ROBOTS IN ASSISTED
LIVING ENVIRONMENTS

UNOBTRUSIVE, EFFICIENT, RELIABLE AND MODULAR SOLUTIONS FOR INDEPENDENT AGEING

Research Innovation Action

Project Number: 643892 Start Date of Project: 01/04/2015 Duration: 36 months

DELIVERABLE 4.7

Integrated Robotic Platform II

Dissemination Level	**Public**
Due Date of Deliverable	Project Month 30, 30 September 2017
Actual Submission Date	Project Month 34, 30 January 2018
Work Package	WP4*: Physical home architecture development and integration of cost-effective, reliable and power-efficient RADIO components for elder monitoring and caring*
Task	T4.3: *Robotic platform design and integration*
Lead Beneficiary	ROBOTNIK
Contributing beneficiaries	NCSR-D
Type	DEM
Status	Submitted
Version	Final

Project funded by the European Union's Horizon 2020 Research and Innovation Actions

Abstract

This deliverable reports on the RADIO robotic platform first design, corresponding to the work done at Task 4.3 during months 16 to 34. This document gives the design of the RADIO Robot, both from the hardware and software side, and points to the relevant repositories for the delivered software.

History and Contributors

Ver	Date	Description	Contributors
00	9 June 2017	Document structure, assuming D4.6 as a starting point.	NCSR-D
01	19 Dec 2017	Updates to Section 2, with most new material in Subsection 2.2.	ROBOTNIK
02	11 Jan 2018	New Subsection 3.3 describing new software components and edits to the rest of Section 3 with updates on software already in D4.6.	NCSR-D
03	18 Jan 2018	New demonstrations, based on the new robot prototype (Section 4).	NCSR-D
04	29 Jan 2018	Internal peer review	AVN
05	30 Jan 2018	Addresses peer review comments	NCSR-D, ROBOTNIK
Fin	30 Jan 2018	Final preparations and submission.	NCSR-D

Abbreviations and Acronyms

NCSR-D	National Centre for Scientific Research “Demokritos”
TWG	Technical Educational Institute of Western Greece
RUB	Ruhr Universitaet Bochum
ROBOTNIK	Robotnik Automation SLL
S&C	Sensing & Control Systems S.L.
AVN	AVN Innovative Technology Solutions Ltd.
FSL	Fondazione Santa Lucia
FHAG	Fundació Hospital Asil de Granollers
FZ	Frontida Zois
ADL	Activities of Daily Life
BLE	Bluetooth Low Energy
CPS	Cyber-Physical Systems
ICT	Information and Communications Technology
RTD	Research, Technological Development
SME	Small Medium Enterprise
ROS	Robot Operating System

CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

1.1 Purpose and Scope

This deliverable demonstrates the first RADIO Robot prototype, including the hardware design and the integration of the ADL recognition prototypes developed in WP3. Within the scope of this deliverable is to demonstrate the hardware prototype and to publish the source code of the software developed for the robot, besides the ADL recognition software.

1.2 Approach

This deliverable is prepared within *Task 4.3: Robotic platform design and integration*. This task covered all aspects of hardware design and integration, developed drivers and controllers for the hardware components, and the development of software components related to the robot, besides the ADL recognizers which have already been integrated into the first robot prototype.

1.3 Relation to other Work Packages and Deliverables

Besides work on the robot platform itself reported here, this deliverable also updates the previous prototype (D4.6) with relevant developments in the rest of WP4, as reported in D4.3 *Architecture for extending smart homes with robotic platforms*. The resulting prototype is used for the final *Prototype of the integrated RADIO Home* (D4.9).

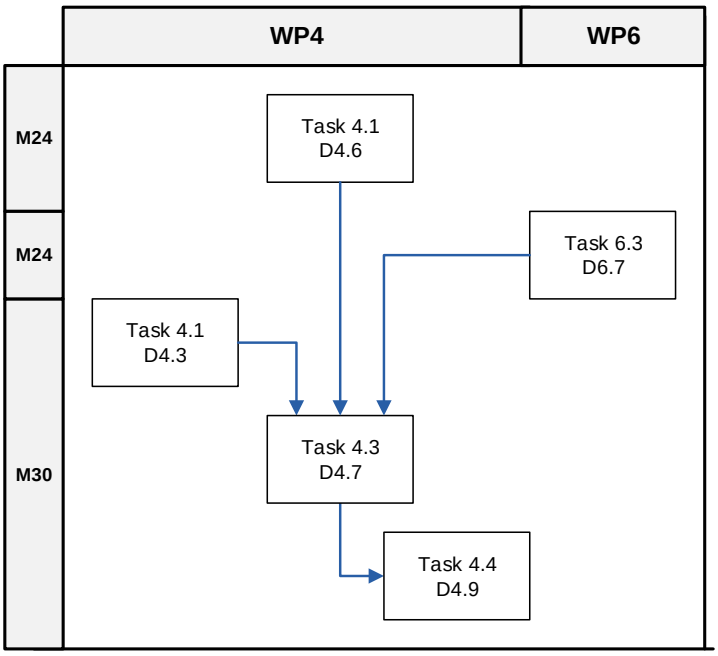

Figure 1: Relation to other Work Packages and Deliverables

2 PROTOTYPE HARDWARE

2.1 Robotic Base

The starting point for the RADIO robotic platform is the TurtleBot2 mobile robot. TurtleBot2 is an open robotic platform designed for education and research on state of the art robotics. It is also a powerful tool to teach and learn ROS and make the most of its cutting edge technology. Equipped with a 3D sensor, it can map and navigate indoor environments. Its highly accurate odometry, amended by a factory-calibrated gyroscope, enables precise navigation.

TurtleBot 2 (Kobuki base) is the new version of the successful platform TurtleBot and it has many advantages over its predecessor: odometric measurement precision, open protocol, greater autonomy, greater load, higher speed, and greater mobility, larger diameter wheels and capacity to overcome obstacles up to 12 mm. Figure 2 presents the TurtleBot2 and Table 1 gives its standard specifications.

The main reasons for using this platform as a base for the RADIO robot development have been the following ones:

- Great quality price ratio, allowing the project to have multiple units for experimentation and piloting
- Excellent software support. All the software controllers are open source.
- Robustness: It is based on a standard and commercial robot cleaner sold by the manufacturer of this robot.
- Open platform ideal for experimentation and adding new hardware and software modules.

Figure 2: TurtleBot2 robotic platform

Table 1: Standard technical specifications of the TurtleBot2 base	
Dimensions	31.5 x 43 x 34.7 cm
Weight	5 kg, also depending on configuration
Load capacity	5 kg
Speed	0.65 m/s
Controller	Open architecture ROS
Autonomy	7h. (big battery), 3h. (small battery)

2.2 Hardware Integration and Modifications

In order to accommodate all the hardware needs of the RADIO project, a number of hardware components have been integrated into the RADIO Robot. Here is a list of these components:

- Orbbec Astra 3D camera mounted on an adjustable telescopic bar
- HOKUYO-UST10LX 2d laser and mounting board
- On-board PC: INTEL NUC BOXNUC5CPYH Celeron N3050,4GB RAM, 120GB SSD
- Asus RT-N12 Router (and mounting board)
- Emergency stop button
- TB2 RADIO covers
- Logitech gamepad for teleoperation
- Big Battery Pack 5x Batteries 4S2P in parallel (22 Ah)
- Docking station for autonomous charge
- Battery charger 4A

Furthermore, the original mobile base has had to be modified in some ways. Some changes in the power system were necessary to provide longer battery autonomy as well as mounting all the sensors needed for the project's localization, navigation and perception tasks. Safety, performance, communications have been also taken in account for the design of the RADIO Robot.

2.2.1 Battery System

The original TurtleBot2 design assumes that the on-board computer is a netbook with its own battery system, charged independently from the robot battery. This gives an autonomy between four and nine hours depending on the selected battery (two models available), as shown in Figure 3.

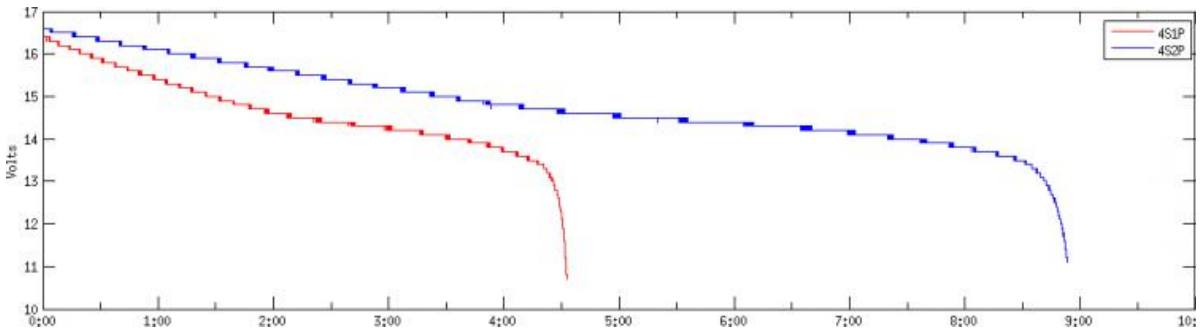

Figure 3: Default discharging profile of the Turtlebot2 robot, continuously spinning and with a Kinect camera

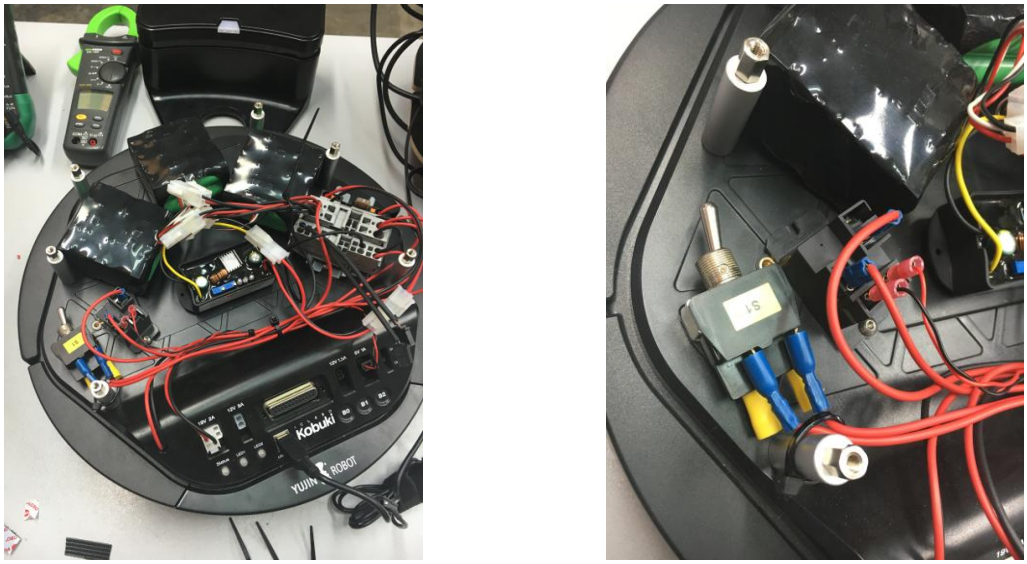

Figure 4: Installation of batteries and electronics for the new battery system

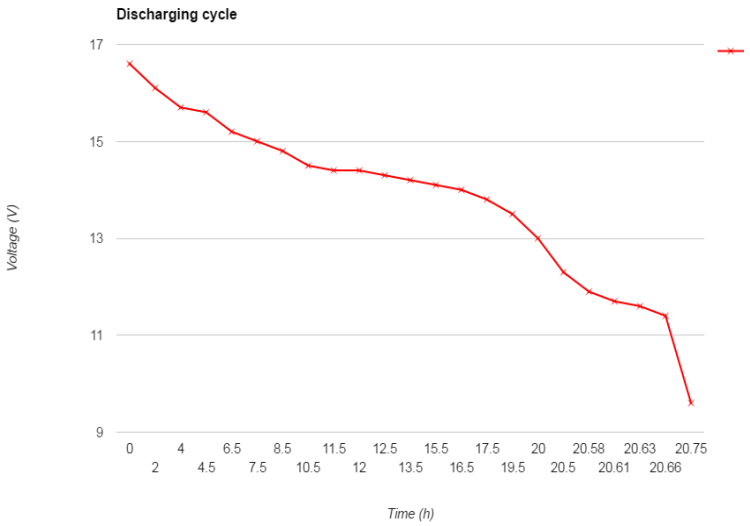

Figure 5: Discharging profile of the RADIO robot, continuously spinning with all the components and sensors working

Due to the increased number of sensors, devices and the inclusion of a PC controller plus the option of a Pico-Zed board, the default battery solution was not enough to keep the system working for an acceptable time.

The hardware was modified to use five batteries 4S2P (4400 mAh) instead of one (Figures 4 and 5).

This necessitated that the hardware of the docking station is modified, to be able to charge at higher power in order to shorten the charging time of the robot. With the new docking station and the new battery charger the system is now able to be charged up to 4A versus the 2A able by default (Figures 6, 7, and 8).

Figure 6: RADIO docking station after hardware modifications

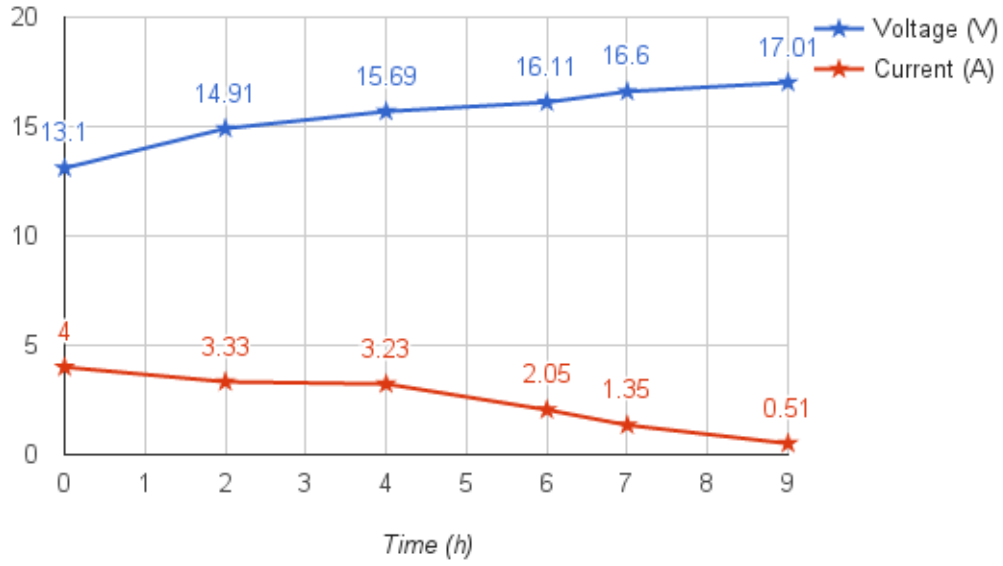

Figure 7: RADIO Robot charging profile

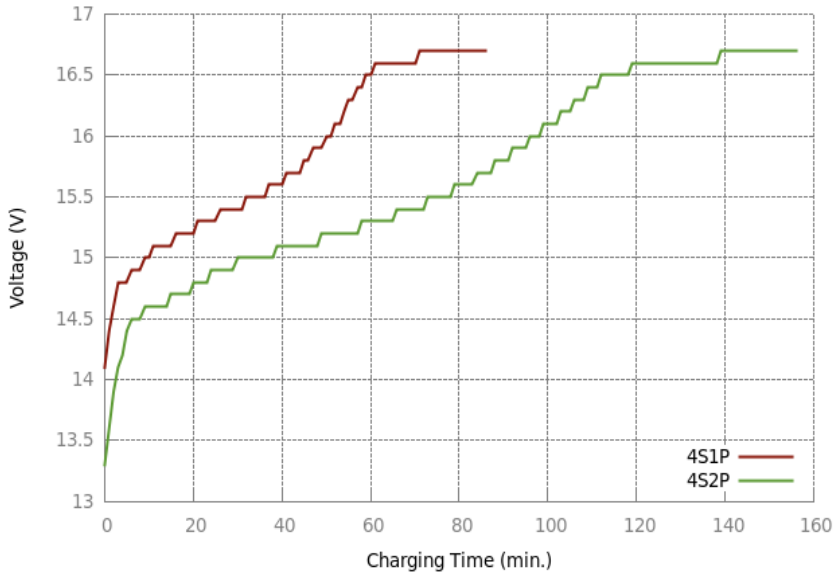

Figure 8: Default Turtlebot 2 charging profile

2.2.2 Telescopic bar

The Telescopic bar is the one supporting the RGBD camera that is mounted on the top. It permits 4 different positions for the camera:

- Position 1: 0.607 m
- Position 2: 0.867 m
- Position 3: 1.135 m
- Position 4: 1.384 m

2.2.3 RGBD camera

The Orbbec is a RGBD camera providing the 3D perception capability, needed to create 3D maps of the environment, collision avoidance and identifying the ADLs.

Table 2: ORBBEC Technical Specifications	
Power	5V (USB 2.0)
Range	0.4 - 8 m
Depth Image Size	640*480 (VGA) 16bit @ 30FPS
RGB Image Size	1280*960 @ 10FPS
Data Interface	USB 2.0
Microphones	2
Weight	300 g
Software	Orbbec Astra SDK + Openni (ROS compatible)

Figure 9: Telescopic bar and Orbbec camera

The bar is mounted on the middle and top plates of the Turtlebot and the following parts are needed for its assembly:

- Telescopic bar adapter
- Hexagonal Head M6 Screw
- M6 washer
- Middle plate with M6 hole
- Allen M6

Figure 9 shows the assembly of the telescopic bar and the cameras onto it.

2.2.4 Laser Scanner

The selected 2D laser scanner model is the Hokuyo URG-10LX. It is a high accuracy, high resolution and wide-angle sensor, which provides sufficient scan data to enable autonomous robots moving in an unknown environment. Thanks to this sensor, the robot can map, localize and navigate through the environment in a safe way. Table 3 gives the technical specifications.

The Hokuyo is mounted in the middle plate of the TurtleBot2, and the following parts are needed to assemble it:

- Metallic extension plate
- 3 M3 bolts
- 3 M3 nut
- 2 M2 countersunk head screw

Figure 10 shows the assembly of the Hokuyo laser sensor.

Table 3: Hokuyo Technical Specifications	
Supply voltage	DC 12V/DC 24V (operation range 10 to 30V ripple within 10%)
Supply current	150mA or less (during start up 450mA is necessary.
Light source	Laser semiconductor (905nm), Laser class 1(IEC60825-1:2007)
Accuracy	±40mm
Interface	Ethernet 100BASE-TX
Ambient temperature and humidity	-10°C to +50°C, below 85%RH (without dew, frost)
Storage temperature and humidity	-30°C to +70°C, below 85%RH (without dew, frost)
Vibration resistance	10 to 55Hz double amplitude of 1.5mm for 2hrs in each X, Y, and Z direction 55 to 200Hz 98m / s2 sweep of 2min for 1hr in each X,Y and Z direction
Weight	130g(Excluding cable)

Figure 10: Hokuyo mounted on the robot

2.2.5 On-board Computer

The selected device is a very powerful mini PC capable of providing extended processing capabilities. It is composed by an Intel Celeron N3050 Dual Core Processor (1.6 - 2.16GHz), 64-bit and Intel HD Graphics (320 - 600MHz). It supports up to 8GB of memory (DDR3L; 1.35V; 1333MHz Minimum) in 1 Slot and a 2.5" SATA Drive (6.0Gbps MAX.), including Intel 802.11ac wireless and Bluetooth 4.0.

This component runs most of the core software functionalities of the robot: base controllers, sensors controllers, localization and navigation processes.

Figure 11: Intel NUC

Figure 12: ASUS RT-N12 Router

Figure 13: Emergency stop button

2.2.6 WiFi Router

The selected device is the Asus RT-N12 Router, mounted in the middle plate of the robot (Figure 12). This router works at 2.4GHz and allows many encryption algorithms as AES, 128-bit WEP, 64-bit WEP, TKIP, WPA, WPA2, WPA-PSK, WPA2-PSK. This router is used to interconnect the internal components: pc, laser scanner (now with Ethernet interface) and the PicoZed. This router has the capabilities of bridging external networks.

2.2.7 Emergency stop button

The emergency stop button, once pressed, activates a signal to inform the central controller that the robot must stop. Although the robot is not dangerous for the people around it, there was a requirement made by the end users to disable the robot (in a hardware way) to not disturb during the work of the medical team.

2.2.8 Robot shell

Covers were designed and manufactured for the robot during the last stage of the project. This shell (see Figure 14) hides and protects most of the electronic components and sensors. This covers are made of plastic in order to (a) avoid interferences with the BLE transceiver needed for localization purposes, and (b) keep weight low, to improve battery autonomy.

Besides the protection of the internal parts of the robots, this cover also serves to improve the look of the system to the end users.

Figure 14: Left robot with shell - right robot without shell

Figure 15: The parts of the RADIO Robot shell

Figure 16: Broken robot cover being shipped to FPHAG

The development of this component had two design cycles based on the results of the first prototype.

First version: The parts named as “Tapa A”, “Tapa B”, “Soporte tapa A” and “Soporte tapa B” are made of black methacrylate with a thickness of 3 mm. This first design proved to be too fragile and a set of covers were broken during a transport to FPHAG.

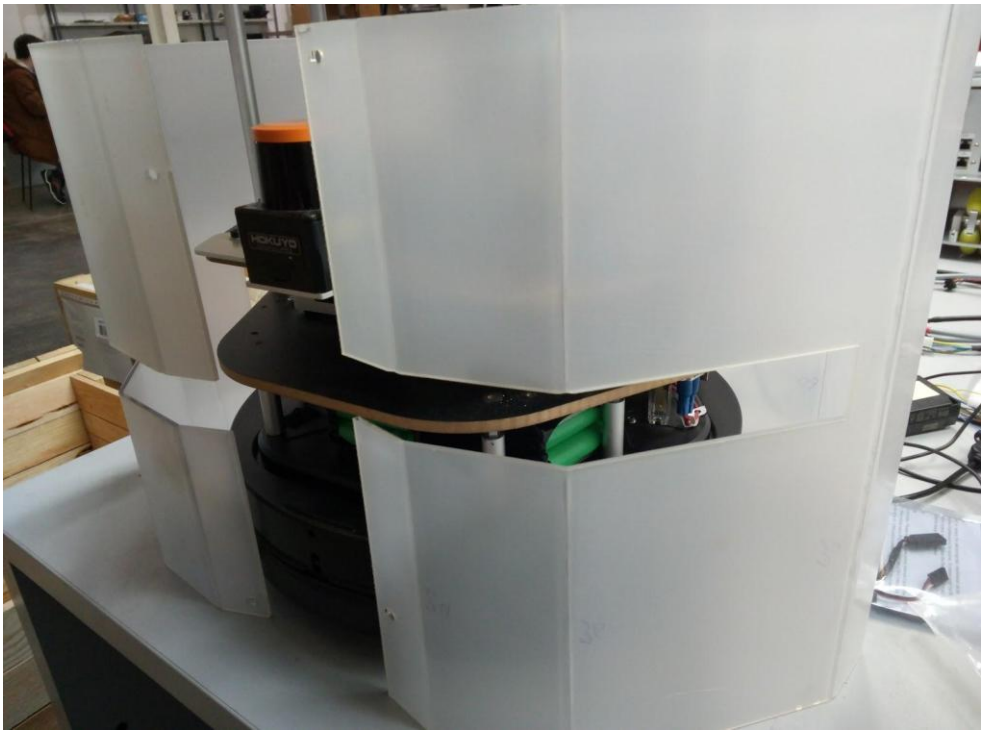

Figure 17: RADIO Robot transparent covers

Figure 18: Logitech gamepad

Second version: The parts “Tapa A” and “Tapa B” keep the same thickness but the material is changed to PETG, a transparent plastic sheet with good impact resistance and outstanding thermoforming characteristic. Due to the fact that the material is transparent, it is possible to use multiple colours for the cover and not only black. The parts “Soporte tapa A” and “Soporte tapa B” are still made of methacrylate but the thickness was increased to 6 mm to increase the resistance.

2.2.9 Gamepad controller

The last component is the gamepad. The robot is teleoperated by means of a Logitech gamepad. This device allows remote manual control of the robot, mainly for maintenance and setup operations.

3 PROTOTYPE SOFTWARE

All the developed software is integrated on the ROS middleware (www.ros.org).

The project has registered a Github organization (https://github.com/RADIO-PROJECT-EU) in order to gather and integrate all the Git repositories used for development. Repositories for new software developed for the RADIO project, are created in this organization's space. When adapting and extending existing software, either by one of the RADIO beneficiaries or by third parties, the original repository is forked into RADIO-PROJECT-EU and then the RADIO fork is used to track development within RADIO.

3.1 Core Packages

These are the core motor control, sensor drivers, localization, and navigation packages. Where possible, they were adapted from TurtleBot and Kobuki packages, with some packages specifically developed for the RADIO Robot. With respect to the first RADIO Robot (D4.6), the major changes are in that turtlebot_radio_bringup and turtlebot_apps repositories.

Repository turtlebot_radio_bringup has been updated to be consistent with changes in the ROS nodes it configures and launches. The most prominent change is that all ADL recognition nodes are now launched at start-up in paused mode, and activated when needed. This is more reactive than launching nodes when needed and shutting them down when not needed to preserve resources (cf. Section 8.2 "Task Switching" in D4.3 *Architecture for extending smart homes with robotic platform*).

In the turtlebot_apps repository, we address the problem observed with navigation experiments at FHAG: the robot would remember that a corridor was congested with people and refuse to navigate to a goal that required passing through that corridor even long after the congestion has cleared. To address this, we added the provision that if the goal cannot be reached, previously discovered obstacles that are currently not visible are removed from the costmap and the costmap is re-initialzed from the static map, forcing the robot to double-check if the obstacles persist. The robot will only give up after trying twice (cf. also Section 4.2 "Physical Demonstration" in this document and Section 8.1 "Navigation in Cluttered Spaces" in D4.3 *Architecture for extending smart homes with robotic platform*).

Package name and description	Source code repository and release used for this deliverable	
turtlebot: The basic drivers for running and using a TurtleBot with ROS. Forked from the official Turtlebot repository and modified and adapted for the RADIO Robot, creating a new model of the robot and sensors.	https://github.com/radio-project-eu/turtlebot	v1.1
turtlebot_apps: A group of demos and examples used as templates for TurtleBot/ROS packages. Forked from the official Turtlebot repository and modified and adapted, creating configuration files specific to RADIO localization and navigation.	https://github.com/radio-project-eu/turtlebot_apps	v2.0

Table continues on next page

Table continued from previous page

Package name and description	Source code repository and release used for this deliverable	
kobuki: Software controllers for the Kobuki mobile base. Forked from the official Turtlebot repository and slightly modified for the RADIO Robot.	https://github.com/radio-project-eu/kobuki	v1.0
kobuki_core: Software controllers for the Kobuki mobile base. Forked from the official Turtlebot repository and modified for the auto-docking procedure of the RADIO robot.	https://github.com/radio-project-eu/kobuki_core	v1.0
turtlebot_radio_bringup: Bringup files for the robot. It contains all the configuration and launch files to run all the RADIO Robot components.	https://github.com/radio-project-eu/turtlebot_radio_bringup	v1.2
robotnik_msgs: Definition of messages and services used by the core packages.	https://github.com/radio-project-eu/robotnik_msgs	v1.0
turtlebot_radio_emergency: Node that implements safety stop. It manages the emergency button of the robot and disables any movement.	https://github.com/radio-project-eu/turtlebot_radio_emergency	v1.0
marker_mapping: Package to localize ar_track_alvar markers and use them to initialize the localization module with a known pose every time the robot initializes. To avoid manual initialization, this node localizes visual QR markers onto the map and uses them inversely to calculate the correct pose of the robot.	https://github.com/radio-project-eu/marker_mapping	v1.0

3.2 Robot Perception Packages

The robot perception stack, delivered as Integrated Data Analysis System (D3.9/D.10) is also installed on the RADIO Robot. For completeness, the list of packages is repeated here. Note that some of the packages listed in D3.9/D.10 are omitted, as they are relevant to the integrated RADIO Home. These will be listed in D4.9 *Integrated smart home with robotic platform extensions*.

The most prominent change is that all ADL recognition nodes and their wrappers now implement the suspend service used by the overall system to suspend nodes when not needed to preserve resources and quickly re-activate them when needed (cf. Section 8.2 "Task Switching" in D4.3 *Architecture for extending smart homes with robotic platform*). Other changes include using compressed topics for ROSVisual, an update in motion analysis to handle the poor lighting conditions observed in some of the FHAG sessions, and other minor bug fixes in the reports generated by the wrappers.

Description	Source code repository and release used for this deliverable	
HumanPatterRecognition: Recognizes human walking patterns in laser scans and tracks walking.	https://github.com/radio-project-eu/HumanPatternRecognition	v3.0.0
HPR Wrapper: Uses HPR output to recognize and time "walked 4m" events.	https://github.com/radio-project-eu/hpr_wrapper	v2.0
ROSVisual: Tracks moving objects in the RGB/depth modality and classifies motion as bed or chair transfer.	https://github.com/radio-project-eu/ros_visual	v2.0
ROSVisual Wrapper: Uses the output from ROSVisual to time chair and bed transfer events and to recognize and time "walked 4m" events.	https://github.com/radio-project-eu/ros_visual_wrapper	v2.0
Motion Analysis: Recognizes motion and classifies it as "bed transfer" and "pill intake" events.	https://github.com/radio-project-eu/motion_analysis	v2.0
Motion Analysis Wrapper: Uses the output from motion analysis to time the bed transfer event.	https://github.com/radio-project-eu/motion_analysis_wrapper	v2.0
AUROS: Recognizes talking, watching TV, listening to music, and doing housework by acoustic analysis of the audio modality.	https://github.com/radio-project-eu/AUROS	v1.0
Map convergence: Consumes robot pose messages from the localization package and converts them to the corresponding coordinates in the RADIO Home model.	https://github.com/radio-project-eu/map_convergence	v1.0

3.3 Distributed RADIO Robot

This second RADIO Robot prototype includes two completely new components, which address engineering issues that stem from the distributed nature of the RADIO system. These components are used to guide the architecture design so that ROS nodes are distributes between the various computation nodes available to the system (both on-board and off-board) and to have the ROS master processes monitor network connectivity with each other (cf. Section 6.2 "Designing a System of Distributed ROS Nodes" in D4.3 *Architecture for extending smart homes with robotic platform*).

Description	Source code repository and release used for this deliverable	
rostune: Helps ROS developers distribute their nodes in the most effective way. It collects statistics for topics and nodes, such as CPU and network usage.	https://github.com/radio-project-eu/rostune	v1.0.7
connectivity checker: Checks the health of the ROSCore processes foreseen by the architecture of a multi-master ROS system (here, on-board NUC, RasberryPi, and FPGA on-board ARM)	https://github.com/radio-project-eu/radio_ros_connectivity_checker	v1.0

3.4 Robot Behaviour and RADIO Home Integration

Although some of them execute on the robot's on-board computer, the packages that implement robot behaviour, control by the end user, and interactions with the RADIO Home Main Controller are listed in D4.8 *Integrated smart home with robotic platform extensions*, as they relate to integrating the robot into the RADIO Home environment.

4 DEMONSTRATION

4.1 Simulated Demonstration

In order to test the localization and navigation capabilities of the RADIO robot in preparation of the actual trials at FZ, a simulated environment has been created for the Gazebo simulator,[1] resembling one of the private residences used in the FZ pilots. The environment is at:

https://github.com/RADIO-PROJECT-EU/roboskel_gazebo_resources

The simulated RADIO Robot for using with environment is at:

https://github.com/RADIO-PROJECT-EU/turtlebot_simulator

This package contains the configurations and definitions needed to have a virtual RADIO robot in a Gazebo simulation. The robot model is based on the standard TurtleBot2 model, but has been significantly adapted to account for the modifications carried out in RADIO.

A simulated demonstration of the robot's navigation capabilities in this environment can be seen here:

https://vimeo.com/251644417

4.2 Physical Demonstration

The final prototype of the RADIO Home (including the robot) was deployed at FHAG premises and at multiple private homes.

A physical demonstration of the robot's navigation capabilities can be seen here:

https://vimeo.com/251605364

The video demonstrates costmap re-initialization (Section 3.1), addressing the problem observed with navigation experiments during one of the RADIO project's pilots: the robot would remember that a corridor was congested with people and refuse to navigate to a goal that required passing through that corridor even long after the congestion has cleared.

In the video, the RADIO Robot first tries to get past the static obstacle in the middle from the left, but finds the path blocked by another robot and re-plans to get past the static obstacle from the other side of the static obstacle that is free, based on the static map information it possesses at that stage (00:12). Once the RADIO Robot that that route is also blocked, it clears the dynamically created obstacles and re-discovers the original plan (00:34), which fails again as the other robot has not moved (00:52). It clears the dynamic obstacles map for a second (and last for this goal) time, and successfully navigates to the goal as the person blocking the second route has now left.

[1] Cf. http://gazebosim.org