AUTONOMOUS WHEELED ROBOT PLATFORM TESTBED FOR NAVIGATION AND MAPPING USING LOW-COST SENSORS

D. Calero [1], E. Fernandez [1], M.E. Parés [1]

[1] Centre Tecnològic de Telecomunicacions de Catalunya (CTTC/CERCA), Av. Carl Friedrich Gauss, 7. Building B4, 08860 Castelldefels, Spain - (dcalero, efernandez, epares)@cttc.es

Commission II

KEY WORDS: 3D modelling, Kinect, Navigation, ROS, SLAM

ABSTRACT:

This paper presents the concept of an architecture for a wheeled robot system that helps researchers in the field of geomatics to speed up their daily research on kinematic geodesy, indoor navigation and indoor positioning fields. The presented ideas corresponds to an extensible and modular hardware and software system aimed at the development of new low-cost mapping algorithms as well as at the evaluation of the performance of sensors. The concept, already implemented in the CTTC's system ARAS (Autonomous Rover for Automatic Surveying) is generic and extensible. This means that it is possible to incorporate new navigation algorithms or sensors at no maintenance cost. Only the effort related to the development tasks required to either create such algorithms needs to be taken into account. As a consequence, change poses a much small problem for research activities in this specific area. This system includes several standalone sensors that may be combined in different ways to accomplish several goals; that is, this system may be used to perform a variety of tasks, as, for instance evaluates positioning algorithms performance or mapping algorithms performance.

1. INTRODUCTION

Geomatics community is constantly developing new positioning and mapping algorithms. Those algorithms aim at determine precise and accurate maps in a faster and cheaper way. By map we mean both 3D point clouds or georeferenced images. Moreover, the community may take advantage of new mobile platforms available in the market (i.e. small drones like DJI Phantom 4) plus new software and tools able to generate the ortophotos or 3D point clouds easily (i.e. Pix4D and ArcGIS). These tools are speeding up the development of new applications in fields such as civil engineering and precision agriculture, for example. Furthermore, they rely fully on a good imaging sensor positioning and orientation, or equivalently on a good platform trajectory determination. In the context of this paper, a trajectory might be a time series of positions, velocities and attitudes of a moving object plus the calibration parameters of the instruments used to determine those and the estimated covariances of all those values.

The mapping research community is currently dealing with two challenges: the use of new technologies and their use in new harsh scenarios. New sensors, able to provide data suitable for geo-applications, appear constantly in the market. As stated in (Groves et al., 2014a; Groves et al., 2014b), technology has to deal with new sensors –like plenoptic or photon-mixing cameras–, new performances –like the inertial sensors found in smartphones– and new environments –like indoor or urban canyons–. Furthermore, there are still many issues to solve concerning the achievement of target precision, accuracy and reliability in the realm of mapping. The following are some examples: when available, GNSS accuracy is almost unachievable due to strong multipath in narrow urban environments (Xie and Petovello, 2015); shadows and poor illumination conditions are a threat for tie points extraction and matching (Fang and Zhang, 2015).

The examples above define a scenario where continuous and intense research in a steadily changing technological environment is taking place. This constant, uninterrupted change and evolution process constitutes a challenge (for instance, from the software and hardware engineering standpoint) for the research activities in this area.

Currently, a researcher can find a wide range of helpful tools available both in the "traditional" market as in the open-source / free license market. From the point-of-view of hardware, the combination of physical interfaces and protocols available in the market may slow down the integration of new hardware pieces on any system. The industry is facing this diversity with the creation and adoption of common interfaces for products that may have a similar use, but at the same time new ones are being created.

From the point-of-view of software, most platforms run with a specific operative system (OS). This limits their use on other platforms that may run with a different OS. This is the case of GNU-Linux, Windows, iOS and Android based platforms (using their own libraries). GNU-Linux core system may be the most common one in robotics due their easiness, flexibility and open source code that allow any user to know and check what a code is doing at any moment. Other solutions, closed from the point of view of a developer, like Windows or iOS, may offer better assistance and support at the early stages of product use, but they are less flexible to support new developments. To help to standardize the software developments on the robotics fields, a Robotic Operative System (ROS) was developed some years ago and it is keep in continuous development by their own community. ROS core is based on a GNU-Linux OS (Quigley et al., 2008).

This paper presents the concept of an architecture for a system whose target is to provide a reliable framework where research related to low-cost 3D mapping may take place. The goal of such system is to become the basic toolset for researchers in these areas, avoiding the need to start anew each time a new project starts. Additionally, an implementation of the aforementioned concept and architecture is presented here: ARAS.

ARAS consists on a wheeled robot platform equipped with several low-cost navigation sensors, such as a GNSS receiver, a low cost IMU, rotary encoders, and a RGBD Kinect v2 depth sensor. To increase its operation capabilities it has a wireless communications interface and a wired interface. The testbed also includes an indoor scenario inside the CTTC building for executing real tests.

Section 2 review the main concepts related to autonomous terrestrial mapping, section 3 describes the architecture (and components) from a conceptual and an implementation standpoint (showing how the ARAS system materializes the concept). Section 4 details how this system may be used in different scenarios.

2. STATE OF THE ART

2.1 Low-cost vehicles

Among the different robots types possible, there is a group of robots called "autonomous" which must be able to take decisions based on previously defined criteria without the intervention of a human operator (Siegwart et al., 2011). In the particular case of mobile robots and for the purpose of this research, it is defined an autonomous rover as a robot capable of navigating throughout an environment without the intervention of a human operator. These robots are already preferred over humans in some areas, in example on hazardous environments or risky situations. Mobile robots can fit in five categories from the platform point of view: legged, wheeled, flying or swimming robots.

Wheeled robots are defined as robotic platforms that navigate around the ground using motorized wheels to propel themselves. The main advantages they offer are that are easy to implement in mechanical terms and that are very stable in most environments, compared to other mobile platforms such as legged robots. The main drawback of wheeled robots is that they are not able to perform a good navigation over some surfaces, such as rocky terrain, sharp declines, or areas with low friction (Ghotbi et al., 2016).

In terms of application, autonomous wheeled robots are one of the most used platforms in nowadays engineering: For instance, domestic robots such as *Roomba* vacuum cleaner, rovers used in space applications such as the Mars Curiosity (Tunstel et al., 2005), robots in Amazon automated warehouses, multipurpose robots designed by Clearpath robotics or the Google self-driving car (Jiang et al., 2015) now under the Waymo project.

2.2 Positioning and navigation

For this research, navigation is defined as the control of robots motors attached to the wheels. The navigation module determines the power to be applied to each motor to reach the specified waypoint from its actual localization. Sometimes, it can work without waypoints information, in a free-velocity mode, for example, when a Radio Control (RC) is used.

A path-planning module may assist the navigation module providing it with the next waypoint to be reached. It can we operated on a fixed waypoints mode or on a most advanced dynamic behaviour, in which new waypoints are created during the robot operation (either by the user or any other module or algorithm). Nevertheless, this module is only mandatory if waypoint navigation is expected and thus, could be avoided if desired by the user.

Furthermore, to develop an autonomous robot, which needs of a navigation module, is mandatory to provide to the system some kind of localization service, understood as the position coordinates and attitude at a specific time. Localization, or positioning, consists on determining where the robot is in the environment, either in local or global coordinate's frames. In autonomous rovers, localization consists on determining a set of TPVA (Time-Position-Velocity-Attitude), at least for the vehicle chassis or main frame (there might be other mobile parts in the robot, such as a robotic arm). Hence, TPVA describes the robot's main frame position, velocity and attitude at a particular instant of time (Parés and Colomina, 2015).

2.3 Low-cost localization sensors

Hence, to compute and obtain a TPVA solution a set of sensors is needed. The selection of the sensors used depends on the environment in which the robot will operate (i.e. indoor or outdoor scenarios).

Among the possible sensors studied, localization sensors can be classified as inertial and non-inertial sensors (or visual, etc…). Inertial sensors are instruments that measure rotation and translation forces observed by the object. These sensors are usually used to perform dead reckoning, which is a method of localization that relies on estimating the position, speed and orientation of the robot based on earlier known positions. The most used inertial sensors in autonomous rovers are gyroscopes and accelerometers.

Gyroscopes are used to measure rotational forces. In autonomous rovers they are used mainly to determine the heading of the vehicle (Titterton and Weston, 2004). Gyroscopes measure reactive torque that is produced due to the movement of the sensor to give absolute orientation respect their spin axis. Torque (τ) is proportional to the spinning speed (ω), the precession speed (Ω), and the wheel's inertia (I).

$$\tau = I\omega\Omega \qquad (\text{eq.1})$$

The main drawback of the gyroscopes used for dead reckoning is that the typical error of the system is accumulative. This means that if they are used for a long time, a small constant drift will provide a big error.

An accelerometer is a device that measures static (gravitational) and dynamic acceleration forces (Titterton and Weston, 2004). Given the mass (m) of the accelerometer, the measure of the dynamic forces (F) provides the acceleration (a) of the sensor in every axis:

$$a = F/m \qquad (\text{eq.2})$$

With the acceleration, the speed and position are easy to obtain with integral calculation.

However, all inertial sensors suffer of stochastic errors such as biases and random walks effects among others that limit the system performance when operating only with this kind of sensors.

An IMU (Inertial Measurement Unit) is an embedded sensor that usually combines 3-axis accelerometer and 3-axis gyroscope orthogonally placed between them. In some cases, IMUs also carry magnetometers. IMUs are ones of the most used sensors in robotics because of their capability of measuring translation and rotation (Titterton and Weston, 2004).

Non-inertial sensors do not measure directly the forces on the object. There are plenty of sensors and technologies that can be used for localization (cameras, barometers, Wi-Fi, Ultra Wide Band, etc.), but the most used on autonomous rovers are magnetometers, rotatory encoders and GNSS (Global Navigation Satellite System). In addition, RGB-D or LIDAR sensors can be used for localization and mapping using techniques such as SLAM (Simultaneous Localization And Mapping) (Davison, 2007).

Magnetometers determine the direction of Earth's magnetic north, and there are two types: Hall Effect and flux gate compasses (Titterton and Weston, 2004). Regardless of the type of compass used, a major drawback concerning the use of the Earth's magnetic field for mobile robot applications involves disturbance of that magnetic field by other magnetic objects. This issue is greater for indoors robots, since most buildings have metallic structures that can disturb the magnetic field inside. Moreover, they are very noisy sensors that require of relatively high time integration values to obtain an accurate measurement. In return, they provide a global measurement, very useful when combined with other sensors.

Rotatory encoders track the angular position of any rotatory device to generate digital information. In rovers, are used to determine odometry (change of position over time) by monitoring the number of turns made by the wheels. If an encoder is able of determining the direction of the rotation, it is called quadrature encoder. There are many types of encoder, like magnetic or mechanical, but one of the most used in robotics is the optical encoder. This may include any visual odometry source, a kind of odometry measurements extracted from visual imagery processing (McCarthy and Barnes, 2004).

Radiofrequency systems such as GNSS mainly, but also NFC, RFID, UWB and others, very useful in GNSS-denied scenarios such as indoor buildings (Montañés et al., 2013) and (Navarro and Nájar, 2011).

The GNSS is a beacon based localization (Titterton and Weston, 2004). There are at least twenty-four operational GNSS satellites at all times. Each satellite continuously transmits data that indicate its location and the current time. The GNSS satellites synchronize their transmissions so that their signals are sent at the same time. When a GNSS receiver receives the RF (Radio-Frequency) signal of a satellite, the arrival time is measured and used to determine the relative distance to this satellite, usually known as pseudorange. By combining four or more pseudoranges, the position of the receiver is determined by triangulation means. The main advantages of this system are low price of GNSS receivers and the fact that they do not have accumulative error. In the other hand, GNSS receivers cannot be used as a reliable localization sensors on robots that have to work in low coverture areas, such a dense forest or indoors. This limits its use to a limited number of scenarios where an open view of the sky is available and the RF thermal noise is on normal levels.

Acoustic systems, as the one developed by the LOPSI group from the Spanish CSIC (Jiménez et al., 2009).

There are available some commercial indoor localization service providers such as mapspeople, deepmap and infsorf, among others.

2.4 Low-cost perception/imaging sensors

Depth perception consists on determining the distance between the robot and the surrounding objects than conform the environment. This is done by taking measurements using one or a set of sensors, and then extracting meaningful information from those measurements. There are a lot of types of depth perception sensors, and most of them imitate human or animal senses. For instance, an ultrasonic sensor imitates bats echolocation (Fiorillo, 1999). This type of sensors provides a direct measurement of distance to the surrounding objects interacting with the environment. There are different technologies used by these sensors, but the most used in robotics is the time of flight or arrival (ToF or ToA).

Optical sensors are designed following two main architectures, either CMOS or CCD cameras. CMOS are cheaper than CCDs for the same amount of pixels integrated, therefore being of the preferred technology in low cost devices.

New developments and cost reduction of products have led to the rising of stereo cams, with a pair of image sensors instead of traditional mono cameras with a single sensor, such as Kinect and Zed cameras among others available in the market. They allow the automatic generation of cloud points with their dual camera integration technology with and extreme easiness of use.

2.5 SLAM

SLAM is a technique that involves the use of perception sensors, such as RGB-D sensors, LIDARs, cameras and sometimes inertial sensors. It consists on trying to simultaneously localize the sensor with respect to its surroundings, while at the same time mapping the structure of that environment (Davison, 2007). The sensors are obtaining depth measurements from the environment and comparing the obtained pattern with a map of the surroundings. The map can be introduced by the user or generated and actualized during the operation of the robot. If the detected pattern coincides with a pattern of the environment, the robot is able to get a localization. At the same time, the environment is being mapped. It is a good localization technique, but requires a post process period that is time consuming (Zienkiewicz et al., 2016).

3. SYSTEM ARCHITECTURE

3.1 Requirements

The architecture (and components) of a framework willing to provide with a useful set of tools and procedures to facilitate the

research tasks related to a specific set of disciplines may obviously vary depending on the actual experience of the people involved first in its inception and, later on, in its design and implementation stages. This paper presents, therefore, the view of its author, view that has been heavily influenced by their participation in a series of projects where the framework discussed here played a key role.

The components included in a low-cost mapping research framework should be able to offer, at least, the following set of features:

- Data acquisition from sensors,
- Localization (TPVA) of robot chassis,
- Autonomous Navigation including path planning (waypoints),
- 3D Mapping capability

Each of the features in the list above is important by itself. For instance, the ability to collect data from a stereocam is a key factor for the 3D mapping feature. In short, these tools help to save time and reduce costs, thus facilitating the work of researchers. However, it is the combination of different subsets of these features (tools) what reveals the versatility of the concept, and how it responds to different research use cases (see section 4).

ARAS implements these features, providing a module for each feature (subsection 3.3.3).

3.2 Architecture and components

The system is designed to work as is or in reduced versions using less logic units and sensors, and thus simplifying its scheme. In its full-equip version it has three main logic units:

- ODROID XU4, a microprocessor as the ROS core unit.
- Arduino Due, a microcontroller that acquires data from most of the sensors integrated on the rover.
- Motor Controller, another microcontroller for power and PWM control of the four DC motors.

For example, the microprocessor and a microcontroller (Arduino Due) can be removed. This reduces the system to an R/C vehicle. Instead, having both microcontrollers and a microprocessor unit, allow to connect a huge variety of sensors, and in most cases to any of them (for example, the IMU). If no RGBD sensor is used, then the Microprocessor can be removed from the system reducing considerable its power consumption by more than 45W just in processing power.
Calibration parameters are measured and provided to the system for all the sensors integrated in the robot, but it is a decision of the user to use or modify them. The use of calibration parameters improves the output quality performance on all levels.

Future integration of new sensors will be eased by the adoption of ROS. Hence, many sensors are already providing ROS drivers or software interfaces for this purpose but in the case that they are not available, the integrator just must work to publish or subscribe to the adequate ROS topic. Moreover, an integrator may benefit from a new hardware ROS proposed standard: HROS {}.

The development of new algorithms will be also eased by the adoption of ROS. A developer must only have to take into account which ROS topics must subscribe or publish to fit its new development into the already existing ecosystem.

3.3 Actual implementation

3.3.1 ARAS platform

A *Kit 4WD1 Aluminium (Lynxmotion)* is the basis frame used to develop this Rover. As mentioned before, it includes:

- An Aluminium support frame (A4WD1E-KT). It has also some plastic pieces and covers.
- Four DC brushed motors. Model GHM-04 @ 12V. Each one with a quadrature encoder attached to their motor shaft.
- Four 4.75" tires made of plastic.

The set of sensors integrated are:

- GPS+SBAS Skytraq RAW-1315F. It provides both GPS coordinates and pseudorange measurements.
- IMU Invensense MPU9250, triaxial gyroscopes, accelerometers and magnetometers (9 DoF). It has also a temperature probe for temperature compensation.
- Four quadrature rotary encoders with up to 12000 counts by wheel revolution (motors do 400 counts by revolution but also have a 30:1 gear head). They are attached to the motor shaft.
- A Microsoft Kinect V2 RGB-D depth sensor. It includes a dual camera system (RGB and IR) plus an IR point-grid emitter.

This rover is able to provide a continuous TPVA solution, both in local or global coordinates and regardless of indoor or outdoor scenarios. Hence, the TPVA solution will take into consideration only the available sensor's data at the moment of the processing.

Figure 1. ARAS rover

The set of processors or main logic units are:

- Roboclaw has an ATMEL IC, can decode dual quadrature encoder's data and has two PWM output channels for brushed DC motors. 5V and 3.3V tolerant.
- Arduino Due has an ATMEL SAM3X8E ARM Cortex-M3 @ 84 MHz. Only 3.3V tolerant except for their USB ports that are 5V tolerant.
- ODROID XU4 has a Samsung Exynos5422 Cortex™-A15 2 GHz and Cortex™-A7 Octa core CPUs. Mali-T628 MP6 (OpenGL ES 3.1/2.0/1.1 and OpenCL 1.2 Full profile). 2 GB

RAM memory. Two USB 3.0 Host, one USB 2.0 Host and a Gigabit Ethernet port. We have included an USB hub connected to the 2.0 USB port with a Wi-Fi USB 802.11 bgn compliant.

The approximate costs for the platform at the moment of this writing are summarized in Table 1:

Component	Price (€)
Kinect 2	90
Kinect cable adapter	50
Odroid XU4 + eMMC + adapters	100
Arduino Due	30
Power adaptation circuit	100
Rover Chasis Kit	320
LiPo Battery 4S1P 3000mAh	30
TOTAL	**720 €**

Table 1. Platform prices list

Dependencies: RoboClaw firmware has been updated to the latest version available at this moment. Arduino's library was downloaded prior to its use.

Arduino Due requires of the following software and libraries: Arduino IDE 1.6 or higher; Arduino DUE programming library; MPU9250 library; RoboClaw library; Rosserial library; Skytraq GNSS library.

ODROID XU4 requires of the following software and libraries: LUBUNTU 14.04; ROS Indigo: Rosserial ROS module; Kinect2_bridge ROS module; IAI_kinect ROS module; RTAB-MAP ROS module (http://introlab.github.io/rtabmap/); OPENCV2.4; OPENCL; GCC4.6.

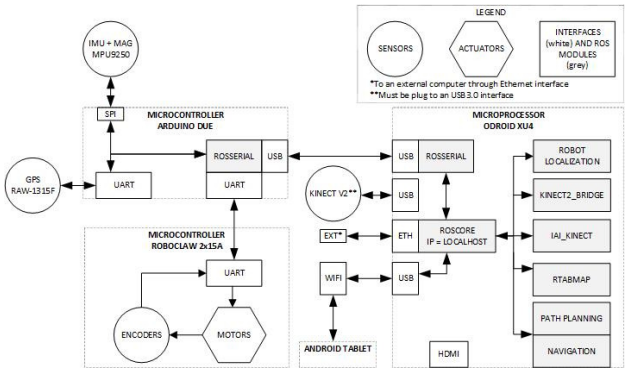

Figure 2. Schematic diagram

3.3.2 ARAS ROS data acquisition modules

ROS topics are the way ROS operates and they allow the different modules to interchange information. This is the list of ROS topics, either published or subscribed, from sensor's data processing:

ODOM: Odometer information from encoders. It may include position coordinates (X, Y, Z), quaternions orientation (QX, QY, QZ and QW), covariance matrixes and twist information: linear (VX, VY and VZ) and angular (WX, WY and WZ) with its respective covariance matrixes.

IMU: IMU information MPU9250. It may include quaternion orientation (QX, QY, QZ and QW), angular velocities (WX, WY and WZ) and linear acceleration (VX, VY and VZ) and their respective covariance matrixes.

ODOM_FILTERED: KF output from ROBOT_LOCALIZATION ROS module. Similar to ODOM.

TF: Transformation topic (lever-arms). It may include translation (TFX, TFY and TFZ) and rotation (TFQX, TFQY, TFQZ and TFQW) transforms.

CMD_VELOCITY: Command velocity. It may include linear (VX, VY and VZ) and angular (WX, WY and WZ) velocities.

KINECT is a particular case in which many topics are published at the same time form a single source. The lists of the most relevant topics related to Kinect are: three different qualities (HD, quarter HD and SD), colour or mono images, and depth or cloud images. Compressed images can be obtained on demand.

All the topics (except for the Kinect related ones) are listed in Table 2 with their appropriate ROS type definition.

Topic	Type
ODOM	nav_msgs/Odometry
IMU	sensor_msgs/Imu
ODOM_FILTERED	nav_msgs/Odometry.
TF	tf
CMD_VELOCITY	cmd_vel

Table 2. Topics and types list

All Topics include a time-tag field.

3.3.3 ARAS ROS data processing modules

Main system operates with ROS INDIGO running on LUBUNTU 14.04. The different ROS modules executed may publish or subscribe to different ROS topics as mentioned before.

ROSCORE is always executed first and is setup to its localhost IP address.

ROSSERIAL is installed and configured to allow serial intercommunication between the Arduino Due (USB Programming port interface) and ODROID XU4 (USB). It is configured at 57600 bps speed. Arduino Due subscribes to CMD_VELOCITY and publishes ODOM and IMU ROS topics (ODROID opposed subscription/publication).

ROBOT_LOCALIZATION is executed next. It subscribes to ODOM and IMU and publishes ODOM_FILTERED ROS topics. This can be changed by CTTC's NAVEGA module.

ROSKINECT2_BRIDGE is launched once the USB connectivity is validated (executing a terminal *lsusb* command must make appear a list with three Microsoft devices, either is not working properly). It specifies OPENCL processing and it uses IAI_KINECT.

RTABMAP is last executed. It subscribes to Kinect images (2D photos or 3D cloud points), TF (constant lever arm between Kinect's RGB and IR cameras) and ODOM_FILTERED (the

topics published by this module are not detailed on this research). This module allows SLAM execution. RTAB-Map (Real-Time Appearance-Based Mapping) is a RGB-D Graph SLAM approach based on a global Bayesian loop closure detector. A memory management approach is used to limit the number of locations used for loop closure detection and graph optimization, so that real-time constraints on large-scale environments are always respected. RTAB-Map can be used alone with a hand-held Kinect or stereo camera for 6DoF RGB-D mapping, or on a robot equipped with a laser rangefinder for 3DoF mapping. (RTAB-Map)

Acquisition module
Conceptually, there is module responsible of the gathering the information coming from the sensors. This module is executed both in the Arduino Due and ODROID XU4, acquiring and communicating with Odometers, GPS and IMU (Arduino Due) and Kinect (Odroid XU4). Data is pre-processed (units and axis conversion). This module is interfaced with ROSSERIAL module.

Robot Localization module
The coordinates origin are in the platform centre, X-axis forward, Y-axis right and Z-axis down. The IMU is placed in the rover mass and rotation centre, with its own axis orientation adapted to rover axis orientation. Lever-arm are measured and provided to the system through convenient topics publications.

Path planning and navigation module
This module can work either in manual or automatic configurations. It is executed on all logic IC: RoboClaw, Arduino Due and ODROID XU4. Motor control commands travel through ROSSERIAL ROS module.

At its most basic configuration, the path planning and navigation module work as a manual control device through an Android app running on a tablet. This application publishes CMD_VELOCITY topic messages and requires of human interaction controlling a graphical joystick.

At its most advanced configuration, the Path planning and navigation module is working with WAYPOINTS. In this particular case, it computes the CMD_VELOCITY to be published from the actual rover localization output (ODOM_FILTERED) and the next WAYPOINT to be reached. Waypoint can stored in memory previously, automatically created during execution and/or created using the previously mentioned Android app.

Anyway, RoboClaw accepts three types of motor control serial messages:
- Velocity: Just a velocity value.
- Velocity and acceleration: A specific velocity and acceleration values.
- Velocity, acceleration and distance: The same as above but until reach and specific distance (encoder quadrature counts).

Although these three types of messages can be send to RoboClaw, by now and to simplify the system architecture and be fully compatible with ROS commands, only velocity serial messages from Arduino Due are send to RoboClaw. This module can be changed by the ROS NAVIGATION module but initial tests were not working as expected and then it was decided to not use it by now until more work is done with this module.

Depth map and Point cloud module
The generation point clouds are automatically done by the Kinect itself together with the KINECT_BRIDGE ROS module. Hence, any user may subscribe to the published cloud points to obtain them.

Building map module
This module is directly related to RTABMAP ROS module which is responsible of the creation of a 3D map based on the point cloud received, plus the TF of the cameras and rover localization (ODOM_FILTERED ROS topic).

Obstacle avoidance module
This has been the first ROS module created and tested for this platform and it requires of the obstacle detection module.

The objective is to surround any obstacle, like in bug algorithm, but instead of following the border of the object the algorithm tries to find where the obstacle ends, similarly as in the "follow the gap" philosophy, to set a new waypoint beside it. To do so, the robot turns until the obstacle is no longer detected. Next figure explains graphically the ARAS obstacle avoidance algorithm.

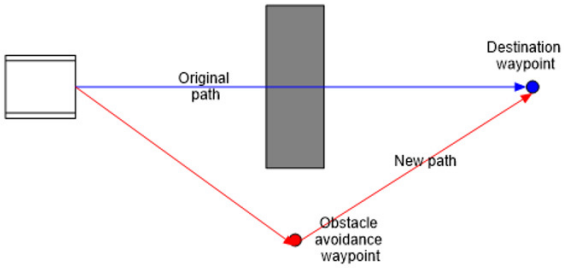

Figure 3. Obstacle avoidance algorithm schematic

The **obstacle avoidance** module uses Kinect depth images to determine if there is an obstacle in rover's straight line path. Obstacles situated beyond the Kinect central point at the X axis (15 cm) are considered obstacles at left and obstacles above this point are considered to be at right.

The **obstacle avoidance** module is executed on the microprocessor (ODROID XU4). The general operation works as follows: this module is only enabled when an obstacle has been detected. Its main task is to compute a new waypoint that guides the robot into avoiding the obstacle. To do so, the robot performs a scan of the environment and searches obstacle free areas. When a suitable area is met, the algorithm creates a new point on the route. The navigation module reads this as the new waypoint.

Therefore, this module subscribes to Kinect depth images and publishes either velocity commands (CMD_VELOCITY ROS topic) or waypoints (advanced mode).

4. USE CASES

A tool such the one presented in this paper may be used in a variety of research use cases, just combining in different ways the components that integrate it. The most usual use cases are:

- Verification and validation of localization and mapping new sensors and algorithms
- Positioning and Mapping in Real-life Environments

It has been stated several times in this paper that ARAS is generic and extensible. This means that new sensors may be included into the system---or those that have been modified easily adapted-- at almost no cost.

Basically, including a new algorithm means just to determine which ROS topics must either subscribe or publish. There is no need to change already existing software to make the new data available to ARAS; that is, when adding new sensors there is no need to maintain, change or adapt the existing code base.

As it happens to any kind of algorithm, it is necessary to verify and validate new models in order to guarantee their correctness (that is, the new code contains no bugs) and performance (the new sensor is correctly modelled). This is the algorithm verification and validation use case.

4.1 Verification and validation of new mapping technology

It has been stated several times in this paper that GEMMA is generic and extensible. This means that new sensors may be included into the system—or, those that have been modified, easily adapted—at almost no cost.

Testing new algorithms is such simple as to substitute current ROS modules with the new ones, to carry on predefined missions and to check for the performance of the results.

It has been created a 3D map of one of our laboratory to validate the RTABMAP output as seen in Figure 4. It has been rendered using CloudCompare software.

4.2 Positioning and Mapping in Real-life Environments

Loaded with the appropriate set of modules ARAS might be used as a real-time server providing successive position and attitude values to some other subsystem(s). For instance a RF mapping payload might decide to delegate the task of positioning the vehicle platform to ARAS, feeding this component with data coming from the different sensors on board, instead of performing this process itself. ARAS is now providing a position solution with 5 cm. precision and 10 cm accuracy after 5 minutes of mission.

A variation of the example above would compute mapping solution in real-time but log these data to permanent storage instead. This capability is of special relevance in emergency scenarios where quick charting is required (Angelats and Navarro, 2017). ARAS would collect raw data of positioning and perception sensors and would process it on-board. Thanks to its obstacle and avoidance module it is able to autonomously map an entire floor. Once the acquisition mission finish, the user can download the processed point cloud and start its processing. ARAS is now providing 3D point cloud with 10 cm precision and 20 cm, accuracy after 5 minutes of mission.

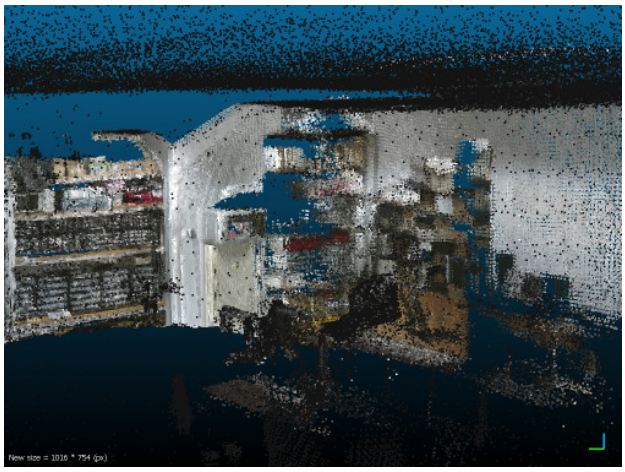

Figure 4. RTABMAP model of our laboratory

5. CONCLUSIONS

ARAS is not only a concept but also a real, seasoned, research-oriented system, aiming to pave the way to make possible a significant number of research projects, covering a wide spectrum of situations within the realm of geodesy, positioning, and mapping. This has been possible thanks to the principles pillaring its architecture, mainly genericity and extensibility. These principles were crucial to cope with change and innovation from the very beginning, and still are.

ARAS is composed of a variety of tools whose combination opens the path to its exploitation (as a research tool) in the most usual scenarios where mapping is involved, as the evaluation / modelling of new sensors or real-time mapping.

The research group responsible of this publication has been able to integrate their existing navigation and localization SW (C++). This has been always hardly integrated on others systems because of some lack of standardization and homogeneity in the research and industry fields. With the adoption of ROS as the core software system, the group expects to avoid this issue from now on. Then, the already developed algorithms can be tested together or separately from others modules that are part of the robot.

The map generation may be useful in a variety of scenarios, some of them already mentioned as RF mapping for indoor scenarios (i.e. inside urban buildings). It has also 3D building information modelling (3D BIM) capabilities obtained from its RGBD sensors. With a 3D BIM functionality, the robot can inventory the elements in an office, for example.

The actual platforms costs (approx. 700€) are similar or slightly lower to other robotic commercial platforms (such as the one offered Clearpath or Erlerobotics manufacturers). But this one, that has not been designed expecting any income return or benefit, offers to its creators the certainty that everything may be changed, modified or adapted without limitations.

There has been found many problems with Ubuntu 16.04, ROS, Kinect_bridge (with IAI_KINECT) and OpenCV3.0, working all together. So, the platform has been keep to Ubuntu 14.04 and OpenCV2.4. This can be a future limitation when trying to use newest OpenCV3 functionalities, for example. This may be due that Kinect driver for ODROID XU4 is not well supported or maintained.

ODROD-XU4 GPU has been proved not powerful for image and cloud point real-time processing and visualization. Moreover, this GPU does not have the best drivers supports. Hence, the GPU is a bottleneck for the system, slowing down the image and cloud processing (0.3-0.5 fps) and resulting on a ultra-slow SLAM processing (0.1-0.2 fps) with RTAB-MAP.

6. OUTLOOK

After the execution of this research, it has been proposed to move to ZED stereocam with an NVIDIA Jetson GPU for better driver support, software integration and speed up image processing with CUDA®. This is a near future possibility, although more test with the Kinect v2 device are forecasted.

ACKNOWLEDGEMENTS

To the ROS community and more in particular to the developers of Kinect_2bridge and iai_kinect ros modules, that allowed an easy integration of Kinect v2 sensors on ROS environment. Also to the developers of robot_localization ROS module. Without their previous work this research could not have been done.

To Marc Chesa and Novak Vukmirika by their contributions to ARAS in the frame of their bachelors' thesis, and to CTTC's Communication Systems Division for their help with the design of the rover frame.

REFERENCES

Angelats, E. and Navarro, J.A., 2017. Towards a fast, low-cost indoor mapping and positioning system for civil protection and emergency teams. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 42, Proceedings of LowCOST3D 2017. 28-29 November 2017, Hamburg, Germany.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., ... and Ng, A. Y., 2009. ROS: an open-source Robot Operating System. *ICRA workshop on open source software*, 3(3.2), p. 5.

Fang, Z. and Zhang, Y., 2015. Experimental Evaluation of RGB-D Visual Odometry Methods. *International Journal of Advanced Robotics Systems*, 12, pp. 1-16.

Fiorillo, A. S. 1999. Design of an ultrasonic sensor to emulate bat bio-sonars. *IEEE Proceedings of Ultrasonics Symposium,* 1, pp. 409-412.

Ghotbi, B., González, F., Kövecses, J. and Angeles, J., 2016. Mobility Assessment of Wheeled Robots Operating on Soft Terrain. *Field and Service Robotics*, Springer International Publishing, pp. 331-344.

Groves, P. D., Wang, L., Martin, H. and Voutsis, K., 2014a. Toward a unified PNT - Part 1, complexity and context: Key challenges of multisensor positioning. *GPSWorld,* 25(10), pp. 18-49.

Groves, P. D., Wang, L., Martin, H. and Voutsis, K., 2014b. Toward a unified PNT - Part 2, ambiguity and environmental data: Two further key challenges of multisensor positioning. *GPS World,* 25(11), pp. 18-35.

Jiménez, A.R., Prieto, J.C., Ealo, J.L., Guevara, J. and Seco, F., 2009. A computerized system to determine the provenance of finds in archaeological sites using acoustic signals. *Journal of Archaeological Science*, *36*(10), pp.2415-2426.

McCarthy, C. and Bames, N., 2004, April. Performance of optical flow techniques for indoor navigation with a mobile robot. *IEEE International Conference on Robotics and Automation,* 5, pp. 5093-5098.

Montañés, J.A.P., Rodríguez, A.M. and Prieto, I.S., 2013. Smart Indoor Positioning/Location and Navigation: A Lightweight Approach. *International Journal of Interactive Multimedia & Artificial Intelligence*, *2*(2).

Navarro, M. and Najar, M., 2011, Frequency domain joint TOA and DOA estimation in IR-UWB. *IEEE transactions on wireless communications*, 10, pp.1-11.

Parés, M.E. and Colomina, I., 2015. On software Architecture Concepts for a Unified, Generic and Extensible Trajectory Determination System. *Proceedings of the ION GNSS+*, 08-12 September 2015, Tampa, Florida (USA).

Siegwart, R., Nourbakhsh, I.R. and Scaramuzza, D., 2011. Introduction to autonomous mobile robots. *MIT press.*

Jiang, T., Petrovic, S., Ayyer, U., Tolani, A. and Husain, S. 2015. Self-driving cars: Disruptive or incremental. *Applied Innovation Review*, 1, pp. 3-22.

Titterton, D. and Weston, J.L., 2004. *Strapdown inertial navigation technology.*Vol. 17, IET, 558 p.

Tunstel, E., Maimone, M., Trebi-Ollennu, A., Yen, J., Petras, R. and Willson, R., 2005. Mars exploration rover mobility and robotic arm operational performance. *IEEE International Conference on Systems, Man and Cybernetics,* 2, pp. 1807-1814.

Xie, P. and Petovello, M. G., 2015. Measuring GNSS multipath distributions in urban canyon environments. *IEEE Transactions on Instrumentation and Measurement,* 64(2), pp. 366-377.

Zienkiewicz, J., Davison, A. and Leutenegger, S., 2016, October. Real-time height map fusion using differentiable rendering. *IEEE/RSJ International Conference on Intelligent Robots and Systems* (IROS), pp. 4280-4287.