localization and orientation data to the VR system. This includes the development of robust sensor fusion techniques that combine multiple sensing modalities to achieve reliable robot pose estimation for transmission to the custom VR environment developed by a parallel thesis project.

The integration of advanced perception systems for human detection enables the creation of virtual human representations within the VR environment. These capabilities support the transfer of real-time human pose information to the VR operator, creating virtual avatars that represent the humans in the robot's actual surroundings and enable informed interaction decisions within the custom-built virtual space.

System reliability and performance improvements focus on developing a robust platform capable of sustained operation during extended interaction sessions. This includes mechanical enhancements, improved sensor integration, and fail-safe behaviors that ensure consistent performance across diverse operational conditions.

1.3.2. Research Objectives

The research dimensions of the Tino V2 project contribute novel approaches to VR-mediated robotics and human-robot interaction. The investigation of immersive teleoperation paradigms explores how virtual reality interfaces can enable more natural and intuitive robot control, potentially improving the quality of mediated human interaction through robotic embodiment.

The development of adaptive movement systems designed specifically for VR control represents an innovative approach to robot teleoperation. This research investigates how complex robot behaviors can be decomposed into intuitive control primitives that feel natural when operated through virtual reality interfaces.

The study of real-time sensor fusion and environmental awareness in VR-controlled robots addresses the challenges of providing accurate robot localization and human pose data to the custom VR environment. This research explores techniques for reliable pose estimation and human detection that can be transmitted to the VR system developed by a parallel thesis project, enabling operators to make informed interaction decisions within the manually designed virtual space.

Advanced human-robot interaction paradigms emerge from the enhanced sensing and processing capabilities of the Tino V2 platform. The research investigates how VR-

mediated control can preserve and enhance the expressive movement capabilities that enable emotional communication and empathy formation between humans and robots.

1.4. Thesis Structure

This thesis is organized to provide a comprehensive understanding of the Tino V2 development process, from foundational research through implementation and evaluation:

- **Chapter 1: Introduction:** Presents the motivation for social robotics, the Tino project context, and research objectives.
- **Chapter 2: Background:** Covers the background research, technology survey, and detailed analysis of the legacy Tino system.
- **Chapter 3: Conceptual Work:** Details the overall system architecture, design decisions, and integration approach.
- **Chapter 4: Implementation:** Focuses on the technical implementation including hardware redesign, sensor fusion, human detection systems, and VR integration.
- **Chapter 5: Evaluation:** Provides evaluation results, testing procedures, and performance analysis.
- **Chapter 6: Conclusions:** Summarizes findings, contributions, and future research directions.

2 | Background

2.1. Simultaneous Localization and Mapping (SLAM) Technologies

Simultaneous Localization and Mapping (SLAM) represents one of the fundamental challenges in autonomous mobile robotics, particularly for social robots operating in dynamic indoor environments [15]. SLAM systems enable robots to navigate in unknown environments while building environmental maps, providing the dual advantage of localization and environmental perception [33]. This section provides a comprehensive analysis of available SLAM and localization technologies, examining their theoretical foundations, practical implementations, and suitability for social robotics applications.

2.1.1. Feature-Based SLAM Systems

Monocular visual SLAM systems, exemplified by PTAM [23] and later ORB-SLAM [28], utilize feature extraction and matching to estimate camera motion and reconstruct environmental structure. While computationally efficient, these systems suffer from inherent scale ambiguity and require careful initialization procedures to establish metric scale.

ORB-SLAM3 [11] represents the current state-of-the-art in feature-based SLAM systems, supporting multiple sensor configurations including monocular, stereo, and RGB-D cameras. The system demonstrates robust performance in feature-rich environments through sophisticated ORB feature matching algorithms and advanced loop closure detection mechanisms. The system offers several advantages including proven accuracy in academic benchmarks such as TUM RGB-D and EuRoC datasets, support for multiple sensor modalities, and robust handling of dynamic environments.

However, practical implementation reveals significant challenges including substantial computational requirements that typically necessitate GPU optimization for real-time performance [22]. The system can struggle in textureless environments or under poor lighting conditions where ORB feature extraction becomes unreliable. Compilation and

integration challenges, particularly on ARM64 architectures, can also present development obstacles.

Stereo visual odometry addresses the scale ambiguity problem by utilizing depth information from calibrated stereo camera pairs. These approaches provide accurate trajectory estimation through triangulation-based depth reconstruction, though they face significant challenges in textureless environments or under poor lighting conditions where feature matching becomes unreliable [19].

2.1.2. Direct and Semi-Direct SLAM Systems

Semi-direct Visual Odometry (SVO) [18] offers reduced computational requirements compared to feature-based SLAM systems by tracking pixels directly rather than extracting and matching discrete features. The system demonstrates particular compatibility with fisheye and catadioptric cameras, making it suitable for wide field-of-view applications.

Direct Sparse Odometry (DSO) [17] represents another approach in this category, utilizing direct photometric error minimization without feature extraction. While these systems offer computational advantages, they require well-textured environments for reliable tracking and can struggle in highly dynamic scenes. SVO presents compilation challenges on ARM64 architectures and lacks comprehensive map management capabilities.

2.1.3. Graph-Based SLAM Systems

RTABMap (Real-Time Appearance-Based Mapping) [24] employs appearance-based mapping techniques combined with graph optimization to provide robust performance across various environmental conditions. The system offers comprehensive map management capabilities including reliable save and load functionality, multi-session mapping support, and effective relocalization performance.

RTABMap demonstrates particular strength in practical deployment scenarios, offering stable map persistence and reliable relocalization capabilities essential for operational robot systems. The system provides excellent integration with standard robotics frameworks including ROS and ROS2, supporting various sensor configurations including RGB-D cameras and stereo systems.

The integration of RGB-D sensors enables more reliable feature tracking and geometric reconstruction compared to traditional stereo approaches. However, these systems face limitations including reduced operating range (typically under 5 meters) and sensitivity

to lighting conditions that affect depth sensor performance [21]. The primary advantages include robust map management, reliable relocalization, excellent framework integration, and proven performance in real-world deployments, making RTABMap particularly suitable for applications requiring consistent long-term operation and map reuse capabilities.

2.1.4. Ultra-Wideband Positioning Technology

Ultra-Wideband (UWB) technology has gained significant attention for indoor positioning applications due to its potential for centimeter-level accuracy [20]. The technology operates in the 3.1–10.6 GHz frequency range, providing high temporal resolution that enables precise time-of-flight distance measurements.

UWB positioning systems require infrastructure deployment including multiple anchor points with known positions to enable triangulation-based localization. Research has demonstrated the technology’s advantages including low power consumption, minimal interference with other wireless systems, and potential penetration through materials including fabrics [2].

However, UWB systems face significant challenges in Non-Line-of-Sight (NLOS) conditions where multipath effects can degrade positioning accuracy [26]. The technology primarily provides position information and requires additional sensors for orientation estimation, typically through fusion with inertial measurement units.

2.1.5. LiDAR-Based SLAM

LiDAR systems provide high-resolution 3D environmental mapping through laser scanning, offering excellent spatial resolution and range performance [38]. These systems enable accurate simultaneous localization and mapping through point cloud analysis and clustering algorithms, providing robust environmental perception capabilities.

LiDAR-based SLAM offers several advantages including operation in various lighting conditions, precise distance measurements, and detailed environmental reconstruction. The technology is particularly effective for large-scale outdoor environments and can provide reliable localization even in feature-poor environments where visual SLAM systems might struggle.

However, LiDAR systems present several challenges for social robotics applications. The mechanical scanning components can be sensitive to vibration, making them unsuitable for robots with soft or flexible structures. The cost and size of LiDAR systems can also be prohibitive for many social robot applications. Additionally, the range and resolution

capabilities of LiDAR systems may represent overkill for typical indoor social interaction scenarios.

2.1.6. Sensor Fusion Approaches

Modern localization systems increasingly rely on sensor fusion techniques to combine information from multiple sensing modalities. Extended Kalman Filter (EKF) and particle filter approaches enable fusion of visual odometry, IMU data, and wheel encoder information to achieve robust localization performance [22].

Inertial Measurement Units (IMUs) combined with wheel encoders represent a low-cost localization approach suitable for many mobile robot applications. While these systems offer advantages including low cost and independence from environmental conditions, they suffer from drift accumulation over time due to integration of noisy sensor measurements [9].

Performance can be particularly poor on uneven surfaces where wheel slip affects encoder accuracy, or in applications involving impulse-based movement patterns that can saturate IMU sensors. The drift characteristics make these approaches unsuitable as standalone solutions for long-term autonomous operation.

2.2. Human Detection and Pose Estimation Technologies

Human detection and tracking capabilities are essential for social robots that must operate safely and effectively in human-populated environments. This section examines various sensing modalities and computational approaches for human perception in robotics applications.

2.2.1. Computer Vision-Based Human Detection

RGB-D Sensing Systems

RGB-D cameras provide simultaneous color and depth information, enabling robust human detection and pose estimation. Intel RealSense cameras and similar structured light devices offer synchronized data streams that support skeleton tracking through established frameworks.

OpenPose [12] represents a breakthrough in real-time multi-person pose estimation,

utilizing Part Affinity Fields (PAFs) to associate body parts with individuals in crowded scenes. The system provides detailed information about human posture and gesture, enabling rich interaction capabilities for social robotics applications.

MediaPipe [25] offers an alternative framework optimized for mobile and embedded platforms, providing real-time pose estimation with reduced computational requirements. These systems demonstrate comprehensive human pose information extraction with reasonable computational demands.

Integration challenges arise when cameras must be concealed within robot structures, particularly for robots with soft exteriors where camera visibility may be constrained.

Machine Learning-Enhanced Detection

Modern deep learning approaches have revolutionized human detection capabilities. YOLO (You Only Look Once) architectures, particularly YOLOv8 [32] and recent variants including YOLOv11, provide real-time human detection and pose estimation capabilities with single-stage detection frameworks.

These systems offer several advantages including lower hardware costs, reduced physical profile compared to RGB-D cameras, and sophisticated pose estimation capabilities. Monocular depth estimation networks such as MiDaS [31] can provide approximate depth information without requiring specialized depth sensors.

The primary challenges include computational requirements for real-time processing on embedded platforms and dependency on lighting conditions. However, advances in embedded AI processing platforms, including NVIDIA Jetson series and specialized AI accelerators, are making these approaches increasingly viable for mobile robotics applications.

2.2.2. Alternative Sensing Modalities

Thermal Imaging

Thermal cameras detect infrared radiation emitted by objects, making them particularly effective for human detection regardless of lighting conditions [3]. Thermal imaging offers potential advantages including operation in complete darkness and possible penetration through certain materials including fabrics.

The primary benefit for social robotics applications is the ability to detect human presence even when optical cameras may be obscured. However, thermal sensors provide limited contextual information beyond heat signatures, making it difficult to extract de-

tailed pose information or distinguish between different individuals.

LiDAR-Based Detection

LiDAR systems provide high-resolution 3D environmental mapping through laser scanning, offering excellent spatial resolution and range performance [38]. These systems can accurately detect human presence and track movement through point cloud analysis and clustering algorithms.

However, LiDAR systems present several challenges for social robotics applications. The mechanical scanning components can be sensitive to vibration, making them unsuitable for robots with soft or flexible structures. The cost and size of LiDAR systems can also be prohibitive for many social robot applications.

2.3. Social Robotics and Human-Robot Interaction Foundations

Social robotics has emerged as a distinct field focusing on robots designed to interact with humans in natural, socially meaningful ways [10]. Understanding the theoretical foundations of human-robot interaction provides essential context for the technological requirements of social robots.

2.3.1. Non-Verbal Communication in Robotics

Research in human communication demonstrates that non-verbal cues carry significant emotional weight, often conveying information that verbal communication cannot express [27]. Ekman's seminal work on universal facial expressions shows that fundamental emotions are communicated across cultures through body language and facial expressions [16].

These insights have profound implications for social robotics design. Robots capable of expressing emotions through coordinated physical movements can potentially achieve meaningful communication without relying on anthropomorphic features or verbal interaction [10]. This approach enables exploration of movement as a primary communicative tool, independent of human anatomical associations.

2.3.2. Telepresence and Mediated Interaction

Telepresence robotics research investigates how humans can effectively control remote robotic systems to achieve natural interaction with distant environments [34]. The field has evolved from simple teleoperation to sophisticated systems that preserve social presence and emotional connection across physical distances.

Recent advances in virtual reality technology have opened new possibilities for immersive robot control, where operators can experience robot embodiment through first-person perspectives. This approach enables investigation of how virtual reality interfaces can enhance rather than diminish the emotional and empathetic qualities of robot-mediated human interaction.

2.4. VR Integration in Robotics

Virtual Reality integration with robotics systems represents an emerging area of research with significant potential for advancing human-robot interaction capabilities. This section examines the current state of VR-robotics integration and its implications for social robotics applications.

2.4.1. Immersive Teleoperation Systems

Traditional robot teleoperation relies on external displays and control interfaces that create cognitive distance between the operator and the robot [34]. Immersive VR interfaces offer the potential to bridge this gap by providing first-person robot perspectives and natural control paradigms.

Research in VR teleoperation has demonstrated improved operator performance and reduced cognitive load when controlling robots through immersive interfaces. These systems enable operators to leverage natural spatial reasoning and motor skills for robot control, potentially improving the quality and naturalness of robot movements.

2.4.2. Challenges in VR-Robot Integration

Successful VR-robot integration faces several technical challenges including latency requirements, sensor data processing, and spatial correspondence between virtual and physical environments. Real-time performance requirements typically mandate processing latencies below 100 milliseconds to maintain immersive experiences and prevent motion sickness in VR applications.

The integration requires sophisticated data processing pipelines to convert robot sensor data into meaningful VR representations while maintaining real-time performance constraints. Spatial registration between virtual environments and physical robot spaces presents additional challenges, particularly when virtual environments are custom-designed rather than directly mapped from physical spaces.

2.5. Legacy Tino System Analysis

Understanding the limitations and capabilities of the original Tino robot system provides essential context for the improvements implemented in Tino V2. This comprehensive analysis examines the legacy system architecture and identifies specific areas requiring enhancement.

2.5.1. Original System Architecture

The original Tino robot, developed as part of previous research in social robotics [13], utilized a Raspberry Pi-based control architecture with inherent computational limitations. The system employed a Triskar omnidirectional base providing three-degree-of-freedom mobility through three independently controlled wheels arranged in a triangular configuration.

While this omnidirectional configuration offered excellent theoretical maneuverability for social interaction scenarios, practical deployment revealed significant reliability issues under the robot's operational weight. Wheel degradation and mechanical wear patterns limited system reliability and required frequent maintenance interventions.

The legacy sensor suite was extremely limited, consisting only of a basic Pi camera used exclusively for video streaming to remote operators. The system lacked any environmental sensors or perception capabilities, with the camera providing raw video feed without any on-board processing for computer vision, object detection, or environmental analysis.

2.5.2. Software Architecture Limitations

The original software architecture employed monolithic Python scripts with limited modularity and debugging capabilities. This approach made system maintenance and feature development challenging, particularly when attempting to integrate new sensing modalities or advanced control paradigms.

Communication between system components was handled through simple serial inter-

faces without the robust messaging frameworks required for complex multi-sensor systems. The absence of standardized robotics frameworks made it difficult to leverage existing libraries and tools for advanced robotics capabilities.

2.5.3. VR Integration Requirements

The motivation for Tino V2 development arose primarily from requirements for VR integration that were not feasible within the original system constraints. VR teleoperation demands real-time processing of commands, sophisticated sensor fusion for accurate localization, and low-latency communication protocols.

The legacy system lacked the computational capabilities required for real-time computer vision processing, advanced SLAM algorithms, or sophisticated human detection and pose estimation systems. These limitations prevented implementation of the data processing pipelines necessary for meaningful VR integration.

Most critically, the original architecture could not support the real-time sensor fusion and environmental mapping capabilities required to provide accurate robot localization and human pose data to VR systems. Without these capabilities, meaningful VR-mediated social interaction remained impossible.

2.5.4. Identified Enhancement Requirements

Analysis of the legacy system revealed several critical enhancement requirements that motivated the comprehensive redesign undertaken in Tino V2:

- **Computational Platform Upgrade:** Migration from Raspberry Pi to more powerful embedded computing platforms capable of real-time AI processing and sophisticated sensor fusion.
- **Advanced Localization Systems:** Implementation of robust SLAM capabilities with map persistence and reliable relocalization for consistent spatial awareness.
- **Human Detection and Pose Estimation:** Integration of real-time computer vision systems capable of detecting and tracking human pose for VR representation.
- **Modular Software Architecture:** Development of ROS-based modular architecture enabling easier integration of new capabilities and robust debugging.
- **VR Communication Infrastructure:** Implementation of low-latency communication systems capable of real-time data exchange with VR environments.

- **Mechanical Reliability Improvements:** Redesign of the mobility platform to address reliability issues while maintaining expressive movement capabilities.

These requirements collectively defined the scope and objectives for the Tino V2 development project, establishing the foundation for the technological solutions presented in subsequent chapters.

3 | Conceptual Work

3.1. Technology Research and Selection Methodology

This section will detail the systematic approach undertaken to research and select appropriate technologies for the Tino V2 system redesign. The research methodology will be established, covering the comprehensive literature review process used to identify state-of-the-art solutions in SLAM, sensor fusion, human detection, and VR integration, the evaluation criteria developed to assess technologies based on accuracy, computational requirements, integration complexity, and long-term maintenance considerations, and the experimental validation approach used to test promising technologies before final selection. The localization technology analysis will be presented, covering the systematic comparison of visual odometry approaches including ORB-SLAM3, SVO, and RTABMap with detailed evaluation of their computational requirements and accuracy performance, the assessment of UWB positioning systems including anchor configuration requirements and accuracy characteristics under various environmental conditions, and the analysis of sensor fusion approaches that combine complementary positioning technologies to address individual limitations. The human detection technology evaluation will be detailed, covering the comparison of different YOLO variants and their performance on embedded hardware, the assessment of depth camera technologies including Intel RealSense and Oak-D Pro systems, and the analysis of optimization frameworks like TensorRT for real-time inference on NVIDIA platforms. The VR integration technology selection will be examined, covering the evaluation of Unity versus other VR development platforms, the assessment of communication frameworks for ROS2-VR integration, and the analysis of VR hardware compatibility and user experience considerations.

3.2. System Architecture Design

This section will present the comprehensive system architecture developed for the Tino V2 platform, covering both hardware and software components and their integration

strategies. The overall architecture philosophy will be established, covering the modular design approach that enables independent development and testing of subsystems, the ROS2-based communication framework that provides standardized interfaces between components, and the scalability considerations that allow for future expansion and modification of system capabilities. The hardware architecture will be detailed, covering the migration from Raspberry Pi to NVIDIA Orin Nano as the primary computing platform with analysis of performance improvements and power consumption considerations, the integration of Oak-D Pro depth camera for simultaneous SLAM and human detection capabilities, the UWB positioning system implementation including anchor placement strategies and coordinate system alignment, and the redesigned mechanical systems including the differential drive base and improved Stewart platform head mechanism. The software architecture will be presented, covering the ROS2 node structure that organizes functionality into discrete, communicating processes, the message passing interfaces that enable real-time data exchange between localization, detection, control, and VR systems, the data recording and analysis frameworks that support system debugging and performance evaluation, and the launch file organization that enables easy switching between mapping and localization modes. The integration strategy will be examined, covering the synchronization mechanisms that ensure consistent timing across all system components, the error handling and recovery procedures that maintain system stability during component failures, and the calibration procedures required to achieve accurate sensor fusion and coordinate system alignment.

3.3. Sensor Fusion Strategy

This section will detail the conceptual approach developed for fusing UWB positioning data with RTABMap visual SLAM to achieve robust indoor localization. The sensor fusion rationale will be established, covering the complementary nature of UWB and visual SLAM technologies where UWB provides absolute positioning without drift while visual SLAM provides accurate orientation and short-term relative positioning, the challenges addressed by sensor fusion including UWB signal degradation in non-line-of-sight conditions and visual SLAM drift accumulation over extended operation, and the advantages gained through fusion including improved localization reliability and reduced dependence on environmental conditions. The fusion architecture will be presented, covering the decision to use UWB for global position estimation while relying on RTABMap for orientation data, the coordinate system transformations required to align UWB and camera coordinate frames, and the filtering and smoothing strategies employed to handle measurement noise and temporary signal loss. The implementation strategy will be detailed, covering

the ROS2 message structures used to combine positioning data from multiple sources, the timing synchronization mechanisms that ensure consistent sensor fusion despite different update rates, and the parameter tuning approaches used to optimize fusion performance for the specific operating environment. The validation methodology will be outlined, covering the experimental protocols developed to assess fusion accuracy compared to individual sensor performance, the test scenarios designed to evaluate system behavior under challenging conditions like UWB signal obstruction or visual feature scarcity, and the metrics developed to quantify localization accuracy and reliability improvements achieved through sensor fusion.

3.4. Atomic Movement Control Architecture

This section will present the conceptual framework developed for the atomic movement control system that enables predictable VR interaction with the physical robot. The atomic movement concept will be established, covering the principle that all robot movements must be discrete, completion-guaranteed operations that provide clear feedback to VR users, the four-state control architecture that separates robot behaviors into rest, attention-getting, preparation, and execution phases, and the synchronization requirements that ensure coordinated behavior between base locomotion and leg articulation systems. The VR interaction design will be detailed, covering the mapping of VR user actions to appropriate robot movement commands, the feedback mechanisms that inform users about robot state and movement completion, and the safety considerations that prevent conflicting commands and ensure predictable robot behavior. The movement coordination strategy will be presented, covering the locking mechanisms that prevent simultaneous conflicting movements, the command queuing system that handles user inputs during ongoing movements, and the timing optimization that ensures natural-looking robot behavior while maintaining system responsiveness. The state machine architecture will be examined, covering the discrete states defined for leg and base controllers, the transition conditions that govern movement between states, and the error handling procedures that maintain system stability when movements cannot be completed as planned. The pulse command system will be detailed, covering the implementation of discrete command signals that automatically return to idle state, the timing parameters that ensure complete movement execution, and the integration with the VR system that provides appropriate user feedback for each movement phase.

3.5. Real-time Performance Optimization

This section will address the conceptual approaches developed to achieve real-time performance across all system components despite the computational demands of simultaneous SLAM, human detection, and VR communication. The performance requirements analysis will be established, covering the timing constraints imposed by VR interaction that require low-latency response to user inputs, the computational load distribution across SLAM processing, pose detection, and system coordination tasks, and the memory management strategies required for sustained operation on embedded hardware with limited resources. The optimization strategy will be presented, covering the parallel processing architecture that distributes computational load across multiple ROS2 nodes, the priority scheduling that ensures critical functions like localization maintain consistent performance, and the resource allocation strategies that balance accuracy with computational efficiency. The real-time processing pipeline will be detailed, covering the image processing optimization that enables simultaneous SLAM and human detection from the same camera stream, the communication optimization that minimizes latency in VR data exchange, and the memory management that prevents performance degradation during extended operation. The scalability considerations will be examined, covering the modular architecture that allows selective activation of system components based on current requirements, the configuration management that enables performance tuning for different operational scenarios, and the monitoring systems that provide real-time feedback about system performance and resource utilization to support ongoing optimization efforts.

4 | Implementation

4.1. Hardware Platform Migration and Integration

This section will detail the complete hardware migration from the legacy Raspberry Pi system to the modern NVIDIA Orin Nano platform and the integration of new sensing technologies. The computing platform upgrade will be presented first, covering the migration process from Raspberry Pi 4 to NVIDIA Orin Nano including performance benchmarking that demonstrated substantial improvements in processing capability, the power system redesign using DC-DC converters to provide stable 19V power for the Orin Nano from 12V battery systems, and the cooling and thermal management solutions implemented to ensure reliable operation during intensive computational tasks. The camera system integration will be detailed, covering the replacement of the basic Pi camera with the Oak-D Pro depth camera system, the mechanical mounting solutions developed including the tripod-based camera support system and protective housing that shields the camera while maintaining visibility, and the camera calibration procedures required for accurate depth estimation and SLAM performance. The UWB positioning system implementation will be examined, covering the anchor placement strategy developed for the laboratory environment, the coordinate system calibration procedures that align UWB global coordinates with the robot's local reference frame, and the integration of UWB receivers with the Orin Nano through USB interfaces and custom ROS2 drivers. The mechanical system upgrades will be addressed, covering the complete redesign of the base locomotion system from omnidirectional trike wheels to differential drive configuration, the motor and driver upgrades that provide increased power and reliability for the heavier Orin Nano system, and the Stewart platform head improvements including new arm designs with heim joints that eliminate flex and improve reliability.

4.2. ROS2 Software Architecture Implementation

This section will present the complete implementation of the ROS2-based software architecture that replaced the legacy monolithic Python scripts with a modular, main-

tainable system. The node architecture design will be established first, covering the development of specialized ROS2 nodes including `gamepad_node.py` for Xbox controller input processing, `hardware_interface_node.py` for Arduino communication management, `robot_controller_node.py` for central coordination and behavior management, and `vr_interface_node.py` for Unity integration and data exchange. The communication framework implementation will be detailed, covering the ROS2 topic structure that enables real-time data sharing between nodes, the service interfaces used for system configuration and control commands, the parameter management system that allows runtime configuration of system behavior, and the launch file organization that enables easy switching between mapping and localization operational modes. The Arduino integration will be examined, covering the serial communication protocols developed for reliable data exchange with head, base, and leg control systems, the device symlink configuration that ensures consistent device addressing across system restarts, and the command processing logic that translates high-level movement commands into appropriate motor control signals. The data recording and analysis infrastructure will be presented, covering the implementation of comprehensive logging systems that capture all sensor data, robot states, and user interactions for later analysis, the data extraction tools developed for processing recorded sessions, and the visualization systems that enable real-time monitoring of system performance and debugging of operational issues.

4.3. SLAM and Sensor Fusion Implementation

This section will detail the implementation of the RTABMap SLAM system and its integration with UWB positioning for robust indoor localization. The RTABMap configuration and optimization will be presented first, covering the parameter tuning process that optimized performance for the Oak-D Pro camera system, the memory management configuration that enables long-term mapping without excessive resource consumption, and the loop closure detection settings that ensure reliable map consistency during extended operation. The UWB integration implementation will be detailed, covering the development of custom ROS2 nodes for UWB data processing and coordinate transformation, the sensor fusion logic that combines UWB position data with RTABMap orientation information, and the coordinate system alignment procedures that ensure consistent mapping between UWB global coordinates and robot local coordinates. The mapping and localization modes will be examined, covering the implementation of distinct operational modes for initial map creation and subsequent localization within existing maps, the map saving and loading procedures that enable persistent environment representations, and the automatic relocalization capabilities that allow the robot to recover its position after

temporary tracking loss. The performance optimization will be addressed, covering the computational optimization that enables real-time SLAM processing on the Orin Nano platform, the memory management strategies that prevent performance degradation during extended mapping sessions, and the error handling procedures that maintain system stability when sensor data is temporarily unavailable or degraded.

4.4. Human Detection and Pose Estimation Implementation

This section will present the implementation of real-time human detection and pose estimation using YOLOv11 with TensorRT optimization. The YOLOv11 model preparation and optimization will be detailed first, covering the conversion process from PyTorch models to TensorRT optimized engines for the Orin Nano platform, the model quantization and optimization strategies that balance detection accuracy with inference speed, and the integration with the Oak-D Pro camera system that enables simultaneous RGB and depth data processing for 3D pose estimation. The real-time processing pipeline will be examined, covering the image preprocessing and postprocessing steps that prepare camera data for inference and extract meaningful pose information, the coordinate transformation calculations that convert 2D detection results to 3D world coordinates using depth information, and the tracking algorithms that maintain consistent person identification across multiple frames. The pose estimation implementation will be detailed, covering the 17-keypoint skeleton detection that provides comprehensive human pose information, the depth integration that enables accurate 3D position calculation for each detected person, and the filtering and smoothing algorithms that reduce noise and provide stable pose tracking despite occasional detection errors. The integration with robot systems will be addressed, covering the ROS2 message structures used to publish human detection and pose data for use by other system components, the coordinate system transformations that align human pose data with robot and world coordinate frames, and the performance monitoring systems that track detection accuracy and computational load to ensure real-time operation requirements are maintained.

4.5. VR Integration and Atomic Movement System Implementation

This section will detail the implementation of the complete VR integration system including Unity-based user interfaces and atomic movement control architecture. The Unity

VR application development will be presented first, covering the creation of immersive VR environments that provide intuitive robot control interfaces, the implementation of bidirectional communication systems that enable real-time data exchange between Unity and ROS2, and the user interface design that makes robot control accessible to users without technical expertise. The ROS-TCP-Endpoint integration will be examined, covering the implementation of reliable communication bridges between Unity and the ROS2 ecosystem, the message serialization and deserialization procedures that handle complex data structures, and the error handling and reconnection logic that maintains communication stability despite network interruptions. The atomic movement system implementation will be detailed, covering the four-state control architecture implemented in both leg and base controllers, the synchronization mechanisms that ensure coordinated movement between robot subsystems, and the command queuing and locking systems that prevent conflicting movement commands. The audio system integration will be addressed, covering the implementation of bidirectional audio communication that enables natural conversation between VR users and people near the robot, the audio processing pipelines that handle microphone input and speaker output, and the integration with the VR system that provides seamless audio transmission. The data recording and analysis implementation will be presented, covering the comprehensive logging systems that capture all VR interactions, robot responses, and sensor data for detailed analysis of human-robot interaction patterns, the data extraction and visualization tools that enable researchers to analyze user behavior and system performance, and the real-time monitoring systems that provide immediate feedback about system operation and performance metrics during VR sessions.

4.6. System Integration and Testing Infrastructure

This section will present the implementation of comprehensive system integration procedures and testing infrastructure that ensure reliable operation of all system components. The integration testing methodology will be established, covering the systematic approach used to validate individual subsystem functionality before full system integration, the interface testing procedures that verify reliable communication between ROS2 nodes, and the end-to-end testing protocols that validate complete system operation from VR input to robot response. The calibration and configuration management will be detailed, covering the sensor calibration procedures required for accurate camera, UWB, and coordinate system alignment, the parameter management systems that store and maintain optimal configuration settings for different operational scenarios, and the automated configuration validation tools that detect and report configuration errors or drift. The debugging and monitoring infrastructure will be examined, covering the comprehensive logging systems

that capture detailed system state information for troubleshooting and optimization, the real-time monitoring displays that provide immediate feedback about system performance and health, and the automated error detection and reporting systems that identify and classify system issues to support rapid problem resolution. The deployment and maintenance procedures will be addressed, covering the automated setup scripts that streamline system installation and configuration, the backup and recovery procedures that protect against data loss and enable rapid system restoration, and the update and maintenance protocols that enable safe system modification and capability enhancement without disrupting ongoing research activities.

5 | Evaluation

5.1. Localization System Performance Analysis

This section will present a comprehensive analysis of the localization system performance, comparing the sensor fusion approach against baseline SLAM-only operation to demonstrate the improvements achieved through UWB integration. The experimental methodology will be established first, covering the controlled testing environment setup using fixed reference points marked on the laboratory floor, the systematic data collection protocols that captured position estimates from both SLAM-only and UWB-fused systems during identical movement patterns, and the statistical analysis methods used to quantify localization accuracy and consistency. The SLAM-only baseline performance will be analyzed, covering the drift characteristics observed during extended operation where position estimates deviated by up to 1.2 meters from true positions at fixed reference points, the orientation accuracy that remained reliable even when position estimates suffered significant drift, and the specific scenarios that triggered substantial localization errors including proximity to walls and areas with limited visual features. The sensor fusion performance improvements will be detailed, covering the substantial reduction in position error achieved through UWB integration where maximum deviations were reduced to centimeter-level accuracy, the improved consistency of position estimates across multiple test runs at identical reference points, and the enhanced reliability of localization during challenging scenarios that previously caused significant SLAM drift. The coordinate system alignment results will be presented, covering the successful integration of UWB global coordinates with RTABMap local coordinate systems, the accuracy of coordinate transformations validated through systematic position measurements at known reference points, and the stability of alignment parameters that maintained consistent coordinate mapping throughout extended testing sessions.

5.2. Human Detection and Pose Estimation Performance

This section will analyze the performance of the YOLOv11-based human detection and pose estimation system, focusing on accuracy, processing speed, and reliability under various operational conditions. The detection accuracy analysis will be presented first, covering the successful implementation of 17-keypoint human pose detection that provides comprehensive skeletal tracking information, the depth integration performance that enables accurate 3D position estimation for detected humans using Oak-D Pro stereo camera data, and the tracking consistency that maintains stable person identification across consecutive frames despite variations in pose and position. The real-time performance evaluation will be detailed, covering the achievement of real-time inference speeds through TensorRT optimization that enables simultaneous SLAM and human detection processing, the computational resource utilization analysis that demonstrates efficient operation within the constraints of the Orin Nano platform, and the system responsiveness that provides immediate detection results for integration with robot control systems. The robustness testing results will be examined, covering the system performance under varying lighting conditions and camera perspectives that demonstrated reliable detection across realistic operational scenarios, the handling of multiple people in the camera field of view with appropriate tracking and identification capabilities, and the error handling performance that maintains system stability when detection temporarily fails or produces uncertain results. The integration effectiveness will be addressed, covering the successful coordination between human detection and robot behavior systems that enables responsive human-robot interaction, the coordinate system accuracy that properly aligns detected human poses with robot and world coordinate frames, and the data quality that provides meaningful input for research analysis and VR system integration.

5.3. VR Integration and User Experience Analysis

This section will evaluate the effectiveness of the VR integration system and the user experience provided by the atomic movement control architecture. The VR communication performance will be analyzed first, covering the low-latency data exchange achieved between Unity and ROS2 systems that enables responsive robot control from VR environments, the bidirectional audio communication quality that supports natural conversation between VR users and people near the robot, and the system reliability that maintains stable communication despite network variations and computational load changes. The

atomic movement system evaluation will be detailed, covering the successful implementation of the four-state control architecture that provides predictable and intuitive robot behavior for VR users, the synchronization accuracy between leg and base movements that creates natural-looking robot locomotion, and the command queuing effectiveness that handles user inputs during ongoing movements without losing commands or creating conflicts. The user interface effectiveness will be examined, covering the intuitive VR control interfaces that enable effective robot operation without requiring technical expertise, the feedback systems that provide users with clear information about robot state and movement progress, and the safety mechanisms that prevent dangerous or conflicting robot behaviors during remote operation. The overall user experience will be assessed, covering the natural interaction feel achieved through careful timing and coordination of robot movements, the immersion quality that makes users feel genuinely present in the robot's environment, and the accessibility features that accommodate users with varying levels of VR experience and technical background.

5.4. System Reliability and Performance Metrics

This section will present comprehensive analysis of overall system reliability, computational performance, and operational characteristics under extended use conditions. The system stability analysis will be established first, covering the uptime performance achieved during extended testing sessions with the ROS2 architecture demonstrating robust operation without memory leaks or performance degradation, the error recovery capabilities that maintain system functionality when individual components experience temporary failures, and the graceful degradation behavior that preserves core functionality when non-critical subsystems encounter problems. The computational performance evaluation will be detailed, covering the resource utilization analysis that demonstrates efficient operation within the constraints of the Orin Nano platform while simultaneously running SLAM, human detection, and VR communication systems, the real-time performance maintenance that preserves responsive operation despite varying computational loads, and the power consumption characteristics that enable practical battery-powered operation for meaningful durations. The integration robustness will be examined, covering the communication reliability between ROS2 nodes that maintains consistent data flow despite network variations and system load changes, the sensor fusion stability that provides consistent localization performance across different environmental conditions and operational scenarios, and the modular architecture effectiveness that enables independent debugging and optimization of individual subsystems. The operational flexibility will be addressed, covering the ease of switching between mapping and localization modes

that enables efficient system deployment in new environments, the configuration management effectiveness that allows system optimization for different operational requirements, and the maintenance and troubleshooting capabilities that support ongoing research and development activities with minimal downtime and technical expertise requirements.

5.5. Comparative Analysis and Lessons Learned

This section will synthesize the evaluation results to provide comparative analysis against baseline systems and extract key lessons learned from the implementation and testing process. The performance improvement quantification will be presented first, covering the measurable improvements achieved through the migration from Raspberry Pi to Orin Nano architecture including processing speed increases and capability expansion, the localization accuracy improvements demonstrated through UWB sensor fusion compared to SLAM-only operation, and the system reliability enhancements achieved through modular ROS2 architecture compared to legacy monolithic implementations. The technology validation results will be detailed, covering the successful validation of RTABMap SLAM for indoor social robot applications despite challenges with visual feature scarcity and dynamic environments, the effectiveness of UWB positioning for absolute coordinate reference despite occasional signal degradation and non-line-of-sight conditions, and the practical feasibility of real-time human pose detection on embedded platforms through careful optimization and model selection. The implementation insights will be examined, covering the critical importance of systematic testing and calibration procedures for achieving reliable sensor fusion performance, the value of modular software architecture for enabling efficient debugging and system optimization, and the significance of user-centered design principles for creating effective VR interaction systems. The research contribution significance will be assessed, covering the advancement of practical sensor fusion approaches that address common indoor robotics challenges, the development of atomic movement architectures that enable predictable VR-robot interaction, and the demonstration of comprehensive integration strategies that combine multiple advanced technologies in a cohesive, reliable robotic system suitable for social robotics research and applications.

6 | Conclusions

6.1. Summary of Achievements

This section will synthesize the major achievements accomplished through the Tino V2 development project, highlighting the successful transformation from a legacy system to a modern, capable social robotics platform. The technological transformation will be summarized first, covering the complete migration from Raspberry Pi-based architecture to NVIDIA Orin Nano systems that provided substantial improvements in computational capability and real-time performance, the successful integration of advanced sensing technologies including Oak-D Pro depth cameras and UWB positioning systems that enhanced the robot's perception and localization capabilities, and the implementation of modern ROS2-based software architecture that replaced monolithic legacy code with modular, maintainable systems. The sensor fusion success will be highlighted, covering the development and validation of UWB-SLAM fusion that addressed critical indoor localization drift issues while maintaining real-time performance requirements, the achievement of centimeter-level positioning accuracy that enables reliable VR integration and precise navigation, and the demonstration of robust localization performance across various environmental conditions and operational scenarios. The VR integration accomplishments will be detailed, covering the successful implementation of bidirectional communication systems that enable natural remote interaction through Unity-based VR environments, the development of atomic movement architectures that provide predictable and intuitive robot control for non-technical users, and the creation of comprehensive data recording and analysis systems that support ongoing research in human-robot interaction. The human detection achievements will be presented, covering the successful implementation of real-time pose estimation using optimized YOLOv11 models that provide 17-keypoint skeletal tracking with depth information, the achievement of simultaneous SLAM and human detection processing on embedded hardware, and the integration of detection results with robot control systems for responsive human-robot interaction.

6.2. Technical Contributions and Innovation

This section will detail the specific technical contributions made to the field of social robotics and the innovative approaches developed during this thesis work. The sensor fusion methodology contribution will be established first, covering the practical demonstration of UWB-visual SLAM integration that addresses common indoor robotics localization challenges, the development of coordinate system alignment procedures that enable reliable fusion of absolute and relative positioning data, and the validation of sensor fusion approaches that maintain real-time performance while improving localization accuracy and reliability. The atomic movement architecture contribution will be detailed, covering the development of discrete state-based control systems that enable predictable VR interaction with physical robots, the implementation of synchronization mechanisms that coordinate multiple robot subsystems for natural movement patterns, and the creation of command queuing and conflict resolution systems that handle complex user interactions safely and effectively. The real-time optimization contributions will be examined, covering the successful implementation of simultaneous SLAM, human detection, and VR communication on embedded hardware through careful architectural design and optimization strategies, the development of parallel processing approaches that distribute computational load effectively across multiple ROS2 nodes, and the achievement of low-latency communication between VR and robotics systems that enables responsive remote interaction. The integration methodology contributions will be addressed, covering the development of comprehensive testing and validation procedures that ensure reliable multi-component system operation, the creation of modular software architectures that enable independent development and debugging of complex robotics systems, and the implementation of robust error handling and recovery mechanisms that maintain system stability under challenging operational conditions.

6.3. Research Impact and Broader Implications

This section will examine the broader research impact and implications of the Tino V2 project for the field of social robotics and related disciplines. The social robotics advancement will be assessed first, covering how the demonstrated VR integration capabilities advance the state of the art in telepresence robotics and remote human-robot interaction, the contribution to understanding of user experience design principles for VR-controlled robotics systems, and the validation of practical approaches for implementing complex social robotics capabilities on accessible hardware platforms. The sensor fusion research impact will be detailed, covering the practical demonstration of indoor localization solutions

that address common challenges faced by mobile robots in GPS-denied environments, the contribution to understanding of sensor fusion trade-offs and optimization strategies for real-time embedded applications, and the validation of UWB positioning as a complementary technology for visual SLAM systems. The embedded AI research contributions will be examined, covering the successful demonstration of real-time human pose detection on embedded hardware through systematic optimization approaches, the contribution to understanding of model optimization trade-offs for practical robotics applications, and the validation of TensorRT and similar optimization frameworks for enabling advanced AI capabilities on resource-constrained platforms. The interdisciplinary research impact will be addressed, covering the contribution to understanding of effective integration strategies for combining computer vision, robotics, networking, and VR technologies in cohesive systems, the advancement of user-centered design principles for complex technical systems that must be accessible to non-technical users, and the demonstration of practical approaches for bridging virtual and physical environments in meaningful and reliable ways.

6.4. Limitations and Challenges

This section will provide honest assessment of the limitations encountered and challenges that remain unresolved in the current Tino V2 implementation. The technical limitations will be acknowledged first, covering the remaining dependence on UWB anchor infrastructure that limits deployment flexibility and requires careful setup and calibration procedures, the computational constraints that, while significantly improved, still impose trade-offs between system capability and real-time performance requirements, and the environmental dependencies that affect system performance under certain lighting conditions, visual feature scarcity, or UWB signal obstruction scenarios. The mechanical system limitations will be addressed, covering the ongoing reliability challenges with the Stewart platform head mechanism despite improvements with heim joint implementations, the battery life constraints that limit extended operation periods and require careful power management, and the mechanical wear issues that affect long-term system reliability and require ongoing maintenance and replacement of components. The VR integration limitations will be examined, covering the network latency and reliability dependencies that can affect user experience during remote operation, the learning curve required for users to effectively operate the VR control system despite efforts to create intuitive interfaces, and the limited environmental feedback that restricts user awareness of the robot's physical surroundings and potential obstacles. The scalability challenges will be discussed, covering the manual calibration and setup requirements that limit rapid deployment in new environments, the complexity of system integration that requires technical expertise for

maintenance and troubleshooting, and the current limitation to laboratory and controlled environments that restricts broader real-world deployment and validation.

6.5. Future Work and Research Directions

This section will outline promising directions for future research and development that build upon the foundations established in the Tino V2 project. The localization enhancement opportunities will be presented first, covering the potential for implementing fully autonomous UWB anchor deployment and calibration systems that reduce setup complexity and increase deployment flexibility, the exploration of additional sensor modalities such as LiDAR or thermal imaging that could further improve localization robustness and environmental perception, and the development of adaptive sensor fusion algorithms that dynamically adjust fusion parameters based on environmental conditions and sensor reliability. The VR interaction advancement possibilities will be detailed, covering the potential for implementing more sophisticated haptic feedback systems that provide users with tactile information about the robot's environment and interactions, the development of multi-user VR systems that enable collaborative control and interaction with the robot platform, and the exploration of augmented reality integration that combines real-world robot camera feeds with virtual overlay information for enhanced user experience. The human-robot interaction research directions will be examined, covering the potential for implementing more advanced social behavior algorithms that respond appropriately to detected human emotions and social cues, the development of learning systems that adapt robot behavior based on individual user preferences and interaction patterns, and the exploration of multi-robot coordination capabilities that enable collaborative social robotics applications. The broader deployment opportunities will be addressed, covering the potential for adapting the developed technologies for healthcare applications such as elderly care and patient monitoring, the exploration of educational applications that leverage the VR integration for remote learning and interaction, and the investigation of commercial applications that could benefit from reliable telepresence robotics with advanced perception and interaction capabilities.

6.6. Final Reflections and Conclusions

This section will provide final reflections on the Tino V2 project and draw overall conclusions about the research achievements and their significance for the field of social robotics. The project success assessment will be established first, covering the achievement of all major technical objectives including successful system migration, sensor fusion implemen-

tation, VR integration, and human detection capabilities, the validation of key hypotheses about the feasibility and effectiveness of combining multiple advanced technologies in a cohesive social robotics platform, and the demonstration of practical approaches for addressing common challenges in indoor mobile robotics and human-robot interaction. The methodological insights will be reflected upon, covering the value of systematic technology evaluation and selection processes for complex multi-component systems, the importance of modular architecture design for enabling effective development and debugging of advanced robotics systems, and the critical role of user-centered design principles for creating accessible and effective human-robot interaction systems. The research contribution significance will be assessed, covering how the Tino V2 project advances the state of the art in VR-integrated social robotics through practical demonstration of reliable technologies and effective integration strategies, the contribution to broader understanding of sensor fusion approaches for indoor robotics applications, and the validation of embedded AI optimization techniques for real-time perception and interaction capabilities. The legacy and impact considerations will be addressed, covering the foundation established for future social robotics research through open-source code, comprehensive documentation, and validated system architectures, the potential for broader adoption of demonstrated technologies and approaches by other researchers and developers, and the continued evolution toward more sophisticated and accessible human-robot interaction systems that effectively bridge virtual and physical environments for meaningful social applications.

7 | 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	[11] – Robust monocular/Stereo SLAM. [?] for Monocular and Multicamera Systems
IMU + Wheel Encoders	Low cost; integrates motion data	Drift over time; Stewart tilt issues	[22] – Kalman filtering. [6] for Mobile Robot Localization
UWB-IR	Small footprint; could work with fabric	Requires external anchors	[1] for Indoor Robot Navigation.

External Sensing

Technology	Pros	Cons	Key Papers & Resources
UWB Anchors	High accuracy; no line-of-sight	Setup/calibration required	[14] in non-cooperative industrial environments
AprilTags	Low cost; precise	Line-of-sight; limited area	[29].
MoCap Systems	Sub-mm accuracy	Expensive; fixed environment	[30] – Industrial use cases.

Orientation Technologies

- **Sensor Fusion:** [22] filters for combining UWB, IMU, and encoders. (Waiting for access request)
- **UWB Ori:** [8] with ultra-wideband signals
- **NLOS Mitigation:** [26] and Tracking With Arbitrary Target Orientation, Optimal Anchor Location, and Adaptive NLOS Mitigation
- **RPO:** [2] using UWB

Human Detection Technologies

Onboard Sensing

Technology	Pros	Cons	Key Papers & Resources
Thermal Cameras	Works in darkness; fabric-friendly?	No depth; limited range	[3] – CNN-based approaches.
Ultrasonic Array	Low cost; proximity detection	No human distinction	[4] (In Korean).
Upgraded Camera	Wider FOV; ML-compatible	Fabric obstruction; compute-heavy	[37] – Real-time object detection.

External Sensing

Technology	Pros	Cons	Key Papers & Resources
RGB-D Cameras	Depth data; multi-human tracking	Fixed installation	[12]. [5] – Performance Analysis of Body Tracking.
LiDAR	High-resolution 3D mapping	Expensive; compute-heavy	[38].
WiFi/Radar	Privacy-friendly; fabric-penetrating	Lower resolution	[7]: A New Way to Observe Surroundings

Technologies for Tino Robot Implementation

Localization Technologies

Visual Odometry (VO)

- **Variants:**
 - *ORB-SLAM3* (supports RGB-D): [35]
 - + Synergy with human detection via depth data
 - + Robust feature matching for dynamic environments
 - Higher computational cost (requires GPU optimization)
 - *SVO* (Semi-direct Visual Odometry): [36]
 - + 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 \Rightarrow Requires:
 - IMU sensor fusion (Kalman filtering)
 - RPO/UWB/Ori 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 $<5\text{m}$)

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 R&D 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 Realsense 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 thought 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 outputting some video, also it took a long time to actually grasp 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 scratch. 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 accurate) and they did not have any atlas/map management so I scraped that work. Given that with 2 systems i had similar issues i thought 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 functionality 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 mentioned was using RtabMap, so that's 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 didn't 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 tried again with the standalone version and this time I was able to make it link with the depthAI, and it worked amazing, 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_ros and the rtabmap_ros, it publishes the localization_pose topic that has all of the important information

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 19 Major milestone reached: Complete system architecture migration from legacy Raspberry Pi based system to ROS2 based implementation. Restructured the entire workspace by creating tino_ws for ROS2 development and moved all legacy code to legacy_tino folder for preservation.

Implemented the core ROS2 node architecture:

- `gamepad_node.py`: Handles Xbox controller input with proper D-input to X-input conversion for Jetson compatibility
- `hardware_interface_node.py`: Manages serial communication with all 3 Arduino systems (head, base, leg) using proper device symlinks
- `robot_controller_node.py`: Central coordination node that manages all robot behaviors and movement commands
- `vr_interface_node.py`: Handles VR system integration and data exchange for future Unity integration

Created launch files for both mapping (`rtab_mapping.launch.py`) and localization (`rtab_localization.launch.py`) modes, allowing seamless switching between SLAM creation and navigation modes.

This migration allows for much better modularity, debugging capabilities, and integration with the VR system compared to the monolithic Python scripts from the legacy system.

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.

Apr 23 Major advancement in human detection capabilities: Implemented pose detection functionality using YOLOv11 with TensorRT optimization. The system now provides real-time human skeleton tracking with 17 key body joints detection, including depth information using the stereo camera data. This allows Tino to not only detect humans but also track their pose and calculate their 3D position relative to the robot.

Added audio handling nodes (`audio_node.py` and `audio_loopback.py`) and fully integrated audio functionality into both VR and robot controller systems. The audio system now supports bidirectional communication: capturing audio from the omnidirectional microphone and publishing it to VR, while also receiving audio from VR and playing it through the speakers.

Enhanced gamepad handling with improved command processing and error reporting, making the control system more robust and responsive.

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 dissassembling 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 we 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

May 8 Completed major refactor of serial communication and motor control systems for the new differential drive base implementation. Created new_base_tino.ino with enhanced PID controller specifically designed for the differential drive system, replacing the old omnidirectional control logic.

Updated hardware_interface_node.py with improved serial port configurations, added comprehensive debugging logs, and enhanced command handling to work with the new base architecture. The new system uses proper device symlinks (/dev/ttyBASE, /dev/ttyHEAD, /dev/ttyLEG) to ensure consistent Arduino connections.

Created upload scripts (upload_new_base.sh, setup_arduino_symlinks.sh) for easier development and deployment workflow.

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 airtlab 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 were working again and some fine tuning had to be done to the gamepad (we changed from D input to X input because the jetson did not have 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 these 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.

these are the issues to solve by next week

May 13 This week started by trying to solve the wheel issue, we tried to fill the wheels with hot glue, this worked perfectly, the wheels did not de-bead and the traction was good. Also create a “bumper” to avoid the fabric to get stuck in the wheels, this works on most of the scenarios but there are still some cases that it may get stuck, so we will need to keep testing it.

I didn't have time to create the shell for the camera

May 15 Implemented comprehensive VR data recording functionality for Unity integration. Created `vr_data_recorder_node.py` and `vr_data_extractor.py` to capture and process all robot sensor data, human pose detection, audio streams, and robot state information for VR system development and testing.

The recording system includes service controls for starting/stopping data capture, meta-

data management, and proper data synchronization across all robot systems. This allows for detailed analysis of robot behavior and human-robot interactions for VR system optimization.

Enhanced VR interface message structure documentation and improved serial port configurations for better reliability during VR data exchange.

May 20 before continuing with the camera shell we had a new issue. The current arms for the head were breaking, this was because the head is too heavy and has very aggressive moves, also given the 3-Dof Stewart platform it has a lot of flex in the arms, so we had to change the arms to a more robust design. Our first hypothesis was that the current design of the servo axis was not aligned with the head axis, this could cause the servos to get damaged as the force was being applied to the servo axis and not to the head axis, so we designed a new arm that has the servo axis aligned with the head axis, this way the force is applied directly to the head and not to the servo axis. This new model was designed in Inventor and printed in PLA. The idea of this piece is that it can hold the arm that goes to the top from the middle and have it properly tight, also aligned the servo axis with this arm axis and then the head axis. The pieces were printed and fixed to the head, this worked but we still saw some flex in the arms, that we think is acceptable so we will keep testing it.

May 27 This week I started by trying to create the camera shell, this was done by creating a shell that can be attached to the tripod system that was used to hold the camera on time, this shell needed to have some important features. It needed to of course hold the camera, but it had to have an empty space in the back to allow the camera to cool down, also it needed to have a way to attach it to the tripod system, and finally it needed to have 2 flaps at the top and at the bottom so that we can use them as a point to glue some velcro to hold the fabric in place and avoid it to get in the camera FOV.

Jun 3 This week we finalized the camera shell, we attached the velcro to the flaps and the other side of velcro to the fabric by sewing it. Also we glued a mesh so that we can hide the camera and avoid it to be seen. We tested the camera and it worked perfectly, the fabric did not get in the camera FOV and the camera was able to cool down properly, also the camera was able to see through the fabric so it was able to detect humans.

Also this week the head arms we printed broke, we thought it was because of the 3d printed layer orientation and force applied, so we redesigned the arms to have a more robust design, this time we printed it in the other orientation so that the layers orientation is perpendicular to the force applied, this way we hope that the arms will not break again.

June 24–July 1 This week we got the new brackets for holding the new more powerful motors, this change was done because the old motors suffered a lot of heat due to tino weight. This new ones can support more weight and have a better heat dissipation the only issue they had is that the old motor bracket was not compatible so we had to wait for the new ones to arrive. this new bracket proved difficult to implement as the holes they had where not aligned with the item profiles, also the motors axis was more up so tino was dragging on the floor To fix both of this issues I created a spacer with some metal square profiles, this way the motors where aligned with the item profiles and also the axis was at the right height so that tino could move freely.

Also with this new motors I had to redo some cable connections to the encoders, power and driver simplifying the desing and connections

We tested the new motors and they worked perfectly, there was no dragging and the wheels did not de-bead, also the traction was good and the speed was good enough for tino to move around. We only had to tweak a bit the PID values to make it more stable and not overshoot the target position

Enhanced Raspberry communication system with special command processing and periodic status updates in the main loop for improved reliability and debugging capabilities. This allows for better monitoring of Arduino systems and more robust error handling.

Implemented audio messages system with chime notifications, providing auditory feedback for system events and user interactions. This enhances the user experience and provides clear audio cues for various robot states and actions.

July 1 After a period of use the arms for the head broke again, There is still a lot of flex in the arms, so we decided to redesign the arms again, this time we decided to change the approach. Doing some research on Stewart platforms we found that the arms should have rod end (heim joints) on both ends, this way the arms can move freely and not have any flex. The current design was using a bearing on the servo side and a rod end on the head side, this was not ideal as the bearing was not allowing the arm to move freely, so we redesigned the arms to have rod ends on both sides, this way the arms can move freely and not have any flex. The new design was printed in PLA and combined with some metal heim joints, this way the arms can move freely and not have any flex. The only issue this created was that when stationary the head had some wobble, this simple because the head is held by the 3 arms that can move thanks to the rod end. We dont think this is a major issue as it may add a bit to the “expresiiveness” of tino head movement.

July 8 Based on the VR system I had to modify the current system of movement to better

integrate with the VR environment. The old system using the gamepad was broadcasting a message of movement to the 3 arduinos, this was so that the head, leg, and base could move synchronously. But now in the VR system we need to “independicese” this 3 system.

The head is controlled by the VR movement, this already works given the head command topics, we only had to remove the defined “routines” that it had when moving forward and to the sides. The leg now needs to be controlled with 4 different states. Previously the leg was just resting and when a movement command was sent it did a whole continuous rotation based on the sinusoidal system that was created originally. Now because this new system the idea is that the user in the VR needs to “drag” the same way tino “drags” the leg in the real world, so for this we created 4 states:

0. Resting

1. little push (this was done so that tino can express a “pointing” to something or “looking” at something to try and get the attention of the user)
2. leg forward (moves the leg to the max reach before doing the dragging movement)
3. leg backward (moves the leg back doing the “dragging” movement up to the resting position)

We also had to modify the base movement, this was done by removing the old gamepad movement and implementing this new 4 state system:

0. Resting

1. little push (the base moves forward and backward a bit very fast to express a “pointing”)
2. Does nothing, because at this point the leg is just being moved forward to prepare for the dragging
3. move forward (this is the dragging movement, the base moves forward a determined amount of time to simulate the dragging movement)

In order to make all of these systems work properly we had to make them atomic, and also these new states will be sent as pulses instead of continuous movement. This way the user can control tino in a more natural way, and also the user can express more emotions and actions with tino.

While doing the leg changes we noticed something interesting, the PWM that the driver gets for negative values for some reason is not the usual range from 0 to 255 the motor that moves the leg goes forward from very slow to very fast (as intended) but as one would

expect the negative values to go backward, it does not, it goes slow at -235 and fast at -1. This is not an issue as we can create the respective system to make it work with this values, but it is something to keep in mind for the future.

July 10–13 Enhanced VR interface with improved odometry checking and loss detection, simplifying the condition logic for more reliable operation. Updated motor speed and angular parameters for improved performance and smoother robot movement.

Added comprehensive logging improvements across pose detection and VR interface nodes, reducing log verbosity while maintaining essential debugging information. Modified skeleton publish rate in robot controller for optimal performance balance between real-time tracking and system resources.

Improved ROS-TCP-Endpoint submodule with enhanced error handling for more robust Unity-ROS2 communication bridge.

July 15 This week we prepared the experiment but we also had to finalize the atomic movement system and implement the complete 4-state management for both leg and base controllers.

The major achievement was completing the synchronization between the leg and base atomic movements. We implemented a sophisticated locking system where the base controller can only execute case 3 (forward movement) after the leg controller has completed case 2 (leg extension). This ensures that Tino's movements are perfectly coordinated - the leg extends first to prepare for dragging, then the base moves forward to simulate the actual dragging motion.

We also implemented a pending command system that was crucial for the VR integration. When the user in VR triggers a movement command while case 2 is still running (leg extending), the system stores the command and automatically executes it once case 2 completes. This allows for natural user interaction without losing commands or creating timing conflicts.

The rotation system was expanded to work atomically as well. We implemented cases (3,1) and (3,-1) for right and left rotation respectively, maintaining the same 1.7-second duration as forward movement for consistency. This means the VR user can trigger rotation movements that are perfectly timed and atomic just like the forward movements.

One of the most important improvements was the pulse system we implemented in the gamepad node. Instead of continuous signals, each button press now generates a 3-cycle command pulse that automatically returns to idle. This pulse approach is essential for the VR system because it ensures each user action in VR corresponds to exactly one complete

movement in the physical robot.

We also enhanced the debug system significantly, adding timing logs and state tracking that shows exactly when each phase starts and completes. This was crucial for fine-tuning the 1.5-second case 2 timing cycle and the 1.7-second case 3 movements to ensure perfect synchronization.

The atomic movement system now guarantees that once any movement starts, it must complete fully before another can begin. This prevents partial movements and ensures the VR user always sees Tino complete the action they initiated, creating a much more natural and predictable interaction experience.

As a cleanup task, we also removed the old left joystick analog control since everything now runs through the discrete 4-state button system, making the control scheme consistent with the VR approach.

Added a system to offset the map position by a desired amount of degree, this because it's difficult to control the alignment of the SLAM map when creating it and some times it can be created with a rotation, this in principle is no problem, but given that we need to map this position directly into unity is best if we have the map aligned with the cartesian plane. This value can be changed but the process needs to be trial and error until we find a desired angle, this may change from map to map

After quite the amount of setup for mapping, we had to add a lot of landmarks to help the system not get lost when close to a wall, even tho the system work most of the time there are scenarios where it drifts by at maximum 1.20m we did some test setting 4 positions in the floor and then driving tino around those multiple times, this is the findings

button it1 position: x: -9.07916381186811 y: -3.0198425113288256 z: 0.21915075182914734
orientation: x: -0.02356615282947662 y: 0.03717912817458111 z: 0.9987406916737919 w:
-0.024071783643750885 it2 position: x: -8.186847680416442 y: -3.339181554181038 z:
0.22911125421524048 orientation: x: -0.03578283275599135 y: -0.014825521998614472 z:
0.9886076124808023 w: -0.1454468531069043

i3 after spinning position: x: -9.132201829319898 y: -3.03850808312142 z: 0.16989609599113464
orientation: x: -0.023309146346151374 y: 0.013430598998060258 z: 0.9970844577607847
w: -0.07140670417222031

door it1 position: x: -6.302746616280681 y: -2.937899155623209 z: 0.04246218502521515
orientation: x: -0.041125382354844545 y: 0.04075343561185359 z: 0.9381911838945137
w: -0.3412406188158956 it2 position: x: -7.688149884473027 y: -2.758148921953637 z:

0.26177895069122314 orientation: x: -0.056920978093906345 y: 0.06027535966552371 z: 0.9105215513414311 w: -0.4050646707808027

platform it1 position: x: -5.133887212339737 y: -0.8823759434488573 z: 0.003330887295305729 orientation: x: 0.037969363959043455 y: 0.042694832314988815 z: -0.7221974378925629 w: 0.6893232444590183 it2 position: x: -4.155619022416493 y: -1.1973306038245426 z: 0.18429601192474365 orientation: x: -0.0423178727432749 y: -0.154526999071549 z: 0.8307654381749795 w: -0.5330660364450295

middle bridge it1 position: x: -2.967067619346332 y: -1.0099800292205212 z: 0.020707421004772186 orientation: x: -0.011650759024720693 y: 0.020886688418633118 z: 0.9882899125201247 w: -0.1507018578293635

it2 position: x: -3.0019163829042084 y: -0.9073216003302291 z: 0.1694898009300232 orientation: x: -0.0432093148775227 y: 0.026300433857604333 z: 0.9952330147461537 w: -0.08338188177412909

base on all of this we found the only way to make the system "undrift" or to properly relocalize is to make tino spin, this helps rtab to get more features around and try to correct the position. even tho this is a solution for other scenarios, because we need to create a system that will be manage via the vr by a user with no technical expertise and also because part of the experiemt is let the user figure out stuff we cant tell the user to just "spin around" until it finds the proper relocalizatio also, there is no progragmatic way to find this issue of the drift in the first place, given that this position given is absolute we cant know automatically that we are drifting to try and correct

At least the orientation worked really good that even if on the positionion is lost we still had good orientation so we decided the following we will implement a different system for localization, we will use UWB to mange the global positionion and then we will use the already in place camera and RTAB slam to get the orientation By doing this fusion we hope to get a stable and reliable position adn orientation

Now with the new UWB system implemented we have a better and precise global positioning We did tests on the same 4 spots 3 times each and this was the output

Bridge it1 position: x: 2.95 y: 1.28 z: 0.96 orientation: x: -0.016244746081379848 y: 0.011590778809634706 z: 0.9978177685970177 w: -0.06294018257794398 it2 position: x: 2.92 y: 1.24 z: 1.04 orientation: x: 0.03365805896384339 y: -0.10581931225218563 z: 0.9615574311256412 w: -0.25115088467150715 it3 position: x: 2.96 y: 1.22 z: 0.98 orientation: x: 0.06318550106034526 y: 0.018786558120585363 z: 0.965924510222013 w: -0.2502889073188361

platform it1 position: x: 5.49 y: 1.26 z: 0.98 orientation: x: 0.05060387003762118
 y: 0.04387060236978493 z: -0.7095551199862337 w: 0.7014599325016385 it2 position:
 x: 5.42 y: 1.45 z: 0.97 orientation: x: -0.044399141185597304 y: 0.00893117279072231
 z: -0.7339342771249334 w: 0.6777089090442352 it3 position: x: 5.45 y: 1.21 z: 1.02
 orientation: x: 0.0001595946587705419 y: 0.02375217746987014 z: -0.8009830471763693
 w: 0.598215889888112

door it1 position: x: 6.45 y: 2.72 z: 0.97 orientation: x: -0.02131269555403667 y:
 0.05062973708550628 z: 0.9471005285691299 w: -0.3162009876538764 it2 position: x:
 6.42 y: 2.67 z: 0.96 orientation: x: -0.031268703585232954 y: 0.03502103116732864
 z: 0.9344169219027985 w: -0.3530734466584695 it3 position: x: 6.43 y: 2.69 z: 0.95
 orientation: x: -0.026643789303690477 y: 0.03844515148744251 z: 0.9429357383344411
 w: -0.32967307363487713

button it1 position: x: 7.96 y: 2.72 z: 1.01 orientation: x: -0.0060182617982888105
 y: 0.04144030272178065 z: 0.9977658808453272 w: -0.0520551769152283 it2 position: x:
 7.89 y: 2.72 z: 1.01 orientation: x: -0.004659043734153017 y: 0.040590124652731686
 z: 0.9970509609184363 w: -0.06496374337391156 it3 position: x: 7.85 y: 2.66 z: 0.98
 orientation: x: -0.005168897035844065 y: 0.021907294998073198 z: 0.9981057371155894
 w: -0.05725740236429487

also doing the testing we lost a wheel, the plastic wheel hub broke. We fixed it by
 hotglueing the wheel hub again but thats something may need to be changed in the
 future

Bibliography

- [1] Anonymous. A uwb-ir based localization system for indoor robot navigation. In *IEEE International Conference*, 2007. URL <https://ieeexplore.ieee.org/abstract/document/4380919>.
- [2] Anonymous. Measurement of relative position and orientation using uwb. *IEEE*, 2019. URL <https://ieeexplore.ieee.org/abstract/document/8827149>.
- [3] Anonymous. Real-time implementation of human detection in thermal imagery based on cnn. *ResearchGate*, 2019. URL https://www.researchgate.net/publication/330762720_Real-Time_Implementation_of_Human_Detection_in_Thermal_Imagery_Based_on_CNN.
- [4] Anonymous. Ultrasonic human tracking. *Korea Science*, 2020. URL <https://koreascience.kr/article/JAK0202019962560668.pdf>. In Korean.
- [5] Anonymous. Azure kinect for robotics – performance analysis of body tracking. *IEEE*, 2021. URL <https://ieeexplore.ieee.org/abstract/document/9480177>.
- [6] Anonymous. Extended kalman filter sensor fusion in practice for mobile robot localization. *IJACSA*, 13(2), 2021. URL https://thesai.org/Downloads/Volume13No2/Paper_4-Extended_Kalman_Filter_Sensor_Fusion_in_Practice.pdf.
- [7] Anonymous. Rf-sensing: A new way to observe surroundings. *IEEE*, 2022. URL <https://ieeexplore.ieee.org/abstract/document/9982449>.
- [8] Anonymous. Uwbori: enabling accurate orientation estimation with ultra-wideband signals. *HAL Science*, 2024. URL <https://hal.science/hal-04754129/>.
- [9] J. Borenstein and L. Feng. Measurement and correction of systematic odometry errors in mobile robots. *IEEE Transactions on Robotics and Automation*, 12(6): 869–880, 1996. doi: 10.1109/70.544770.
- [10] C. Breazeal. Toward sociable robots. *Robotics and Autonomous Systems*, 42(3):167–175, 2003. ISSN 0921-8890. doi: [https://doi.org/10.1016/S0921-8890\(03\)00030-9](https://doi.org/10.1016/S0921-8890(03)00030-9).

- 1016/S0921-8890(02)00373-1. URL <https://www.sciencedirect.com/science/article/pii/S0921889002003731>. Socially Interactive Robots.
- [11] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós. Orbslam3: An accurate open-source library for visual, visual-inertial, and multimap slam. *IEEE Transactions on Robotics*, 2021. URL <https://arxiv.org/abs/2007.11898>.
 - [12] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh. Openpose: Real-time human pose detection. *arXiv preprint arXiv:1812.08008*, 2019. URL <https://arxiv.org/abs/1812.08008>.
 - [13] S. Cardillo. Conveying illusion of life and empathy through a non-anthropomorphic robot and non-verbal communication: The tino experience. Bachelor’s thesis, Universidad de los Andes, 2024. Previous work on the original Tino robot system.
 - [14] R. Chen et al. Robot vision ultra-wideband wireless sensor in non-cooperative industrial environments. *International Journal of Advanced Robotic Systems*, 2018. URL <https://journals.sagepub.com/doi/full/10.1177/1729881418795767>.
 - [15] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: part i. *IEEE Robotics & Automation Magazine*, 13(2):99–110, 2006. doi: 10.1109/MRA.2006.1638022.
 - [16] P. Ekman, W. Friesen, M. O’Sullivan, A. Chan, I. Diacoyanni-Tarlatzis, K. Heider, R. Krause, W. LeCompte, T. Pitcairn, P. Ricci Bitti, K. Scherer, M. Tomita, and A. Tzavaras. Universals and cultural differences in the judgments of facial expressions of emotion. *Journal of personality and social psychology*, 53:712–7, 10 1987. doi: 10.1037/0022-3514.53.4.712.
 - [17] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry, 2016. URL <https://arxiv.org/abs/1607.02565>.
 - [18] C. Forster, M. Pizzoli, and D. Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22, 2014. doi: 10.1109/ICRA.2014.6906584.
 - [19] A. Geiger, J. Ziegler, and C. Stiller. Stereoscan: Dense 3d reconstruction in real-time. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pages 963–968, 2011. doi: 10.1109/IVS.2011.5940405.
 - [20] S. Gezici, Z. Tian, G. Giannakis, H. Kobayashi, A. Molisch, H. Poor, and Z. Sahinoglu. Localization via ultra-wideband radios: a look at positioning aspects

- for future sensor networks. *IEEE Signal Processing Magazine*, 22(4):70–84, 2005. doi: 10.1109/MSP.2005.1458289.
- [21] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments. *The International Journal of Robotics Research*, 31(5):647–663, 2012.
- [22] L. Huang et al. Sensor fusion for mobile robot localization using extended kalman filter uwb tof and aruco markers. *ResearchGate*, 2018. URL https://www.researchgate.net/publication/356744906_Sensor_Fusion_for_Mobile_Robot_Localization_Using_Extended_Kalman_Filter_UWB_ToF_and_ArUco_Markers.
- [23] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 225–234, 2007. doi: 10.1109/ISMAR.2007.4538852.
- [24] M. Labbé and F. Michaud. Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. *Journal of Field Robotics*, 36(2):416–446, 2019. doi: <https://doi.org/10.1002/rob.21831>. URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21831>.
- [25] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee, W.-T. Chang, W. Hua, M. Georg, and M. Grundmann. Mediapipe: A framework for building perception pipelines, 2019. URL <https://arxiv.org/abs/1906.08172>.
- [26] J. Luo et al. Uwb system for indoor positioning and tracking with arbitrary target orientation, optimal anchor location, and adaptive nlos mitigation. *IEEE Access*, 2020. URL <https://ieeexplore.ieee.org/abstract/document/8989827>.
- [27] A. Mehrabian. *Silent Messages*. Wadsworth Publishing Company, 1971. ISBN 9780534000592. URL <https://books.google.com.co/books?id=Ast-AAAAMAAJ>.
- [28] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. doi: 10.1109/TRO.2015.2463671.
- [29] E. Olson. Apriltag: A robust fiducial system. In *IEEE International Conference on Robotics and Automation*, 2011. URL <https://april.eecs.umich.edu/papers/details.php?name=olson2011tags>.

- [30] OptiTrack. Optitrack for robotics – industrial use cases, 2024. URL <https://www.naturalpoint.com/optitrack/applications/robotics/>.
- [31] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer, 2020. URL <https://arxiv.org/abs/1907.01341>.
- [32] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection, 2016. URL <https://arxiv.org/abs/1506.02640>.
- [33] D. Scaramuzza and F. Fraundorfer. Visual odometry [tutorial]. *IEEE Robotics & Automation Magazine*, 18(4):80–92, 2011. doi: 10.1109/MRA.2011.943233.
- [34] T. Sheridan. Musings on telepresence and virtual presence. *Presence*, 1:120–125, 01 1992. doi: 10.1162/pres.1992.1.1.120.
- [35] UZ-SLAMLab. Orb-slam3 github repository, 2024. URL https://github.com/UZ-SLAMLab/ORB_SLAM3.
- [36] UZH-RPG. Svo pro open github repository, 2024. URL https://github.com/uzh-rpg/rpg_svo_pro_open.
- [37] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detection. *arXiv preprint arXiv:2207.02696*, 2022. URL <https://arxiv.org/abs/2207.02696>.
- [38] Z. Yan et al. Lidar-based human detection. *Autonomous Robots*, 2019. URL <https://link.springer.com/article/10.1007/s10514-019-09883-y>.

List of Figures

List of Tables

Acknowledgements

Here you may want to acknowledge someone.

