Robótica Cognitiva

Editores

Carlos Cerrada
José A. Cerrada
Enrique Valero
Ismael Abad

Editado: Universidad Nacional de Educación a Distancia (UNED)

Madrid, Julio de 2013

Edita:
Universidad Nacional de Educación a Distancia (UNED)
Imprime:
Universidad Nacional de Educación a Distancia (UNED)

D.L. M-20260-2013
ISBN: 978-84-695-8175-9

Lista de autores

Alacid, A.
Centre for Automation and Robotics CAR (UPM-CSIC)

Alonso, Fernando
Robotics Lab, Universidad Carlos III de Madrid

Aracil, Rafael
Centre for Automation and Robotics CAR (UPM-CSIC)
rafael.aracil@upm.es

Armada, Manuel
Centre for Automation and Robotics CAR (CSIC-UPM)
manuel.armada@car.upm-csic.es

Arredondo, María Teresa
Life Supporting Technologies, ETSIT-UPM

Balaguer, C.
Robotics Lab, Universidad Carlos III de Madrid
balaguer@ing.uc3m.e

Barea, R.
Departamento de Electrónica, Universidad de Alcalá
barea@depeca.uah.es

Barrientos, A.
Centre for Automation and Robotics CAR (UPM-CSIC)
antonio.barrientos@upm.es

Bengochea, José M.
Centro de Automática y Robótica (CSIC-UPM)
jose.bengochea@csic.es

Bergasa, L.M.
Departamento de Electrónica, Universidad de Alcalá
bergasa@depeca.uah.es

Breñosa, José

Centre for Automation and Robotics CAR (UPM-CSIC)

jose.brenosa@upm.es

Cañas, José María
Departamento de Sistemas Telemáticos y Computación (GSyC), Universidad Rey Juan Carlos
jmplaza@gsyc.es

Cogollor, José M.
Centre for Automation and Robotics CAR (UPM-CSIC)
jm.cogollor@upm.es

de la Villa, P.
Departamento de Biología de Sistemas. Universidad de Alcalá

Ejarque, G.E.
Centre for Automation and
Robotics CAR (UPM-CSIC)
gejarque@etsii.upm.es

Fernández, Roemi
Centre for Automation and
Robotics CAR (CSIC-UPM)
roemi.fernandez@car.upm-csic.es

Ferre, Manuel
Centre for Automation and
Robotics CAR (UPM-CSIC)
m.ferre@upm.es

Fioravanti, Alessio
Life Supporting Technologies,
ETSIT-UPM
afioravanti@lst.tfo.upm.es

Fotiadis, E.
Centre for Automation and
Robotics CAR (UPM-CSIC)
efstathios.fotiadis@upm.es

García, C.
Centre for Automation and
Robotics CAR (UPM-CSIC)

García, J.R.
Departamento de Electrónica,
Universidad de Alcalá

García, Jorge
Robotics Lab, Universidad Carlos
III de Madrid

Garzón, M.

Centre for Automation and
Robotics CAR (UPM-CSIC)

ma.garzon@upm.es

González, Maikel
Departamento de Sistemas
Telemáticos y Computación
(GSyC), Universidad Rey Juan
Carlos

González-Fierro, Miguel
Robotics Lab, Universidad
Carlos III de Madrid
mgpalaci@ing.uc3m.es

Gorostiza, J.F.
Robotics Lab, Universidad Carlos
III de Madrid
jgorosti@ing.uc3m.es

Hernández, Alejandro
Departamento de Sistemas
Telemáticos y Computación
(GSyC), Universidad Rey
Juan Carlos

Hernández, D.

Robotics Lab, Universidad Carlos
III de Madrid

Jardón, A.
Robotics Lab, Universidad Carlos III de Madrid
ajardon@ing.uc3m.es

López, E.
Departamento de Electrónica, Universidad de Alcalá
elena@depeca.uah.es

Maldonado, M.A.
Samsamia Technologies, S. L.

Martín, Alejandro
Robotics Lab, Universidad Carlos III de Madrid
aimartin@ing.uc3m.es

Monje, C.
Robotics Lab, Universidad Carlos III de Madrid
cmonje@ing.uc3m.es

Montes, Héctor
Centre for Automation and Robotics CAR (CSIC-UPM)
hector.montes@car.upm-csic.es

Morante, S.
Robotics Lab, Universidad Carlos III de Madrid

Moreno, Luis
Robotics Lab, Universidad Carlos III de Madrid
moreno@ing.uc3m.es

Ocaña, M.
Departamento de Electrónica, Universidad de Alcalá
mocana@depeca.uah.es

Pastorino, Matteo
Life Supporting Technologies, ETSIT-UPM
mpastorino@lst.tfo.upm.es

Poletti, G.
Centre for Automation and Robotics CAR (UPM-CSIC)

Prieto, V.
Grupo de Investigación en Fisioterapia en los Procesos de Salud de la Mujer. Departamento de Fisioterapia. Universidad de Alcalá

Puente, P.
Centre for Automation and Robotics CAR (UPM-CSIC)

Puglisi, Lisandro J.
Centre for Automation and Robotics CAR (UPM-CSIC)
lisandro.puglisi@alumnos.upm.es

Rey, Germán
Hospital RUBER Internacional, Unidad de Radiofísica, Madrid

Ribeiro, Ángela

Centro de Automática y Robótica (CSIC-UPM)

angela.ribeiro@csic.es

Rivas, Francisco
Departamento de Sistemas Telemáticos y Computación (GSyC), Universidad Rey Juan Carlos

Rojo, Javier

Centre for Automation and Robotics CAR (UPM-CSIC)

javier.rojo.lacal@upm.es

Salichs, Miguel A.
Robotics Lab, Universidad Carlos III de Madrid
salichs@ing.uc3m.es

Salinas, Carlota
Centre for Automation and Robotics CAR (CSIC-UPM)
carlota.salinas@car.upm-csic.es

Saltaren, R.
Centre for Automation and Robotics CAR (UPM-CSIC)
rsaltaren@etsii.upm.es

Sarria, Javier
Centre for Automation and Robotics CAR (CSIC-UPM)
javier.sarria@car.upm-csic.es

Sebastián, José María
Centre for Automation and Robotics CAR (UPM-CSIC)
jsebas@etsii.upm.es

Urdaneta, M.A.
Universidad del Zulia, Zulia, Venezuela

Víctores, J.G.

Robotics Lab, Universidad Carlos III de Madrid

jcgvicto@ing.uc3m.es

Yuste, M.J.
Grupo de Investigación en Fisioterapia en los Procesos de Salud de la Mujer. Departamento de Fisioterapia. Universidad de Alcalá

PRÓLOGO

Me es grato presentar el duodécimo *workshop* (taller de trabajo) del programa RoboCity2030-II, subvencionado por la Comunidad de Madrid para el periodo 2010-2013. Este programa agrupa a los centros de investigación en robótica más punteros tanto a nivel nacional como internacional: Universidad Carlos III de Madrid, Instituto de Automática Industrial del CSIC, Universidad Politécnica de Madrid, Universidad de Alcalá, Universidad Rey Juan Carlos y Universidad Nacional de Educación a Distancia. De esta forma, en esta iniciativa única, se obtienen importantes sinergias que permiten potenciar y coordinar las investigaciones en robótica.

Este taller de trabajo es el último de la serie de encuentros periódicos que se planificaron al comenzar la andadura de Robocity2030-II hasta finales de 2013. La experiencia de los todos los *workshops* anteriores ha demostrado que son un foro de encuentro y discusión idóneo para los investigadores de los seis grupos del Programa.

En esta ocasión el taller está dedicado a la *Robótica Cognitiva*, un tema de interés creciente en la comunidad robótica internacional, y se abordan aspectos de actualidad científica y tecnológica sobre la materia. En particular, se presentan trabajos de excelente calidad dentro de tópicos clave en la Robótica Cognitiva, como interacción humano-computador, métodos de aprendizaje, percepción e identificación de objetos y movimientos, tareas de inspección mediante robots y robótica cognitiva en el ámbito de la medicina.

Por último, me gustaría destacar la excelente cooperación de todos los grupos participantes para facilitar la realización de este *workshop*, y en particular la labor de organización llevada a cabo por el grupo de la UNED, en cuya sede se ha celebrado el evento. Entre todos se ha conseguido una cuidada selección de trabajos de gran nivel que espero sean de utilidad y referencia para toda la comunidad robótica española.

Carlos Balaguer
Coordinador Robocity2030-II-CM

ÍNDICE

CAPÍTULO 1

RECENT ADVANCES IN THE JDEROBOT FRAMEWORK FOR ROBOT PROGRAMMING

JOSÉ M. CAÑAS[1], MAIKEL GONZÁLEZ[1], ALEJANDRO HERNÁNDEZ[1] and FRANCISCO RIVAS[1]

[1] Universidad Rey Juan Carlos, Spain

Most of the intelligence of cognitive robots lies on their software, the way they manage knowledge and coordinate their sensing and actuation capabilities. In the last years several frameworks have appeared that simplify and speed up the development of robot applications. They favor the code reuse and take benefits from modern software engineering techniques.

This paper presents recent advances in the open source robotics framework Jderobot (http://jderobot.org). It is a distributed and component oriented software architecture which uses explicit interfaces among components and ICE as communication middleware. Jderobot provides many tools for the programming of cognitive robots: a template for controller components, visual Finite State Machine creation, a library for fuzzy controllers, recording and replaying sensor data, etc. It also includes tools to use new RGB-D sensors like Kinect or Xtion and support for the newest release of Gazebo simulator, which is becoming a *de facto* standard. The main new features and tools are described in detail.

Jderobot framework has more than 60000 lines of code and includes more than 30 different components. Two recent improvements are related with a better project management using CMake and simpler installation using Debian packages. This software has been used for more than ten years in research, teaching and commercial applications. More than one hundred fifty different users, mainly robotic students and developers, have taken advantage of it. Lessons learnt using this robot middleware are also summarized in conclusions.

1 Introduction

Most of the robot intelligence lies on its software. Once the robot sensor and actuator devices are set, the robot behavior is fully caused by its software. There are many different ways to program in robotics and none is universally accepted. Some choose to use directly languages at a very low level (assembler) while others opt for high-level languages like C, C++ or Java.

Good programming practices are an emerging field in the software engineer area but also in robotics. Several special issues of robotics journals (ARS Special Issue on Software Development and Integration in Robotics, 2006), books on the topic have been published (Kernighan, 1999) and also specific workshops and tracks have been created inside ICRA and IROS. The Journal of Software Engineering for Robotics (www.joser.org) has been promoting the synergy between Software Engineering and Robotics meanwhile the IEEE Robotics and Automation Society (TC-SOFT) have founded the Technical Committee for Software Engineering for Robotics and Automation.

Compared with other computer science fields, the development of robot applications exhibits some specific requirements. First, liveliness and real-time processing: software has to take decisions within a fast way, for instance in robot navigation or image processing. The second requirement is about the software architecture; robot software has to deal with multiple concurrent sources of activity, and so tends to be multitasking. Third, computing power is usually spread along several connected computers, and so the robotic software may be distributed. Fourth, the robotic software typically deals with heterogeneous hardware. New sensors and actuator devices continually appear in the market and this makes complex the maintenance and portability to new robots or devices. Fifth, the robotic software usually includes a Graphical User Interface (GUI), mainly for debugging purposes. Sixth, the robotic software should be expansible for incremental addition of new functionality and code reuse. Seventh, the simulators are very useful in robotics software debugging.

Mobile robot programming has evolved significantly in recent years, and two approaches are currently found. In the classical approach, the application programs for simple robots obtain readings from sensors and send commands to actuators by directly calling functions from the drivers provided by the seller. In the last years, several frameworks (SDKs) have appeared that simplify and speed up the development of robot applications, both from robotic companies and from research centers, all of them with closed and open source. They favor the portability of applications between different robots and promote code reuse.

First, they offer a simple and more abstract access to sensors and actuators than the operating systems of simple robots. Using the SDK hardware abstraction layer it deals with low level details accessing to sensors and actuators, releasing the robotics programmer from that complexity.

Second, the SDK provides a software architecture for robot applications. It offers a particular way to organize code, handling of code complexity when the robot functionality increases. There are many options: calling to library functions, reading shared variables, invoking object methods, sending messages via the network to servers, etc. Depending on the programming model the robot application can be considered an object collection, a set of modules talking through the network, an iterative process calling to functions, etc.

Third, usually the SDK includes simple libraries, tools and common functionality blocks, such as robust techniques for perception or control, localization, safe local navigation, global navigation, social abilities, map construction, etc. This way SDKs shorten the development time and reduce the programming effort needed to code a robotic application as long as the programmer can build it by reusing the common functionality included in the SDK, keeping themselves focused in the specific aspects of their application. The robot manufacturers sell them separately or include them as additional value with their own SDKs. For example, ERSP includes three packages in the basic architecture: one for interaction, one for navigation and another for vision.

We present our open-source robotic software framework, named Jderobot, which is component oriented, uses ICE as communication middleware and includes several useful tools and libraries. Several sensor and actuator drivers have been programmed or reused from the open-source community. This framework has been widely used in our group for research and teaching for more than ten years. Jderobot has been designed for scenarios with sensors, actuators and intelligent software in between.

The typical scenario is robotics, but also computer vision and home automation.

Why open-source in robotics research? It provides independence on robot manufacturers and so it may support robots from different companies. With the freedom to use and modify the software its final quality does not depend so much on the debugging speed of a single company. The distribution of the source code makes (the distribution) debugging faster. In research, algorithms and results can be replicated and compared. Standard tools (for instance, simulators) and public access to sensor and data repositories (for instance, databases with input data and ground truth for robot localization), etc. help on this. This sharing makes the improvements come more easily and speeds up the advances in the robotics community. In addition, one strong motivation is the feeling of contributing to the community and returning the favor. We have extensively used open-source libraries and tools in our research (OpenCV, Gazebo, GTK, etc.).

Jderobot has enhanced thanks, in a large part, to the robotics community of the Universidad Rey Juan Carlos (Spain). New versions have been releasing including new functionalities. The last version, which is the 5.1, includes new applications, tools to facilitate and improve the management and features to make easier both the use and installation to standard users and also to developers. The new features to highlight on this version are the support to newer versions of the Gazebo simulator. This support is implemented by the gazeboserver driver which acts as an intermediary between the simulator and other applications though ICE interfaces. Another improved component is introrob*,* a teaching tool successfully used in the robotics subject in the Universidad Rey Juan Carlos.

Tools like CMake (see section 7.1) are also included in the new version of Jderobot. The use of this powerful tool makes the compilation task of all the framework libraries, components and interfaces so simple. In addition, Jderobot has also a set of Debian packages allowing to the user to install each of the components either individually (using atomic packages) or jointly (using virtual packages).

2 Related works

Robotic frameworks can be grouped in two main paradigms, those tightly coupled with a cognitive model in their designs and those designed just from a pure engineering criteria. The first ones force the user to follow a set of rules in order to program certain robotic behavior, while the second ones are just a collection of tools that can flexibly be put together in several ways to accomplish the task.

Cognitive robotic frameworks were popular in the 90s and they were strongly influenced by the Artificial Intelligence (AI), where planning was one of the main keys. Indeed one of the strengths of such frameworks was their planning modules built around a sensed reality. A good example of cognitive frameworks was Saphira (Myers, 1998), based on a behavior-based cognitive model. Even though the underlying cognitive model usually is a good practice guide for programming robots, this hardwired coupling often leads the user to problems difficult to solve when trying to do something the framework is not designed to do.

Key achievements of modern frameworks are the hardware abstraction, hiding the complexity of accessing heterogeneous hardware (sensors and actuators) under standard interfaces, the distributed capabilities that allow to run complex systems spread over a network of computers, the multi-platform and multi-language capabilities that enables the user to run the software in multiple architectures, and the existence of big communities of software that share code and ideas.

Current robotic frameworks focus their design on the requirements that robotics applications need and let the users (the programmers) to choose the organization that better fits with their specific application. Main requirements driving the designs are: multi-tasking, distributed, easy to use and code reusability. Another requirement, probably the main key, is the open source code. That creates a synergy between the user and the developer.

That is why two of the most popular robotic frameworks in the last years are open-source: Player/Stage (Gerkey, 2003) which has been the standard *de facto* in most of the last decade and ROS (Quigley, 2009) which is taking the place currently. As seen in other major software projects as GNU/Linux kernel or the Apache web server, to name but a few, the creation of communities that interact and share code and ideas, could be a great benefit to the robotic community. Main examples of open source modern frameworks are the aforementioned Player/Stage and ROS. Another important example is ORCA (Brooks, 2005). We briefly describe them.

There are other open source frameworks that have had some impact on current the state of the art, like RoboComp (Cintas, 2011) by Universidad de Extremadura, CARMEN (Montemerlo, 2003) by Carnegie Mellon and Miro by University of Ulm. All of them use some component based approach to organize robotic software using ICE, IPC and CORBA, respectively to communicate their modules.

We can find non open source solutions as well, like Microsoft Robotics Studio or ERSP by Evolution Robotics. Those are also very powerful platforms but their main disadvantage is just the open-source advantage: you cannot share code with other robotics groups without a license of the corresponding software.

2.1 Player/Stage

This framework provides a robotics environment dividing it into two sides: Player which is a robot device server and the multiple robots simulator named Stage. To support the development of robotic applications this suit also includes several tools and libraries. Player/Stage has been the most popular framework in the last decade with a big community giving support and also developing a great collection of drivers and algorithms around it. It is completely platform independent and most common programming languages, like C/C++, Python or Java, are supported.

Player provides a network interface to the robot hardware through a collection of standard interfaces that provides a hardware abstraction layer. These standard interfaces are implemented by drivers, one for each different hardware. Following this idea, the user only needs to know the standard way to use of each type of device not every device of every different manufacturers or models. Player exports its devices through a standard TCP network connection (other transport layers are available as well) enabling the user to build distributed systems across a network of computers. Even though, Player was not designed as a component based software, its architecture employs many component based ideas. In addition to hardware drivers, Player has a collection of algorithms like local navigation algorithms, vision related algorithms or localization algorithms.

Stage is a 2D multiple robot simulator that can provide its simulated devices through a Player server. There is a big variety of devices Stage can simulate, like laser rangers, robotic platforms and cameras, to name a few.

The typical organization of a robotic application written in Player/Stage is one or more client programs subscribed to a Player server which are serving the robot hardware. Clients send and receive the data using the known interfaces. So, we can say Player/Stage has a centralized architecture, where Player server assumes the main role.

To implement a new functionality on the server side the user can write a new driver which implements an interface. This is not usual due that developing on this side is a bit harder than on the client side.

2.3 ORCA

Another important example is ORCA which is a component based framework released several years before ROS. ORCA is multi-platform and multi-language as well. The aim of its developers was to increase the software reuse among the robotic community, so they create a component repository for robotics applications and they provide some of the needed glue to connect them. In a previous version, they coded the middleware that enabled components to communicate. Later, they realized that programming a middleware was out of the range of their interests, besides being a complex and time consuming task, so they replaced it with the professional grade middleware Ice from ZeroC (Henning, 2004) that allowed them to focus in the robotic problems. ORCA is the major source of inspiration for the actual version of the Jderobot and some of its core components are closely related to our framework.

2.2 ROS

ROS is one of the biggest frameworks nowadays. It was founded by Willow Garage as an open source initiative. Currently, it has a growing community and its site hosts a great collection of hardware drivers, algorithms and other tools. It is a multi-platform and multi language framework.

The main idea behind ROS is an extremely easy to use middleware that allows connecting several components, implementing the robotic behavior, in a distributed fashion over a network of computers using hybrid architecture. ROS is developed under hybrid architecture by message passing and RPCs. Message passing of typed messages allows components to share information in a decoupled way, where you do not know which component send you a message, and vice versa, you do not know which component or components will receive your message. RPC mechanisms are available as well. Resources can be reached through a well defined naming policy.

The ROS core libraries implement the communication mechanisms and a set of tools to help with tasks as project management (CMake based), system debugging and centralized logging. A set of official libraries implements standard messages (sensors, geometry, action...), well known robotic algorithms for navigation or sensor analysis, and powerful tools as the Rviz 3D visualization environment for robots. A simulation environment is provided through the Gazebo 3D simulator, which started as part of the Player/Stage project, but now is supported by Willow Garage.

A typical application written in ROS is a collection of components interacting with each other (no server/client model) distributed among a network of computational nodes. Often a user can just use some of the components found in the ROS repositories, tweak some parameters and make them interact with his own coded components.

3 Jderobot

The Jderobot platform is a component based framework that uses the powerful object oriented middleware Ice from ZeroC as *glue* between its parts. This important design decision allows Jderobot to run in multiple platforms and to be programmed with the most common programming languages (all the languages supported by Ice indeed). Jderobot components can also be distributed over a network of computational nodes and by extension use all the mechanisms provided by Ice as secure communications, redundancy mechanisms or naming services.

3.1 Design principles

Following current trends in robotic software engineering, we aimed to design a framework with these major requirements:

- Component based.
- Multi-platform and multi-language support.
- Distributed.
- Strongly typed interfaces.
- Open source.

The main unit is the *component*. A component is an independent process which has its own functionality, although it is most common to combine several of these in order to obtain a more complex behavior. There are several types of components, according to the functionality of the component could be classified in:

- Driver-components: offer to developers a HAL (Hardware Abstraction Layer) to communicate with the different devices that are equipped by the robot (sensors and actuators).

- Tools-components: This kind of components uses the driver-components to communicate with the robot (real or simulated) and process the received data. Then the result of this process is sent to the robot through the driver-component.

The communication between Jderobot components occurs through the ICE (Internet Communications Engine) middleware. Using this concept, Jderobot allows directly these components to be distributed as it uses the network as a link. The ICE communication is based on interfaces. This middleware has its own language named *slice* which allows the developer to define their custom interfaces. Those interfaces are compiled using ICE built-in commands, generating a translation of the slice interface to various languages (Java, C++, Python…). This allows communication between components implemented in any of the languages supported by ICE.

The entire configuration needed by the components is provided by its configuration file.

Another important feature of Jderobot is the wide use of third party software and libraries (all of them open source) which greatly extends its functionality. Among them include: OpenCV for image processing; PCL for point cloud processing; OpenNi for the RGB-D support, Gazebo as the main 3D robot simulator and GTK+ for the GUI implementations. Besides, Jderobot provides its own libraries which give to robotics commonly used functionality for the developing of applications under this framework.

3.2 Open source project

As project, Jderobot stands out for being maintained by a community of developers formed largely by the Robotics Group at the Universidad Rey Juan Carlos. As a source of documentations, this framework has its official wiki (http://jderobot.org/index.php/Main_Page) where everyone is able to download the entire project. To complete the support, a mail list is also available to interact with other developers.

Jderobot is a platform that has evolved significantly since its inception and it is currently at 5.1 version. The following sections describe some of the latest developments.

4 VisualHFSM and fuzzy controllers

There are many ways to organize the control and perception code on board a mobile robot. One successful way is the Finite State Machines (FSM). With them the robot behavior is defined by a set of states, each of which performs a particular task. The robot can switch from one state to another through transitions (conditions of stay or change), depending on certain events or conditions, internal or external. A tool has been developed and included in Jderobot to graphically design hierarchies of FSM, insert the specific code of states and transitions, and automatically generate the component source code in C++.

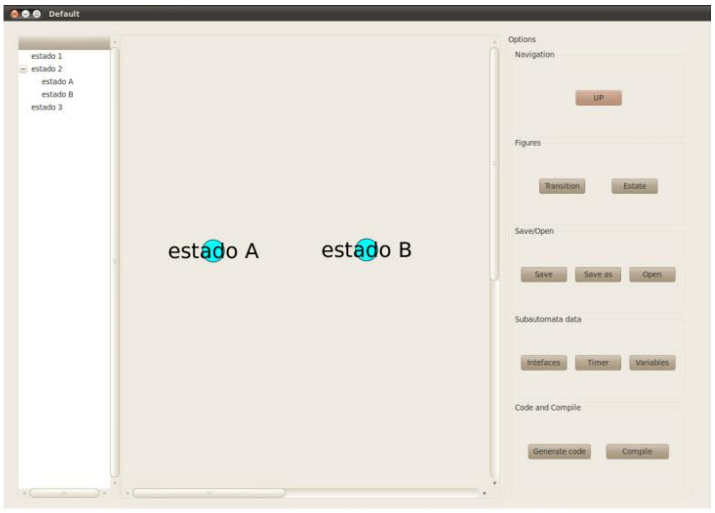


Fig. 1. VisualHSFM Graphic User Interface

Another tool when programming cognitive robots is the fuzzy logic. It allows programming the robot behaviors in terms of simple words and rules which are close to the natural language. In Jderobot we have developed the Fuzzylib library to design and program fuzzy controllers. The control rules are written in a file with simple rules like:

IF (left_obstacle_distance = small)
AND (advance_speed = high)
THEN (rotation_speed = right_high)

The fuzzy labels and variables are also described in such file following a trapezoidal pattern and they are linked in the software to input and output variables of the control program. The controller automatically fuzzyfies the input variables, apply the rules and defuzzifies the output following a Center of Mass combination.

5 Gazebo support

Gazebo[1] (Koening, 2004) is the preferred simulator in Jderobot framework. It is a 3D open source simulator which offers a rich environment to quickly

[1] http://gazebosim.org

test multirobot systems and which simulates several robots and sensors (such as cameras, laser, etc) in a realistic way. All simulated objects have their own mass, velocity, frictions and numerous other attributes that allow them to behave realistically when pulled, knocked over or pushed.

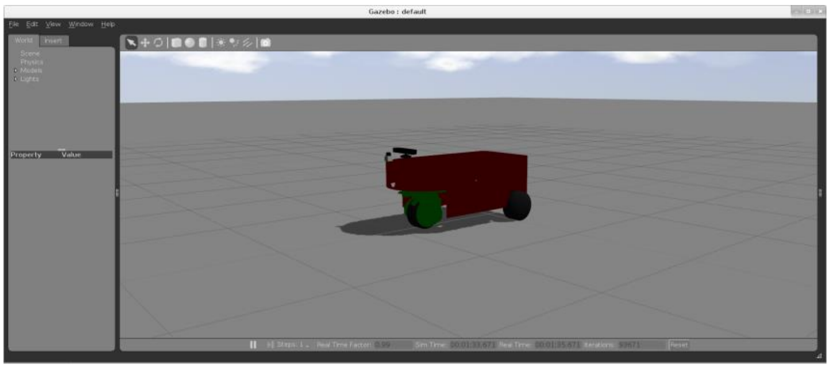


Fig. 2. Gazebo with Autonav robot.

Recently Defense Advanced Research Projects Agency (DARPA) announced its Robotics Challenge for disaster robots. The goal of this challenge is to develop ground robots capable of executing complex task in order to extend aid to victims of natural or man-made disasters and conduct evacuation operations. Selecting Gazebo simulator as the standardized simulation environment and will be provided by the Open Source Robotics Foundation. The teams that do not want or cannot afford to build their own robots will be able to prove themselves using Gazebo simulation environment and later may receive a real robot to use in the competition. The choice of this simulator is successful since it has become a standard in one of the main research project around the world.

Gazebo offers a big variety of sensors, actuators, robots, objects and maps. Also it incorporate tools to design new elements, which integration with the simulator is very simple. Fig. 2 shows a robot integrated in the simulator. This robot is defined with a drive and steering wheel and three sensors such as Kinect, laser and encoders. In the world files is possible to define the simulated world and the different elements that defined the environment. This allows creating a simulated world very close to the real environment where the robot is going to be deployed.

Jderobot 5.1 offers support for the last version of this simulator (Gazebo 1.8). It offers a driver-component call gazeboserver who intermediates between Gazebo and the components in Jderobot 5.1. The architecture of gazeboserver is based on plugins. A plugin is a dynamic library load in execution time when the simulator starts. In each plugin, the functionality of the different devices, what are contained in the robot, is defined. The plugins that the simulator should load must be indicated in the configuration world file.

Plugins are not only connected with the simulator, they also create an ICE interface which allows communicating the simulator with Jderobot components, sending or receiving information about the robot state. Fig.3 shows an example of a component connected with gazeboserver. We can observe that the components of the system are connected to the platform without taking into account if we are using a simulator or the real robot.

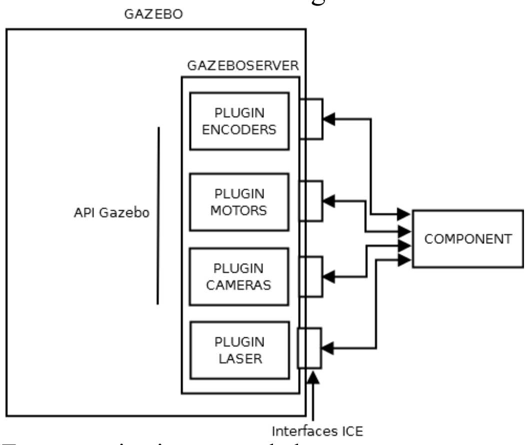


Fig. 3. ICE communication example between a component and Gazebo

6 Introrob component for teaching robotics

Introrob is a teaching tool that Universidad Rey Juan Carlos master students use to develope their practice into the Robotics subject. This component is connected to the Gazebo simulator using gazeboserver. The simulated robot used with introrob is the pioner2dx with a set of sensors (laser ranger, odometry and cameras) and actuators (motors, and cameras Pan&Tilt).

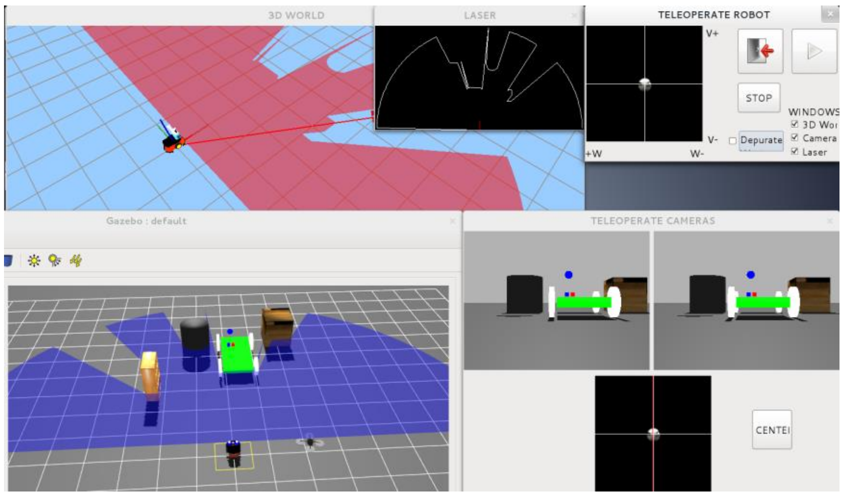


Fig. 4. Introrob running on Gazebo

This component provides to the students a GUI (Fig. 4) where they can find all the data collected from the sensor of the simulated robot. This GUI also includes some features to make easier the debug of their own algorithms. It also gives some functionality to teleoperate the robot using some joysticks changing the linear and angular speed as well as changing the cameras orientation. In addition, introrob shows a 3D viewer based on openGL from which it is possible to perform certain interactions (change the point of view, draw objects, etc…). This GUI also includes a play button to start a certain algorithm developed by the student.

For the algorithm developing, there is available a file called MyAlgorithms.cpp which is launched when pressing the PLAY button. In this way, the student can be abstracted from the rest of the application code and focus only on a single file. Besides, this file also contains a template with examples that allows students to speed up their learning with the tool.

Introrob has also an own API that simplifies even more the developing task to the students offering fully abstract functions. They do not need any prior knowledge about the communications protocol nor data structure. Some examples of this API functions are:

- setV(5): receives as a parameter the speed in mm/s and it is sent to the robot.
- getLaser(): returns a 180 position vector with all the measurements of the laser range scanner.

It also provides some graphical functions to simplify the visualization of their algorithms results.

- drawSphere(x,y,z): receives a 3D point and draws a sphere there in the OpenGL world.
- drawSegment(p1,p2): receives two 3D points and draws a segment that joins them.

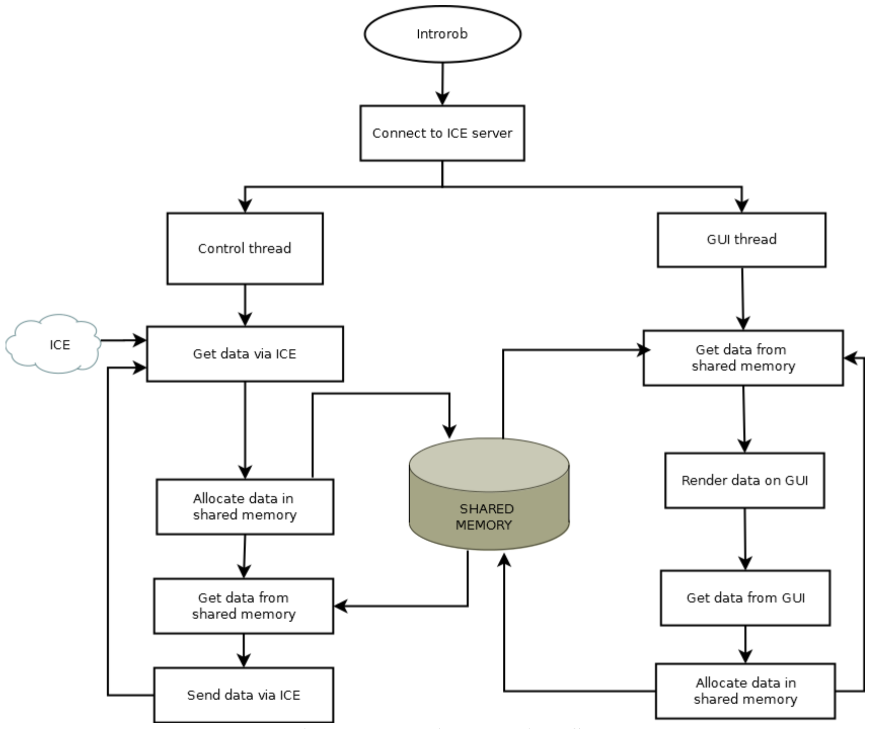


Fig. 5. Introrob execution flow

Internally, the architecture of this component is divided into two main threads: control thread and GUI thread.

- Control thread: this thread asks gazeboserver for the data from the sensors of the simulated robot and locates them in shared memory. Then, it gets the data generated by the GUI thread from shared memory and sends the corresponding orders to gazeboserver.

- GUI thread: this thread is always taking the data allocated by the control thread and displaying it in the GUI. Besides, it puts again in shared memory the data generated, or by the user interaction or their code, to send them to the simulator through the control thread.

Students from Computer Vision (Master degree) use introrob to develope algorithms oriented to the recognition of the environment using cameras carried by a robot. Furthermore, Telematic and Computer Systems (Master degree) students are focused on navigation algorithms with the aim of recognizing and following a line drawn on the floor.

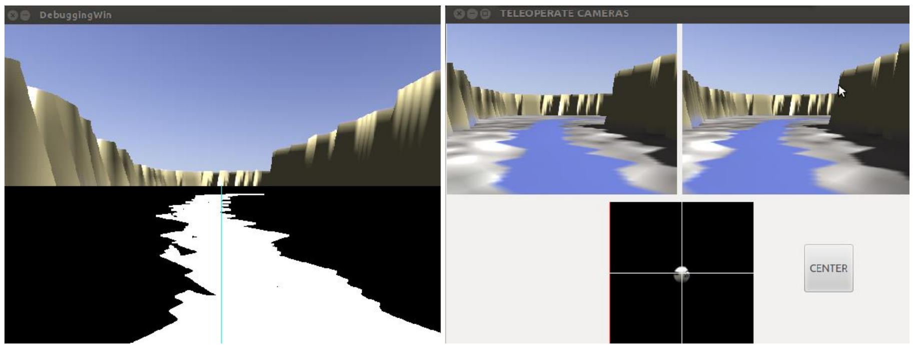


Fig. 6. Student's example algorithm

7 CMake and Debian packages

Jderobot 5.1 has new features that facilitate the management and maintenance of the entire platform and its source code. On the one hand, Jderobot includes the new CMake tool that lets you build simply the entire source code of the platform. On the other hand, it also has a set of Debian packages to simplify the Jderobot installation, both the platform itself and all their third part dependences. These packages are available for the two Linux distributions: Ubuntu 12.04 and Debian Wheezy, both for the 32 bits version.

7.1 CMake

CMake is a build software tool that makes easier the multiplatform code compilation both for users and developers.

At the user level, a feature that highlights is the information it provides about what is happening during the compilation process. This helps the user to resolve any problems that arise. But mostly, the main advantage using CMake is its easiness. To compile a whole project (regardless of the complexity or size) the user has only to execute two commands: *cmake* and *make*.

At developer level, CMake provides agility when adding new items to project and it needs too few configurations to compile and link a lot of software.

The CMake configuration is based on CMakeLists.txt files. These files include everything needed to build the code. The designed solution in Jderobot divides its configuration into two different types of CMakeLists files:

- Type 1 or primary: these files are used directly from the *cmake* command and are the responsible for initiating the process of building. Into them are defined common characteristics for all elements to be compiled and linked (libraries, interfaces and executables).
- Type 2 or secondary: these files are called by other primary files and/or secondary files recursively. They define the relevant data of each element, such as:
  + Source files.
  + Dependences.
  + Location of the dependences (both, headers and libraries).

Following this design, Jderobot 5.1 offers the user three different ways to build the code:

- Compilation from the trunk. The compilation process is initiated by the main CMakeLists.txt located at the top of the trunk project directory. This will build all components, libraries and interfaces of the platform.
- Compilation by components. This second way allows the users not to build all the elements, but those they only need. To do this, the compilation must to start from the main CMakeLists.txt file located in the /build directory of the wanted component. This method requires that the directories hierarchy of the platform has to be respected. This is because the paths used for dependency resolution are defined relatively.

- Independent compilation. This allows the user the ability to download only the source code of a particular component and build it without downloading the entire tree directory of the whole platform. In this case, the prerequisite to build the executable is that the users must have installed on their computer all the libraries on which the component depends.

7.2 Debian packages

Debian packages are compressed files that contain the elements to install on our computer, information on how and where to be installed and information about their dependencies.

There is much heterogeneity among the different systems of students and Jderobot users. This complicates the installation of software consisting of a large number of applications which depends in turn on external software. Debian packages facilitate this task by reducing installation to simple commands allowing the user to have all the necessary software automatically.

For the management and installation of Debian packages, there are tools like *apt* or *aptitude*. Using these commands any user can download and install a package from a repository, resolve its dependencies or uninstall it very easily later.

These tools enhance the user's first experience with the platform, making friendlier their first contact with it.

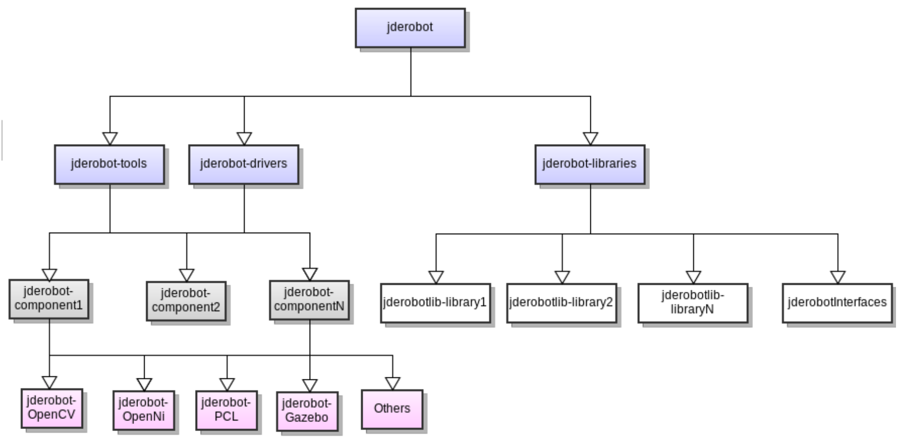


Fig. 7. Jderobot package design

Packet structure designed for Jderobot 5.1 (Fig. 8) is divided into:

- Atomic packages. Jderobot has a package for each of the components (jderobot-componentN) and libraries (jderobot-libraryN) that make up the platform. This allows the users to install, individually, those parts their want to use or develop.

- Third part packages. To make easy the installation, not only the components but the entire external software, Jderobot includes Debian packages for that software used by the components and which is not officially available in the repositories (Ubuntu 12.04 or Debian testing).

- Virtual packages. Virtual packages are packages that do not contain the elements to be installed in our system, but information about which atomic packages should be installed when you install one of them.
Jderobot 5.1 has different virtual packages that encapsulate:

+ Libraries: All Jderobot libraries are grouped into the virtual package jderobot-libraries.
+ Driver-components: jderobot-drivers.
+ Tools-components: jderobot-tools.
+ Components with related functionality, as teaching-robotics which includes introrob, gazeboserver and gazebo.
+ jderobot package. This installs all atomic packages and third party software packages that make up the entire platform.

8 Conclusions

We have presented the Jderobot open-source framework for programming cognitive robots. We have used it in teaching, development of robotic applications and research for more than 10 years. Its software architecture is based on distributed components using the network as communication link. It provides several drivers for accessing different sensors, actuators and robots like Pioneer2DX or Nao humanoid. It also includes a powerful set of tools and libraries that speed up the creation of new robotics applications. In this paper we have described the main new features developed for the Jderobot-5.1 release.

First, some cognitive tools have been incorporated to the framework. The visualHFSM provides a tool for generating robot behaviors using hierarchical Finite State Machines. It lets the programmer to focus on the behavior logic, in terms of states and transitions, more than on implementation details. Most of the final control code is generated automatically by visualHFSM using an automata template and an abstract description of the FSM. Another cognitive tool incorporated is the fuzzy library that eases the programming of fuzzy controllers.

Second, we have improved the connection between Jderobot and the latest releases of the Gazebo simulator. This open source simulator is becoming a *de facto* standard after the ROS developer team chose it as its reference simulator. More recently DARPA chose it for the first stage of the DARPA Robotics Challenge. We have developed a set of new Gazebo plugins that provides the standard Jderobot interfaces to sensors and actuators in the simulated world. The model of a brand new robot has been also created in the new Gazebo.

Third, we have improved the Introrob component that we use in robotics teaching. This Jderobot component is the base for the practices of students from several engineering and master courses. More than two hundred students in the last years have used Jderobot in the last years. They serve as testers giving feedback of the performance of the framework to the developers and providing experimental validation to the system. The specific Introrob component hides some of the complexity of robot programming, and so the students take shorter time to start using the framework, focusing on the robotics algorithms and techniques more than on the platform itself.

And fourth, we have changed two issues of the infrastructure of the whole Jderobot as a software project. We distribute it now as debian packages, both for Debian Linux and Ubuntu LTS. The creation of debian packages makes easier the installation and management of the platform. We have created atomic packages for each Jderobot component, library or tool. We have also prepared several virtual packages that include those atomic packages that make sense together. In addition, the last Jderobot release uses CMake as the project compilation tool, making simpler the inclusion of both new components and libraries. It has been an improvement from the previous compilation tools, automake and autotools, which are more complex.

We are currently working on several lines to extend Jderobot. First, to improve the use of ICE advanced capabilities like Icestorm and Icebox to take benefit of them. Second, on the integration of the new RGB-D commercial sensors (Kinect, Xtion, Primesense sensor) which will boost the developing of reconstruction and localization applications using point cloud information. Third, to give additional functionality to the Gazebo simulator, adding complete models of the Nao humanoid and RGB-D sensors. Fourth, to create tools like ROS's Bags. This kind of applications provides the developers the ability to record logs of scenarios to recreate them later. This feature gives the ability to prove different algorithms on exactly the same data source. Finally, we want Jderobot applications to be easily interactive with regular web browsers and other frameworks. With this concept, human users could easily modulate the applications on real time or see their processing outputs, even in embedded and screenless systems. In addition, Jderobot applications could easily be integrated with other existing frameworks. We are exploring new generation webservices like API-REST for this. Nevertheless these extensions, the framework is stable and with enough functionality. The main efforts will be devoted to extensively use it for research more than to expand it.

References

Brooks A., Kaupp T., Makarenko A., Williams S. and Orebäck A. 2005. Towards component-based robotics. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, (págs. 163-168).

Cintas R., Manso L., Pinero L., Bachiller P. and Bustos P. 2011. Robust behavior and perception using hierarchical state machines: A pallet manipulation experiment. *Journal of Physical Agents* , 35-44.

Galvan S., Botturi D., Castellani A. and Fiorini P. 2006. Innovative Robotics Teaching Using Lego Sets. *EEE International Conference on Robotics and Automation Orlando.* Florida.

Gerkey B.P., Vaughan R.T. and Howard A. 2003. The player/stage project: tools for multirobot and distributed sensor systems. *Proceedings of the 11th International Conference on Advanced Robotics ICAR-2003*, (págs. 317-323). Coimbra (Portugal).

Henning, M. 2004. A new approach to object-oriented middleware. *Internet Computing, IEEE* , 66-75.

Hunt A. and Thomas D. 1999. *The pragmatic programmer: from journeyman tomaster.* Boston: Addison-Wesley Longman Publishing Co., Inc.

Kernighan B.W. and Pike R. 1999. *The Practice of Programming.* Addison-Wesley Professional Computing Series.

Koening N. and Howard A. 2004. Design and use paradigms for gazebo, and open-source multi-robot simulator. *In proceedings of 2004 IEE/RSJ International conference on Intelligent Robots and System.* Senday (Japan).

Konolige K. and Myers K.L. 1998. Artificial Intelligence and Mobile Robots: case studies of succesful robot systems. En R. P. In David Kortenkamp, *The Saphira architecture for autonomous mobile robots* (págs. 211-242). MIT Press, AAAI Press.

Montemerlo M., Roy N. and Thrun S. 2003. Perspectives on standardization in mobile robot programming: the Carnegie Mellon navigation (CARMEN) toolkit. *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-03), volume 3*, págs. 2436-2441.

Quigley M., Conley K., Gerkey B., Faust J., Foote T., Leibs J., Wheeler R. and Ng A. 2009. Ros: an open-source robot operating system. *ICRA Workshop on Open Source Software* .

CAPÍTULO 2

DESCRIPCIÓN GENERAL DEL SISTEMA DE INTERACCIÓN HUMANO-ROBOT RDS (ROBOTICS DIALOG SYSTEM)

F. ALONSO-MARTIN[1], J.F. GOROSTIZA[1] y MIGUEL A. SALICHS[1]

[1]Robotics Lab, Universidad Carlos III de Madrid

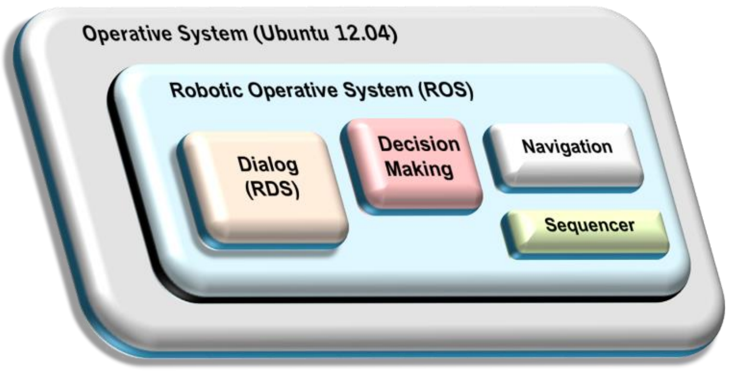
En este artículo se hace una breve descripción del sistema de interacción RDS (Robotics Dialog System). Este sistema, desarrollado dentro del grupo de investigación de robots sociales del RoboticsLab, es capaz de dotar a los robots que lo incorporan de capacidades de interacción que se asemejan a las que poseen los humanos. Se analiza el entorno software con el que convive, los robots donde se usa, y se describen los componentes que forman el sistema general. Finalmente se describen los aspectos generales que caracterizan el sistema.

1 Introducción: el ecosistema

1.1 El entorno Software

El sistema de diálogo que se presenta en este trabajo está formado por multitud de módulos que conviven a su vez con otros módulos pertenecientes a la arquitectura de control. Siguiendo un esquema basado en la modularidad y la abstracción se puede entender el ecosistema en el que convive el sistema de diálogo (ver Fig. 1). En el nivel inferior se encuentran los componentes hardware: robot, computadoras, etc. Cada computadora ejecuta un sistema operativo; en el caso de nuestros robots, concretamente una distribución de Linux: Ubuntu. A su vez, cada sistema Ubuntu, ejecuta la arquitectura de control robótica. La arquitectura de control es capaz de comunicar los distintos módulos que la conforman, proporcionando los mecanismos de comunicación necesarios. A lo largo de los años que ha durado la elaboración de la tesis doctoral en la que se enmarca este trabajo, esta arquitectura de control ha ido evolucionando.

Partiendo de una arquitectura definida teóricamente y conocida como AD, implementada y especificada en trabajos previos del grupo (Barber, 2002), ha sufriendo evoluciones constantes, hasta llegar a la arquitectura actual: una arquitectura híbrida entre AD y el nuevo "estándar de facto" en las arquitecturas robóticas, la arquitectura ROS. Esta nueva arquitectura de control, que se la puede denominar como AD-ROS, permite la convivencia de cualquier módulo desarrollado en "la antigua AD" y "la moderna ROS", de manera que se conserva todo el trabajo desarrollado durante años por nuestro grupo de investigación y se ha incorporado la posibilidad del uso de los miles de módulos desarrollados por la comunidad ROS. Esto, además presenta la ventaja de que cualquiera de los componentes desarrollados para este trabajo, y del sistema de diálogo completo, puedan ser usados por cualquier investigador, de cualquier lado del mundo, sobre cualquier robot capaz de ejecutar ROS. La arquitectura de control facilita la tarea de distribuir el software, ya que cualquier módulo puede estar corriendo en cualquier computador, incluso fuera del robot; en este sentido, la única restricción es dada por el módulo (o los módulos) que necesite interactuar directamente sobre un actuador y/o sensor, ya que necesitará comunicarse con él (de manera local o a través de la red).


Operative System (Ubuntu 12.04)
Robotic Operative System (ROS)
Dialog (RDS)
Decision Making
Navigation
Sequencer
Fig. 1. Ecosistema del sistema de interacción RDS

Dentro de la arquitectura robótica AD-ROS conviven multitud de sistemas: el sistema de toma de decisiones, el sistema de navegación, el secuenciador, etc. El sistema de diálogo es un sistema más dentro de la arquitectura de control, que es capaz de interactuar con el resto de sistemas, proporcionándoles la capacidad y la abstracción de realizar una interacción de alto nivel con el usuario.

1.2 El entorno Hardware

Actualmente, el sistema RDS constituye el sistema de interacción de cualquiera de los robots desarrollados por el grupo de trabajo de robots sociales del RoboticsLab. Los robots en los que está siendo utilizado con éxito son: el robot Maggie (Salichs, 2006) (Gonzalez-Pacheco, 2011), Mopi, y Flory.

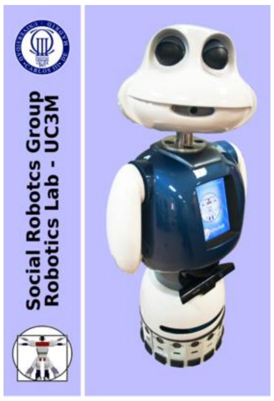


Fig. 2. El robot social Maggie

El robot social Maggie (ver Fig. 2) constituye una plataforma de investigación sobre la que han versado numerosos trabajos del grupo. En estos trabajos el robot ha sido ampliamente descrito. Para el desarrollo de este trabajo ha sido la plataforma preferida del autor, puesto que consta con el mayor tipo de modalidades de comunicación posibles, además de ser el primero de los robots creados y con el que comenzó esta línea de investigación.

Como se ha comentado, el sistema también ha sido probado satisfactoriamente en los demás robots social del grupo. El robot modular Mopi, es el encargado de dotar de movilidad al robot modular Flory.

Actualmente se está trabajando en la integración del sistema en una nueva plataforma asistencial.

2 La multimodalidad soportada por el sistema

El sistema RDS está concebido para trabajar, dentro del ámbito de la arquitectura de control, en múltiples robots. A continuación vamos a detallar cuales son estos canales de entrada y salida que soporta este sistema, poniendo especial hincapié en su uso en el robot Maggie, que soporta todos los canales de entrada y salida disponibles por RDS.

2.1 Canales de entrada de información

- *Audio*. Para recibir audio, es necesario del uso de uno o varios micrófonos situados estratégicamente. En el robot Maggie, el audio es recibido a través de los micrófonos incorporados en su cuerpo. En la base del mismo, formando una circunferencia de 40 cm de radio y a 21 cm del suelo, se encuentran 8 micrófonos que se conectan por USB al ordenador interno del robot. Estos 8 micrófonos se usan fundamentalmente para tareas de localización de la fuente sonora. Para el análisis de voz, el robot incorpora dos micrófonos en la cabeza. Para una interacción en entornos especialmente hostiles (mucho ruido ambiental), también es posible interactuar con el robot mediante auriculares inalámbricos o *bluetooth*, con la consiguiente perdida de naturalidad. El audio captado por el robot se usa fundamentalmente para las siguientes tareas: reconocimiento de voz basado en gramáticas, reconocimiento de voz de texto libre, detección de emociones en la voz, localización espacial de usuarios, cálculo de nivel de *arousal* (excitación) del entorno, generación de sonidos y acompañamiento musical que casen perfectamente con la voz recibida.

- *Visión*. El robot puede estar dotado de tres mecanismos basados en visión de percibir el entorno físico que le rodea: mediante cámara web (Maggie la incorporada en su boca), cámara de profundidad (Maggie, usa una cámara *Kinect*) y telémetros laser (Maggie incorpora un láser situado encima de la base móvil). Estos sensores están dedicados a tareas, algunas actualmente en desarrollo, de: navegación por el entorno, detección e identificación de usuarios, detección de gestos y poses, detección de objetos y lectura de texto escrito (OCR).

- *RFID*. El robot puede estar equipado de uno o varios lectores de etiquetas de radio frecuencia (RFID), concretamente el robot Maggie, consta de varios lectores incorporados en su cuerpo, uno en la cabeza y otro en la base, ambos de corto alcance (unos 20 cm. para leer una tarjeta), y otros dos en sus costados de mayor alcance (unos 100 cm. aproximadamente). La interacción por etiquetas está destinada fundamentalmente a tareas de identificación de objetos, como por ejemplo medicamentos, o incluso al control del robot por diálogos por etiquetas. En este modo por etiquetas de radio frecuencia, cada una de ellas representa, mediante un dibujo adecuado (pictograma), cada una de las posibles habilidades del robot. Este tipo de interacción es adecuada para niños pequeños, personas muy mayores o entornos muy ruidosos donde la interacción por voz se ve notablemente afectada.

- *Tacto*. El robot puede estar dotado de varios sensores capacitivos 10 capaces de detectar cuándo el usuario está tocando el robot en la parte del cuerpo donde esta situado dicho sensor. El sensor capacitivo no es capaz de notar diferencias de presión en el tacto, siendo únicamente binario (tocado / no tocado). Se usa como posible de entrada de información para el sistema de diálogo así como para simular cosquillas en el robot.

- *Smartphones* y *tablets*. Es posible también la entrada de información mediante la tableta incorporada en el pecho del mismo o mediante teléfonos inteligentes. En ambos se presentan un conjunto de opciones (dependiendo del diálogo y la finalidad concreta de la interacción), que el usuario puede ir activando/desactivando mediante sus dedos.

2.2 Canales de salida de información

Como sistema de diálogo multimodal simétrico, la multimodalidad se presenta tanto a la entrada del sistema como a la salida del mismo, por lo que son necesarios diversos canales de salida o expresión de información:

- *Audio*. El robot Maggie tiene un sistema de cuatro altavoces situados debajo de su cabeza, los cuales permiten comunicarse con el usuario mediante voz y sonidos. Estos altavoces son usados para la generación de voz con emociones, sonidos no verbales, generación musical y reproducción de música.

- *Gestos expresivos del diálogo*. Mediante brazos, cabeza, párpados y base móvil el robot es capaz de realizar gestos que complementan el diálogo. Dentro de este repertorio de gestos se encuentran algunos como: negaciones, afirmaciones, seguir con la mirada al usuario, baile, navegación por el entorno. Estos gestos se realizan de manera sincronizada con la voz y los sonidos mediante el propio gestor de diálogo.

- *Infrarrojo*. Maggie es capaz de controlar electrodomésticos mediante comunicación por infrarrojos. En este sentido, el robot incorpora un mando de infrarrojos programable que permite emitir la señal adecuada para encender/apagar televisores, aires acondicionados, cadenas musicales, etc.

3 Componentes de RDS

El sistema de diálogo propuesto para nuestro robot (ver Fig. 3), necesita una entrada de información coherente en el tiempo y de contenido semántico, obtenida de fusionar la información que proporcionan cada uno de las entradas sensoriales posibles.

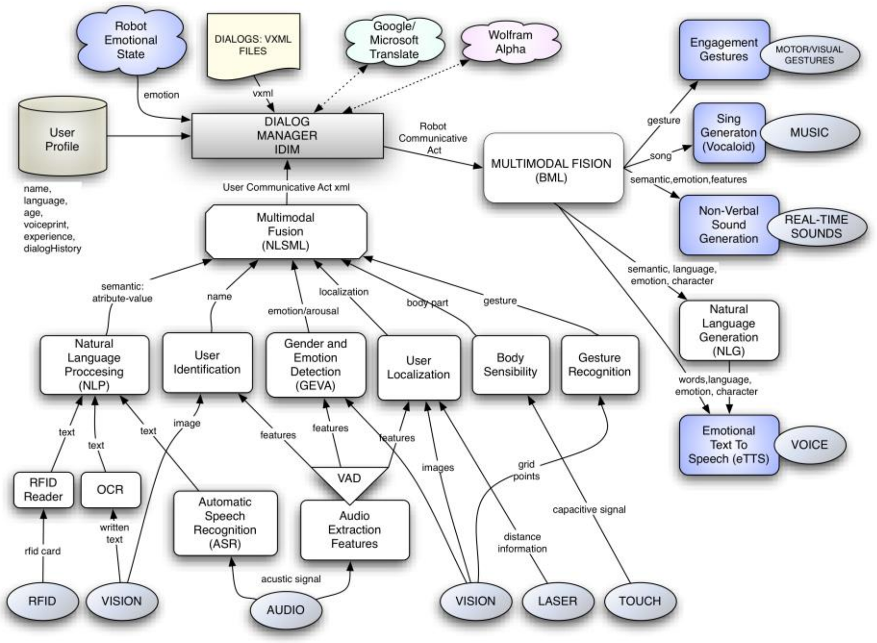


Fig. 3. Robotics Dialog System

El sistema de diálogo está compuesto por diversos de componentes que realizan tareas específicas de manera desacoplada y distribuida. Estos componentes se pueden agrupar en:

1. **Gestor del Diálogo (IDiM)**: que se encarga de gestionar los turnos de comunicación y dadas unas entradas de información generar unas salidas. Es un gestor basado en el paradigma de rellenado

de huecos de información, concretamente basado en el estándar Voice XML 2.1 pero extendido con funciones multimodales. Otros autores, también usan este sistema de interacción, si bien casi ninguno de los estudios se ha realizado sobre robots sociales, estando la mayoría de ellos centrados en entornos telefónicos o web.

Este paradigma trata de rellenar un hueco de información antes de pasar al siguiente. Para rellenar cada uno de los huecos de información es posible hacerlo mediante cualquiera de los canales de comunicación posible: voz, tacto, gestos, etc. o la combinación de ellos en un mismo mensaje. Un ejemplo sencillo puede ser intentar reservar un vuelo, en el que los huecos de información podrían ser: hora de salida, hora de llegada, ciudad origen, ciudad destino y compañía área. Todos estos huecos se pueden ir rellenando mediante sucesivas preguntas y respuestas entre el usuario y el sistema, o mediante una única frase, como por ej: "quiero salir de Madrid a las 7 de la mañana y llegar a París a las 9 de la mañana con Rynair".

2. **Módulos de entrada de información**: los componentes que trabajan sobre las entradas sensoriales para procesar dicha información y entregar al Gestor de Diálogo (IDiM) información relevante. Son los siguientes:

   - *Reconocimiento Automático del Habla (ASR)*: Se encarga de traducir voz a texto. Para ello implementamos un sistema modular que permite conectar varios reconocedores de voz, que pueden trabajar en paralelo sobre las mismas muestras de audio. Actualmente trabajamos con el reconocedor de voz basado en gramáticas de *Loquendo* y el reconocedor basado en el servicio web de *Google ASR*. Ver (Alonso-Martin, 2011a).

   - *Procesamiento del lenguaje natural*: que a su vez trabaja conjuntamente con el módulo de conversión de voz a texto (ASR) para extraer el significado semántico relevante del texto reconocido.

   - *Conversor de texto escrito a texto máquina (OCR):* convierte el texto escrito a mano en texto válido para el sistema de diálogo, para ello usamos técnicas de OCR. Para mayor detalle ver (Alonso-Martin, 2011b).

- *Lectura de etiquetas de radio frecuencia (RFID):* lee la información escrita en etiquetas de radio frecuencia, tanto activas como pasivas. Para ampliar información consultar (Corrales, 2009).

- *Identificación de usuarios*: este componente se encarga de identificar al usuario con el que está dialogando por su tono de voz. Para ello en la fase de registro del usuario con el sistema, el sistema guarda información del timbre del usuario (*voiceprints*).

- *Identificación de emociones en los usuarios*: las emociones del usuario son captadas de manera multimodal: analizando la voz y el rostro del usuario. Cada modo consta de varios clasificadores cuya salida corresponde al estado emocional estimado. Finalmente una regla de decisión determina cual es la emoción real del usuario en función de la confianza que tiene en la estimación hecha por cada clasificador para esa emoción. Dentro de este módulo, conocido como GEVA (*Gender and Emotion Voice Analysis*), se realiza además de la identificación de emociones la identificación del género del interlocutor, detección de la actividad de voz (sistema VAD, capaz de distinguir voz humana de otro tipo de ruidos o sonidos) y un sistema experimental de clasificación de las expresiones sonoras de los perros.

- *Localización de usuarios*: este componente se encarga mediante el sistema auditivo del robot, formado por 8 micrófonos y diferencias en amplitud de la señal recibida por cada uno de ellos, de localizar el origen de la fuente sonora. La localización sonora se apoya en la localización mediante láser para determinar con mayor precisión dónde se encuentra el usuario. Con esta información de disposición espacial, más la información que se han obtenido previamente en experimentos proxémicos del robot con usuarios reales, el sistema de diálogo puede determinar la situación más adecuada del robot frente al usuario. Ver (Alonso-Martin, 2012a).

- *Sistema del tacto*: este componente es capaz de detectar cuándo una extremidad del robot ha sido tocada. Actualmente no es capaz de determinar la presión ejercida por el usuario ni exactamente el gesto con el que ha sido tocado.

- *Detección de poses del usuario*: este componente se encarga de determinar la pose, es decir de clasificar la posición del cuerpo del usuario de entre las posibles. Es capaz de determinar si una persona está sentada, de pie, señalando a la izquierda, derecha o al frente, entre otros. Este sistema usa una cámara estereoscópica de profundidad y mecanismos de aprendizaje automático.

- *Otros*: existen componentes que se encargan de facilitar la conexión de entrada de información por *tablet*, *smartphone* o *joypad*.

3. **Módulos de salida de información**: los componentes que se encargan de expresar o transmitir el mensaje suministrado por el gestor del diálogo (IDiM) al usuario y que trabajan con las salidas del sistema. Ver la Fig. 3.3. Estos componentes son:

    - *Sistema de Texto a Voz emocional (eTTS)*: se encarga de convertir texto a voz con emociones. El módulo de eTTS permite realizar tareas complejas, como gestionar la cola de luciones, traducir textos entre mas de 40 idiomas, además de adoptar distintos tonos de voz en función de los moto- res usados: *Loquendo*, *Festival*, *Microsoft TTS* y *Google TTS*.

    - *Sistema de generación de sonidos no verbales*: permite sintetizar sonidos no verbales en tiempo real, que permiten expresar al robot mensajes de manera similar a los producidos por la voz, pero en su propio lenguaje "robótico". Este tipo de sonidos se generan mediante el lenguaje de programación musical *Chuck*. "El submódulo de canto" permite que el robot sea capaz de cantar mediante el software musical *Vocaloid*. Para ampliar consultar (Alonso-Martín, 2012b).

    - *Generación de Lenguaje Natural (NLG)*: Este sistema de convertir información codificada en la computadora a lenguaje natural. En la práctica lo utilizamos para construir frases en tiempo real partiendo de una idea, o de un valor semántico. De esta manera conseguimos diálogos más variados y naturales. Por ejemplo si la idea a transmitir es la de "saludar" puede ser convertido a los siguientes textos: "Hola, estoy encantado de hablar contigo", o bien "Hola, amigo", entre otros. Este mecanismo de convertir valores "semánticos" a "texto enlatado" puede hacerse de diversas formas. En nuestro sistema se ha usado un modelo muy sencillo de gramáticas o plantillas, muy similar a los usados en el reconocimiento de voz.

- *Gestos expresivos*: permite al robot mediante sus extremidades (brazos, parpados, cabeza, cuello y base) realizar gestos típicos de cualquier diálogo, como son, negaciones, asentimientos, exclamaciones, incluso pasos de baile. Finalmente y aunque no es un componente propiamente dicho, el robot es capaz también de comunicarse mediante imágenes proyectadas en el tablet-PC, controlar componentes electrónicos mediante un mando infrarrojo, reproducir música descargada bajo demanda de Internet.

4. **Otros módulos**: además existen ciertos componentes presentes en el sistema de diálogo, que comunican con servicios web, que suministran entrada de información necesaria para el gestor de diálogo. Estos servicios usados son traductores automáticos (*Google Translate* y *Microsoft Translate*), buscador semántico (*Wolfram Alpha*) y reproductor musical *Goear*).

4 Principales características de RDS

Las principales características que describen al sistema aquí presentado son las siguientes:

- *Interpretado*: El sistema es interpretado, es decir se desacopla la especificación del diálogo concreto a cada situación (en ficheros de texto plano XML) de su interpretación y ejecución por el gestor del diálogo. Por lo tanto, podemos hablar de dos partes: la parte software, que es el propio Gestor de Diálogo IDiM, que ejecuta y interpreta los diálogos propiamente dichos. Y por otra parte, los diálogos quedan especificados en ficheros XML que establecen ciertos huecos de información a rellenar.

- *Adaptable*: el diálogo puede adaptarse a cada usuario en base a características estáticas y dinámicas. Las estáticas son almacenadas en perfiles de usuario, que son aprendidas durante la interacción mediante diálogo natural con el usuario: idioma, nombre, edad y huellas de voz. Los factores dinámicos del usuario como son la experiencia con el sistema, la emoción detectada (computación afectiva), la situación espacial respecto al robot (proxémica) también sirven para personalizar la interacción. Los factores dinámicos del propio robot, como es su estado emocional también puede ser tenido en cuenta por cada diálogo específico. Esta adaptabilidad también se refleja en el multilingüismo: el diálogo es capaz de llevarse a cabo en multitud de idiomas. Para ello está fuertemente basado en la precisión de los sistemas de traducción online.

- *Simétrico multimodal*: La interacción puede ser llevada por varios canales de entrada y salida de información. En este sentido, la multimodalidad se tiene en cuenta tanto a la entrada del sistema (fusión multimodal) como en las salidas (fisión multimodal). Los componentes del sistema de diálogo que lo convierten en simétrico multimodal han sido mencionados previamente.

- *Multisonido*: Por multisonido entendemos un sistema en el que la entrada y salida sonora no se limita única y exclusivamente al reconocimiento y síntesis de voz. La voz puede ser usada para más tareas, además de existir otros medios sonoros no basados en voz. Siguiendo esta definición, el sistema multisonido aquí propuesto es capaz de analizar el audio de entrada para: realizar localización sonora, clasificar la voz del usuario en emociones, detectar el nivel de excitación (*arousal)* del entorno, reconocer la voz, identificar al usuario. Es capaz de sintetizar sonidos para: generar voz con emociones, generar sonidos no verbales robóticos en tiempo real que aumentan notablemente la expresividad del robot, expresarse musicalmente mediante cante y finalmente la capacidad de reproducir de música online.

5 Conclusiones

En este trabajo se describió el sistema de interacción natural entre robots y humanos llamado RDS. Este sistema convive con el resto de sistemas que forman parte del sistema operativo que controla cada robot, AD-ROS. Se ha realizado una descripción general del sistema y de los componentes que lo forman, los canales/modos que gestiona el sistema así como sus principales características.

Referencias

Alonso-Martin, F., Salichs, M., 2011, "Integration of a voice recognition system in a social robot", Cybernetics and Systems, vol. 42, no. 4, pp. 215–245.

Alonso-Martin, F., Ramey, A., Salichs, M., 2011. "Maggie: el robot traductor". In UPM (Ed.), 9 Workshop RoboCity2030-II. Madrid: Robocity 2030. pp. 57–73

Alonso-Martin, F., Gorostiza, J., Malfaz, M., Salichs, M., 2012. "User Localization During Human-Robot Interaction". Sensors.

Alonso-Martin, F., Gorostiza, J., Malfaz, M., Salichs, M., 2012. "Musical Expres- sion in a Social Robot", in Proceedings of the 2012 International IEEE Intelligent Vehicles Symposium. Workshops V Perception in Robotics. Alcalá de Henares.

Barber, R., Salichs, M. 2002. "A new human based architecture for intelligent autonomous robots". Intelligent autonomous vehicles (IAV 2002): a proceedings volume from the 4th IFAC Symposium, Sapporo (Japan). Pergamon.

Corrales, A., Rivas, R., Salichs, M., 2009. "Integration of a RFID System in a social robot". Computer and Information Science, 44. pp 63–73.

Gonzalez-Pacheco,V., Ramey, A., Alonso-Martin, F., Castro-Gonzalez, A., Salichs, M., 2011. "Maggie: A Social Robot as a Gaming Platform". International Journal of Social Robotics, 3(4), 371–381. doi:10.1007/s12369-011-0109-8

Salichs, M., Barber, R., Khamis, A., Malfaz, M., Gorostiza, J., Pacheco, R., Rivas R., Corrales, A., Delgado, E., Garcia, D., 2006. "Maggie: A Robotic Platform for Human-Robot Social Interaction". IEEE Conference on Robotics, Automation and Mechatronics. pp. 1–7.

CAPÍTULO 3

SEMANTIC ACTION PARAMETER INFERENCE THROUGH MACHINE LEARNING METHODS

J. G. VICTORES, S. MORANTE, A. JARDÓN and C. BALAGUER

Robotics Lab, Universidad Carlos III de Madrid; jcgvicto@ing.uc3m.es

This paper presents the initial steps towards a robot imagination system, which aims at providing robots with the cognitive capability of imagining how a set of actions can affect a robot's environment, even if the robot has never seen the specific set of actions applied to its environment before. This robot imagination system is part of a human-inspired and goal-oriented infrastructure, which first learns the semantics of actions by human demonstration, and is then capable of performing the inverse semantic reconstruction process through mental imagery. A key factor in this system is distinguishing how different actions affect different features of objects in the environment. Simple probabilistic and other machine learning methods, tested to perform this first step of the inference process, are presented and compared in this paper. The inference of results of composed actions is generated as the sum of the contributions of each of the query word components. As an initial prototype, the actual learning process has been performed using synthetically created minimalistic environments as datasets, and a limited amount of training words for the learning process.

1 Introduction

Every day, humans perform thousands of actions. For robotic systems to be able help us with our daily life tasks, they must be endowed with recognition capabilities, storage, and reproduction of a great subset of actions to be performed in corresponding situations. The amount of available actions, their imprecision in their execution, and a strong dependence on the preferences of different end-users make the hard-coding of parameters an invalid solution of the general case. At most, one would expect to categorize actions to then evaluate their characteristics. This process of categorization has already been performed: languages classify different actions, binding different categories with different words (specifically, *verbs*). Obtaining knowledge on the mappings between words and instances from the world is often referred to as 'symbol grounding' (Harnad, 1990), or bridging the semantic gap.

In this paper we study the application of different machine learning algorithms to discover the 'grounding' of actions: the relation between words and their corresponding actions, and how these actions can modify the world. The 'grounding' part of this work is closely related to works in the field of action recognition, also known as direct action recognition (Poppe, 2010), and has also been targeted in the field of learning by imitation, also known as robot programming by demonstration (Calinon, 2010). These works are almost completely aimed at learning kinematics of the actions that humans perform, such as the trajectory of a human arm when reaching for an object, or the kinematics and dynamics of performing power grips. However, recent literature from the areas of neuro-science and psychology tends to indicate that the human brain encodes actions as end-goals related to the affordances of objects. For instance, when children imitate others grasping a person's ear, they tend to imitate the action goal (which ear to grasp) rather than the kinematic aspects of the action (which hand is used to perform the grasping) (Bekkering, 2000).

An attempt to learn goal-oriented tasks is shown in (Calinon and Guenter, 2005) where, despite they aim to learn the trajectory to perform an action, they also code some goals to be achieved, even in a distinct way than learned. A deeper understanding of object uses can be found in (Fitzpatrick, 2003), where the effects of pushing/pulling actions on objects to acquire affordances (what actions objects can 'afford') are learnt. A great variety of techniques has been used to try to learn object affordances, such as Self-Organizing Feature Maps (Cos-Aguilera, 2004), SVM (Dogar, 2007), and Bayesian Networks (Montesano, 2008).

Working in the line of learning about actions in a goal-oriented fashion, we focus on action recognition by detecting changes in the object involved. This means, for instance, recognizing a rotation action by noticing the change in the orientation of the object. Additionally, to analyze the effectiveness of the learning and providing the cognitive capability of robot imagination, we ask the system how the composition of these actions can affect the robot's environment, even if the robot has never seen the actions applied conjunctly to its environment before.

The rest of the paper is organized as follows: Section 2 outlines the proposed custom space, Section 3 explains different algorithms to generate action parameters, Section 4 compares the algorithm presented, and Section 5 discusses about limitations and deficiencies providing several conclusions.

2 Semantic Grounding Database

The semantic grounding database is represented and stored as an incremental set of tagged points in an n-dimensional feature space (see Fig. 1), denoted F. Every time an action is performed, we measure the variation of the values of the features we extract from the object we are interested in, and create a new point in F using these variations as coordinates.

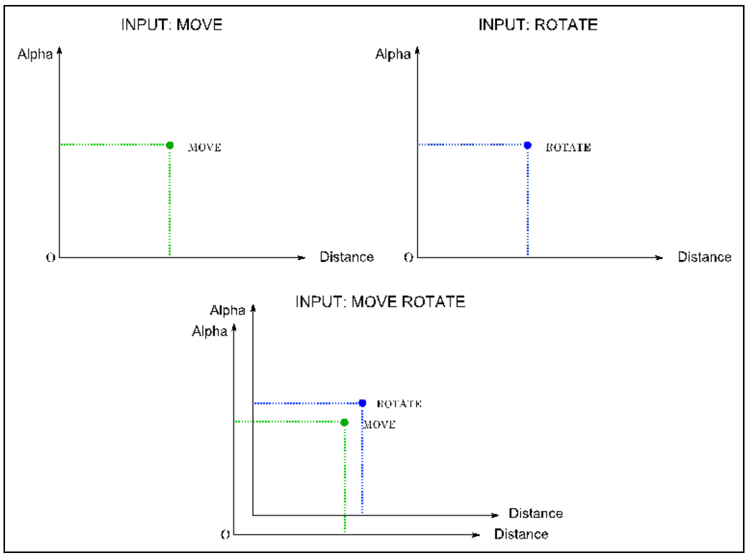


Fig. 1. Semantic Grounding Space example where n=2

This new point is tagged with the name of the action, thus enhancing the point with a semantic label that will be used for posterior inferences. The semantic label is a single word. If more than one word is input, overlapping points with the different words as tags (i.e. 'move rotate' becomes 'move' and 'rotate').

For the experiments presented in this paper, we have fixed the dimensions to three (X, Y and ALPHA), which represent the translation and the rotation of an object in a 2-dimensional space. As semantic input we consider only two words: one for representing a pure translation movement, 'move', and another one for representing a pure rotation movement, 'rotate'. As stated, the grounding process is repeated for each sample, until the database has been populated with a sufficient amount of samples. We will start from this populated situation to test our algorithms.

3 Models of Study

The aim of presented algorithms is to obtain coherent and valuable parameters for the corresponding semantic input we introduce. The methods presented cover the most intuitive approximation, starting with a simple Arithmetic Mean of the samples with the required tag. We also test a Neural Network system, specifically trained to work in an inverse way to its conventional use. We construct a Gaussian Mixture Model and extract the representative parameters. Finally, a Support Vector Machine modified to work as a regressor is also tested.

3.1 Arithmetic Mean Model

The Arithmetic Mean (AM) model is a relatively naive approach. Upon receiving a pair <word, [x, y, alpha]>, the Look-Up Table of each element of the output layer is updated with its corresponding mean. A dependency graph is depicted in Fig. 2.

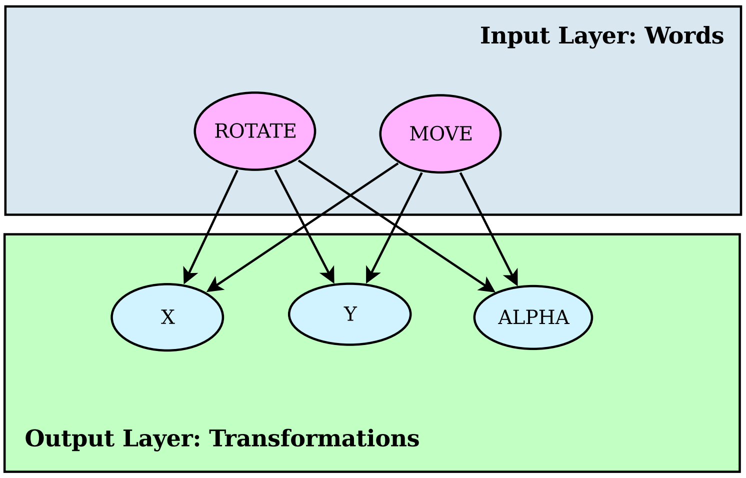


Fig 2. Arithmetic Mean Model fully connected dependency graph

An advantage of the AM is that its incremental form for performing online learning is well-known and simple to implement, as in Equation (1).

$$A_{n+1} = A_n + \frac{v_{n+1} - A_n}{n+1} \tag{1}$$

3.2 Neural Network Model

A classical three-layered fully connected Feed-Forward Neural Network (NN) model has been used for this application, as seen in Fig. 3. Its number of input perceptrons has been set to 2 (corresponding to the 2 input words), its number of output perceptrons has been fixed at 3 (corresponding to the 3 used features), and 50 has been the number of perceptrons selected for the hidden layer implementing a sigmoid function. The NN is trained by back-propagation, a supervised learning method, set at 10 epochs. The number of perceptrons in the hidden layer has been empirically determined, and offers a tradeback with the number of epochs necessary to achieve a certain behavior.

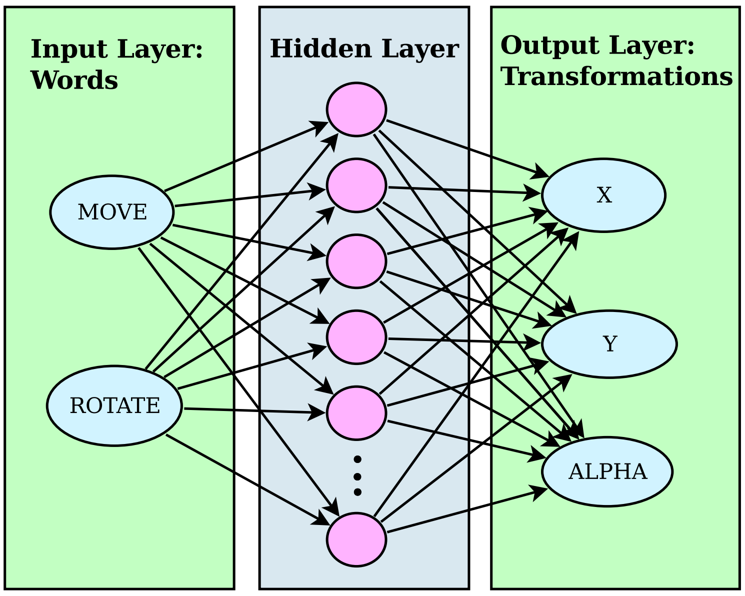


Fig. 3. Three-layered fully connected Feed-Forward Neural Network model

It is important to notice that these NN are used in an inverted way when compared to their habitual use. Instead of classifying a combination of inputs into a delimited number of classes, we use the class as the network input, and the values belonging to the class as the network output. The resulting network we achieve with this method is a kind of 'pseudo-inverse NN', which allows us to interpolate between intermediate values of the cloud of samples.

3.3 Gaussian Mixture Model

The Gaussian Mixture Model (GMM) represents the probability distribution (density estimation) of the points. We set the number of mixtures to one (formally K=1 or M=1), and feed the GMM with all the samples with the same word. The values used are the Gaussian means.

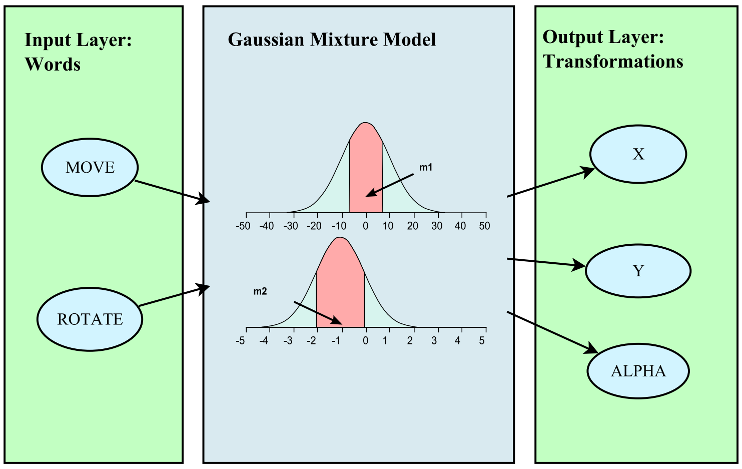


Fig. 4. Gaussian Mixture Model

Because the use of a single component, the GMM actually corresponds to a single multivariate Gaussian model. The only component can be summed out of the general weighted GMM expression of Equation (2).

$$p(x|\lambda) = \sum_{i=1}^{M} w_i g(x|\mu_i, \Sigma_i) \tag{2}$$

This results in a single (unit weight) component expressing the multivariate Gaussian probabilistic density function of Equation (3).

$$g(x|\mu_i, \Sigma_i) = \frac{1}{(2\pi)^{D/2} |\Sigma_i|^{1/2}} exp \left\{ -\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1} (x-\mu_i) \right\} \tag{3}$$

The actual fitting of the parameters to the data is performed using the Expectation-Maximization (EM) algorithm, an iterative method for performing Maximum Likelihood Estimation (Reynolds, 2008).

3.4 Support Vector Regression (SVR)

The Support Vector Regression method (Fig. 5) is a modified version of a Support Vector Machine (SVM). SVM are normally used in classification problems, but SVR are adapted to perform regressions.

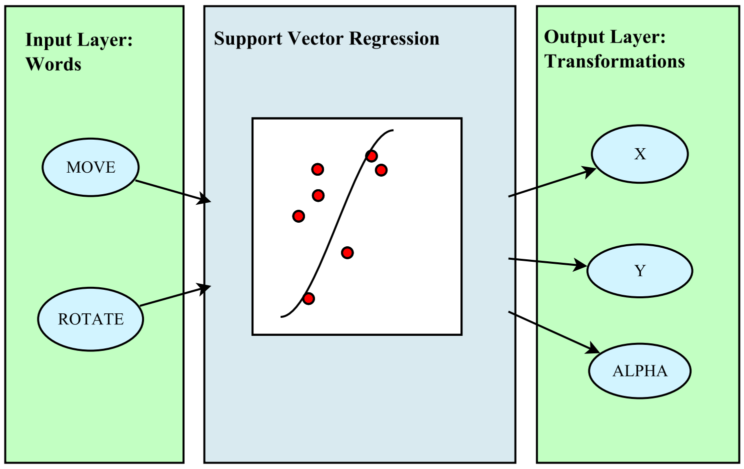


Fig. 5. Support Vector Regression

We use a non-linear regression version, which uses a kernel function that transforms the data into a higher dimensional feature space in order to perform a linear separation. The kernel function is a Radial Basis Function, seen in Equation (4).

$$K(\mathbf{x},\mathbf{x}') = \exp\left(-\frac{||\mathbf{x}-\mathbf{x}'||^2}{2\sigma^2}\right) \tag{4}$$

This process is called ‘Kernel trick’ and it is used to map samples into other spaces, hoping that samples will gain linear structure in the new space. In this new space, SVM uses ‘epsilon intensive loss’ to estimate the quality of estimation. This function considers the residuals in two ways: if residual is less than epsilon, then there is no loss, if it is bigger, then, an amount of loss is added. The main idea of SVR is performing a linear regression in the higher dimension space, with the epsilon intensive function.

4 Experiments for Comparison of Models

To compare the presented models, we train each with the same experimental dataset. The dataset is composed by 5 ‘move’ sequences, and 5 ‘rotate’ sequences. The sequences are composed by 7 frames (100 by 100 pixel black and white images), such as those seen in Fig. 6. The contents of a ‘move’ sequence is a rectangle that advances 60 pixels on X and Y while maintaining its rotation. Each ‘rotate’ sequence depicts a 60º angle rotation while maintaining its position.

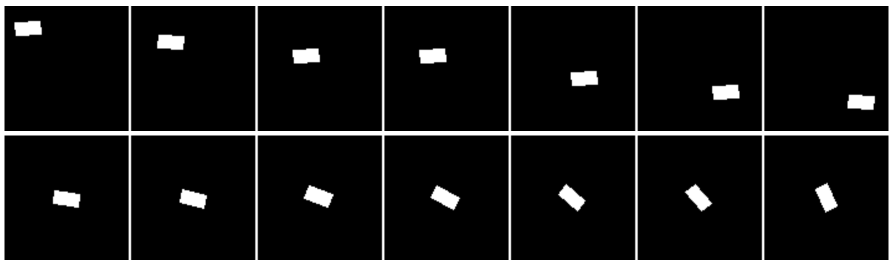


Fig. 6. Image sequences: a) move, b) rotate

An additional 1% standard deviation noise has incorporated to each feature (X, Y and ALPHA) to emulate camera perturbation effects and inefficiencies during segmentation. Note that this noise may significantly affect the behavior of the algorithms, as the training dataset is relatively small. The programming language used was Python, mostly using the tools provided by Scikit-Learn (Pedregosa, 2011).

Table 1 and Table 2 depict the outputs of the different tested algorithms assigned to the ‘rotate’ and ‘move’ word, respectively. For compactness, the value at the right of the ± operator represents the standard deviation.

Table 1. Table for ROTATE values

Employed techniques	X	Y	Alpha
Arithmetic Mean	0	-0.2	65.62
Neural Network (*)	0.05 ± 0.26	-0.12 ± 0.1	64.83 ± 0.22
Gaussian Mixture Model (**)	0 ± 1.41	-0.2 ± 0.74	65.62 ± 2.81
Support Vector Regressor	0	-0.1	64.55

Table 2. Table for MOVE values

Employed techniques	X	Y	Alpha
Arithmetic Mean	59.8	59.6	-0.26
Neural Network (*)	59.75 ± 0.2	59.56	-0.74 ± 0.43
Gaussian Mixture Model (**)	59.8 ± 0.74	59.6 ± 1.35	-0.26 ± 2.8
Support Vector Regressor	59.9	59.1	0.71

It is important to notice the difference in meaning of the standard deviation of the NN algorithm and the GMM algorithm. The NN algorithm's standard deviation is marked with a (*) because the output of the NN is stochastically variable. This is due to the fact that the network's weights are set randomly at initialization. The table values of the NN are the averaged values of running the algorithm 5 times. The GMM algorithm's standard deviation (**), however, describes the dispersion of the input data, a fixed parameter with an accuracy given by the estimation based on the maximization of the likelihood of how the distribution expresses the original training data. It is computed as the square root of the diagonal of the estimated covariance matrix.

The final Root-Mean-Square (RMS) errors on composition are illustrated in Table 3. They have been computed as the direct sum of the values found on Table 1 and Table 2, setting the target values to X=60, Y=60 and ALPHA=60º (which is the intuitive composition of 'rotate' and 'move' a human could deduce from the input). This table represents a final metric on the accuracy of the algorithm, while obviating other metrics or advantages which will be considered in the following final section.

Table 3. Table for MOVE and ROTATE composition errors (RMS)

Employed techniques	RMS
Arithmetic Mean	5.403
Neural Network	4.97 ± 0.45
Gaussian Mixture Model	5.403
Support Vector Regressor	5.367

5 Conclusions

The bases for future research on robot imagination systems of actions and action compositions using algorithms from the field of machine learning have been set. The following points review some of the benefits and drawbacks of the studied algorithms on this specific minimalistic dataset:

- The Arithmetic Mean offers a fast and accurate solution. Additionally, the possibility of easily implementing its incrementally learning version will surely be attractive to many readers. However, it lacks a certain degree of flexibility and deep philosophical interpretations that may be found abundance within other algorithms.

- Our Neural Network implementation has proved to work well, even having being trained inversely. It is the most and perhaps only bio-inspired algorithm of the ones tested. This is also manifest in the fact that its results are not always the same, much like in a human's actions. Moreover, it offers a quite unique possibility of simultaneously activating several inputs. These results were, however, omitted in the experiment section due to bogus results: output more similar to the mean than to the sum of actions.

- We consider the Gaussian Mixture Model the most attractive option. The results are precise as with the Arithmetic Mean, but may be distributed with a standard deviation similar to that of the input values if desired. This can be important if, for example, the algorithm determines a great degree of dispersion is a representative characteristic of a certain action. Additionally, several algorithms which have not been used here can be used to perfect the model. For example, Bayesian Information Criterion (BIC) or Akaike Information Criterion (AIC) may be used to determine an optimal number of Gaussian components other than K=1. Furthermore, recent studies indicate Bayes-Optimal estimators may achieve higher performance than methods based on maximization of likelihood.

- The Support Vector Regressor algorithm has been proved to be a valid method in terms of computation cycles and accuracy. However, the authors consider that further study is required in order to unveil its full potential.

Moreover, the authors would like to transmit a final question to the robotic scientific community: “How are robots going to transform the world if they can't even imagine how?”

Acknowledgements

The research leading to these results has received funding from the ARCADIA project DPI2010-21047-C02-01 funded by CICYT project grant on behalf of Spanish Ministry of Economy and Competitiveness.

References

Bekkering, H., Wohlschlager, A., and Gattis, M. 2000. Imitation of gestures in children is goal-directed. The Quarterly Journal of Experimental Psychology: Section A, 53(1), 153–164.

Calinon, S., D’halluin, F., Sauser, E., Caldwell, D. and Billard, A. 2010. Learning and reproduction of gestures by imitation. IEEE Robotics and Automation Magazine, vol. 17, no. June, pp. 44–54.

Calinon, S., Guenter, F., and Billard, A. 2005. “Goal-directed imitation in a humanoid robot. In Robotics and Automation”. ICRA 2005. Proceedings of the 2005 IEEE International Conference on (pp. 299–304). IEEE.

Cos-Aguilera, I., Hayes, G., and Cañamero, L. 2004. Using a SOFM to learn object affordances. In Procs 5th Workshop of Physical Agents (WAF'04). University of Edinburgh.

Dogar, M. R., Cakmak, M., Ugur, E., and Sahin, E. 2007. From primitive behaviors to goal-directed behavior using affordances. In Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on (pp. 729–734). IEEE.

Fitzpatrick, P., Metta, G., Natale, L., Rao, S., and Sandini, G. 2003. Learning about objects through action-initial steps towards artificial cognition. In Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on (Vol. 3, pp. 3140–3145). IEEE.

Harnad. S. 1990. The symbol grounding problem. Physica D: Nonlinear Phenomena, 42(1):335–346.

Montesano, L., Lopes, M., Bernardino, A., and Santos-Victor, J. 2008. Learning Object Affordances: From Sensory-Motor Coordination to Imitation. Robotics, IEEE Transactions on, 24(1), 15–26.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. 2011. Scikit-learn: Machine learning in Python. The Journal of Machine Learning Research, 12, 2825-2830.

Poppe, R. 2010. A survey on vision-based human action recognition. Image and vision computing, 28(6), 976–990.

Reynolds, D. 2008. Gaussian mixture models. Encyclopedia of Biometric Recognition, 2(17.36), 14-6.

CAPÍTULO 4

COGNITIVE MODEL FRAMEWORK FOR LEARNING AND ADAPTATION OF ROBOT SKILLS TO TASK CONSTRAINTS

D. HERNÁNDEZ GARCÍA[1], C. MONJE[1] and C. BALAGUER[1]

[1]Robotics Lab, Universidad Carlos III de Madrid

The humanoid robots of the future are required to have advanced motor control skills, comprehensive perceptual systems, and a suitable level of intelligence. Developing this kind of robots requires the development of intelligence thinking at the system internal processing, centred on an organization of intelligence in terms of the configuration and interaction of cognitive models. In this work a framework is proposed for a cognitive model for the generation and adaptation of learned models of robot skills for complying with task constraints. The proposed framework is meant to allow: (a) for an operator to teach and demonstrate to the robot the motion of a task skill it must reproduce. (b) To build a database with the knowledge of the learned skills allowing for its storage, classification and retrieval. (c) To adapt and generate learned models of a skill, to new context, for compliance with the current task constraints.

1 Introduction

A major goal in robotics research is to develop human-like robotic systems capable of interacting and collaborating with humans. Humanoid robots are particularly suitable for these duties because they are able to interact with the environment using the same tools designed for humans, (Ambrose, 2000). Since humanoid robots are designed to resemble a human shape and to poses human capabilities, they would be ideally suitable for performing tasks and to safely share the same space and activities with people.

Working alongside humans means dealing with continuously changing environments and a huge variability of tasks, thus the robots should have the ability to continuously learn new skills and adapt the existing skills to new contexts. The ultimate goal is for a robotic platform capable of performing, autonomously, in the unstructured scenario of humans natural environment. Humanoid robots must realize any number of tasks which could be reasonably expected from them by its human operators, during the normal development of a typical workday.

Great advances have been made in humanoid robotics research. There currently exist robots that walk, run or climb stairs; that can handle and manipulate objects, or carry heavy loads, etc. However, despite all advances, the ultimate goal of an intelligent and autonomous humanoid robot companion is still far from reach. Future humanoid robots would need to execute a wide range of movements in a natural human-like manner, process information from multiple sensors into a reliable representation of the world in order to understand and react to their environment. Also, they would need to provide means for a meaningful interaction with their human partners; they must be engaging and responsive. And they must present intelligent, natural, predictable and reasonable behaviours.

2 Intelligent Humanoid Robots Architectures

Development of intelligent systems is a long term goal in the fields of robotics research, artificial intelligence and cognitive science. To truly exploit humanoid robots full potentiality it would be necessary to provide them with an intelligence that is similar to humans. This presents an even greater challenge than endowing humanoids with the ability to replicate human-like motions or simulate human interactions. Particularly since the process of human intelligence is one that is not fully understood, in which many competing ideas can be found and where no generally accepted theory of intelligence exists that satisfies every group.

In (Albus, 1991) intelligence is defined as the ability of a system to act appropriately in an uncertain environment, where appropriate action is understood as those that increases the chances of success for the behavioural goal and subgoals. For (Poole, et al., 1998) an intelligent agent is one that is flexible to changing environments and changing goals, learns from experience, and makes appropriate choices given perceptual limitations and finite computation. For (Legg, et al., 2006) intelligence measures an agent ability to achieve goals in a wide range of environments.

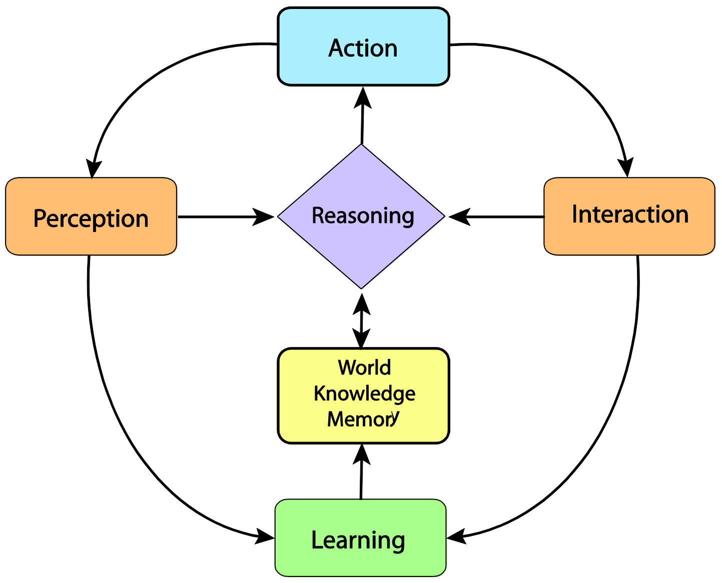


Fig. 1. Model for an architecture of intelligent agents. Systems for perception, action, interaction, reasoning, world knowledge and learning are needed.

(Albus, 1991) proposed a model that integrates knowledge from research in both natural and artificial systems. The model consists of hierarchical system architecture. Different levels of intelligence in the hierarchy can be achieved. A minimal level of intelligent requires at least the ability to sense the environment, make decisions and take actions. Higher levels of intelligence may include the ability to recognize objects and events, to represent knowledge in a world model and to reason about and plan for the future. More elevated forms of intelligence provide the capacity to perceive and understand, to choose wisely, and to act successfully under a large variety of circumstances, (Albus, 1991). The current humanoid robots may only be around the minimum and mid-levels of intelligence. As developments of systems, architectures and algorithms continue to advance the intelligent capabilities of humanoid robots would increase. Humanoid robots need to reach a functional level of intelligence that allows them to function properly interacting with humans and the environment. Humanoid robots need to achieve a sufficiently high level in the hierarchy as to be considered by its human partners as intelligent, namely, a sensing, acting system that perceives, learn, plan, and succeed in achieving its goals in the world. This is a major challenge in humanoid robot research.

The model in (Albus, 1991), identifies four elemental systems of intelligence: sensory processing, world modelling, behaviour generation, and value judgement. Similarly, (Langley, et al., 2009) identified the different functions of cognition as perception, learning, motor control, reasoning, problem solving, goal orientation, knowledge representation and communication. Fig. 1 illustrated a model of an architecture for an intelligent agent.

2.1 Robot Planner-Based Architectures

Efforts in building intelligent robot systems first concentrated in the fundamental modules for perception, reason, and action. Under this view the classical approach from AI emerged, decomposing the control systems for autonomous robots into the three functional elements forming the sense-plan-act cycle. The sensing system translates raw sensor input into a world model. The planning system takes the goals and the world model and generates plans to achieve those goals. The execution system generates the actions prescribed by the plan, (Gat, 2009).

Classical AI approaches focused their efforts on building intelligent systems on the symbolic representation of the physical world entities. And in the believe that intelligent agents could be formulated as information processing systems, taking a representation of the world as input and outputting appropriate set of actions. The development of planning or deliberative strategies, that generates the sequences of tasks to accomplish robot goals, is the central aspect of the classical AI control architectures. Work in hierarchical planner-based or deliberative architectures focused on planning long-term actions. The system intelligence, is said to live, in the planner or the programmer, not the execution mechanism, (Gat, 2009).

The planner-based architecture presented its pros and its cons. They can handle complex tasks by breaking them into more manageable sub task, (Simmons, 1994). They allow for explicitly formulating task and goals of the system and estimating the quality of the agents performance, (Mataric, 1997). And they can produce optimal, domain-independent solution. However, they generally fail to address uncertainty, and are therefore unfit to operate in changing environments, since they are unable to re-plan its actions quickly enough. Planner-based approaches have high computational costs.

2.2 Robot Behaviour-Based Architectures

The ideas that real intelligence is situated in the world and that the intelligence behaviours can only emerge as a result of an agent interaction with the environment, gained preference. This novel AI approach was based on the hypothesis that to build intelligent systems it is necessary to have its representations grounded in the physical world, (Brooks, 1990). Instead of focusing on the design of systems capable of intelligent thinking, the emphasis changed to creating agents that could act intelligently. The behaviour-based or paradigm is founded on the building of behaviours, direct couplings of sensory inputs to a pattern of actions that in turn carries out a specific task (Murphy, 2000).

Reactive architecture is that it does not include any kind of central symbolic world model, and does not use complex symbolic reasoning, (Wooldridge, et al., 1995). The behaviour-based paradigm presents a direct coupling between perception and action. A collection of preprogrammed condition-action pairs is embedded into the agent control strategy. The behaviours in behaviour-based architectures typically consist of a collection of rules, taking inputs from sensors or other behaviours in the system, and sending outputs to the effectors, or other behaviours, (Nicolescu, et al., 2002).

The behaviour-based approaches presented greatly improved performances in robot navigation and obstacle avoidance. They also showed great flexibility and adaptability, and were ideally suitable for performing in dynamic and unpredictable environments. Also, behaviour-based approaches were robust, simple and computationally tractable. However, they also have some drawbacks. Behaviours-based architectures do not include explicitly the achievement of a goal in their behaviour description. Behaviour-based approaches only include `local' information, collected form the environment, they present a short-term view, with no long-term planning capabilities, and offered limited applicability.

2.3 Robot Hybrid Architectures

The hybrid deliberative/reactive architectures emerged as attempts to bridge the two approaches and use the strengths of each other in reducing their respective shortcomings. In practice, this meant the integration of the planning aspect of the hierarchical deliberative paradigm with the rapid execution capabilities of the reactive paradigm, (De Silva, et al., 2008).

The hybrid architectures idea was to attempt a compromise between the purely reactive and deliberative approaches and integrated both of them as subsystems of the architecture. Hybrid architectures usually divide the control system into a layered structure. The architecture needs the integration of three separate components: reactive feedback mechanism for controlling low-level primitive activities; deliberative planning system for decision-making computations; and sequencing system controlling the interactions between the other two components.

Hybrid architectures presented advantages over both purely deliberative and purely reactive architectures. Hybrid architectures combined the rapid real-time responses and ability to adapt to quickly changing environments provided by behaviour-based systems with the higher level reasoning, planning and decision making capabilities of planner-based approaches, enabling them to perform in a wider range of tasks, coupling the strengths of both paradigms, providing more successful intelligent agents. Anyway, these types of architectures are not devoid of problems and critics. Hybrid deliberative/reactive architectures tend mostly to be very specific, application dependent, and lacking general design guiding methodologies.

2.4 Robot Cognitive Architectures

When thinking about building robots with human level intelligence and functionality, the agents architecture structural paradigm shifts from the production and emergence of intelligent behaviours as a system output, towards a viewpoint whose main pursuit is in the development of intelligence thinking at the system internal processing, centred on the mechanism that allows for the generation of thought and the interior workings of cognition. This calls for an organization of intelligence in terms of cognitive models.

Cognitive architectures specify the underlying infrastructure for an intelligent system. A cognitive architecture is an organizational structure, of knowledge representations and functional structures, set for enabling the modelling of cognitive phenomena, (Albus, et al., 2005).A cognitive architecture would attempt to provide the basic primitive computational resources needed for developing intelligent systems. Amount their properties are those related to memory, representation, processing, organization, performance, interaction, reasoning, and learning. Research on cognitive architectures is a very important topic since it supports a central goal of artificial intelligence, cognitive science, and robotics, the creation and understanding of agents build for supporting the same capabilities as humans (Langley, et al., 2009). The cognitive architectures must allow for the execution of skills and actions in the environment. The architecture must be able to represent and store motor skills that enable the agent's activity.

Efforts in cognitive architectures have produced important advances in cognition, reasoning and conceptual aspects of human thinking. (Levesque, et al., 2008) offers an overview of the challenges and efforts taken in the subject of cognitive robotics. Further research in cognitive architectures, frameworks and cognitive models, is important to improve the control and design of the intelligent robotic agents. The most obvious arena for improvement concerns the introduction of new capabilities, and additional research on the structures and processes that support such capabilities, (Langley, et al., 2009), which bear the wide range of human skills and cognitive abilities.

Table 1. Architectures for Humanoid Robots

Architectures	Planner	Behaviour	Hybrid	Cognitive
Design Paradigm	Sense-Plan-Act cycle	Hierarchy of coupled sense-act behaviours	Low-level reactive layers and high-level deliberative layers	Interconnected structure of functional cognitive modules
Strengths	-Planning of long term actions. -Breaks complex task into subtask. -Can produce optimal, domain-independent solutions.	-Improved navigation and obstacle avoidance. -Great efficiency at run-time. -Robust, simple and computationally tractable.	-Combine real-time response and adaptability of reactive systems with planning and decision making of deliberative approaches.	-Solid basis for building intelligent systems. -Interconnected models of cognitive abilities support range of skills and actions.
Challenges	-Fail to address uncertainty. -High computational cost. -Poor performance when frequent replanning.	-No long-term planning. -Limited applicability. -Difficult to debug and understand emerging behaviour.	-Very application dependant. -Lack general design, methodologies. -Difficult to generalize in varying domains.	-Introduce new capabilities. -Address agent physical limited resources. -Experimental methods to evaluate architectures.
Implementations	STRIPS, IRMA, etc.	Subsumption	ATLANTIS, SSS, 3T, AuRA, etc	Soar, ICARUS, ACT-R,EPIC
Applicability Humanoid Robots	Unfit to operate in changing environments	Offered limited applicability confined to low-level tasks	Couples strengths of deliberative/reactive paradigms. Lacks of good theoretical models	Support goal for intelligent artificial systems. Further research is important

In Table 1 the most relevant aspects and shortcomings of the intelligent architecture approaches for developing humanoid robots are summarized.

3 Cognitive Model of Learning and Adaptation of Skills to Task Constraints

The humanoid robots of the future, capable of working autonomously and serving humans, required of them to have advanced motor control skills, comprehensive perceptual systems, and suitable intelligence. With an intelligent agent being understood as in (Poole, et al., 1998), as one that is flexible to changing environments and changing goals, learns from experience, and makes appropriate choices given perceptual limitations and finite computation. The previous sections presented a review of different approaches for developing a robot's functional architecture that would endow them with the capabilities for performing intelligent behaviours in the environment. Clearly this is a very challenging topic in which it has not yet been reached completely satisfactory solutions, although great efforts and advances have been made over the years obtaining important contributions through the field. The deliberative planning approaches, while applicable for state-space search and scheduling systems, proved to be unfit to operate in changing environments as would be require of humanoid robots. The behaviour-based approaches presented great performances in robot navigation and obstacle avoidance, and in dynamic and unpredictable environments, yet their true applicability is limited to low-level behaviours and they would not be suitable to deal with the complexities of behaviours present in humanoid robots. The hybrid approaches attempted to combine the strengths of deliberative and reactive approaches and can be readily employed as the system architecture for several robotic platforms, however when the focus of research goes to building humanoid robots, designed to present a full human level intelligence, different mechanism are necessary to replicate the complex level of skills and operations presented by humans. The agent’s architecture paradigm shifts from the production and emergence of intelligent behaviours as a system output, towards a view in the development of intelligence thinking at the system internal processing, centred on an organization of intelligence in terms of the configuration and interaction of cognitive models. Research in cognitive architectures constitute a solid basis for building intelligent systems, but even though some attempts on the field has been made for providing cognitive process for humanoid robots, there are not fully developed cognitive architectures capable of endowing robots with the needed functional intelligence readily available.

Humanoid robot agents to be successfully employed working alongside human partners would need to address important challenges such as high-level understanding, engaging interactions and quick adaptations to environmental dynamical changes, (Stoytchev, et al., 2001). The ability to self-adapt and learn from experience is a major concern. In order to have humanoid robots acting fluently in the world, interacting with different objects and people, robots must be able to learn and adapt its motor control to dynamic changes in its interaction with the world, robot systems must be continuously self-adapting, (Brooks, 1996). From all the above it is clearly understood that humanoid robots in order to cope with working in continuously changing environments and performing a huge variability of tasks, must be provided with systems that allows them to continuously learn new skills and adapt its existing skills to new contexts, as well as to robustly reproduce its behaviours in a dynamical environment. Research efforts must focus on building the necessary models of cognition that would allow assembling the levels of intelligence.

In this work a framework is proposed for a cognitive model for the generation and adaptation of learned models of robot skills for complying with task constraints, motivated on the design of multi-layered reference model architecture in the spirit of (Albus, 1991), and influenced by the ideas of Dynamical Systems approaches to embodied cognition, as promoted by (vanGelder, et al., 1995), (Clark, et al., 1999), (Clark, 2004), (Beer, 2000). And in the Robot Programming by Demonstration approaches for encoding complex motions as Dynamical Systems first introduced by (Ijspeert, et al., 2001), (Ijspeert, et al., 2002), representing movement plans as mixtures of non-linear differential equations with well-defined attractor dynamics. Following a view which claims that models of cognition must be embodied processes capturing the unfolding of cognition in time, mindful of the associated sensory and motor surfaces embedded in the environment in which cognitive phenomena takes place, for (Schöner, 2008). And that a systems internal representations may be modelled not as simple inner states but as dynamical patterns of just about any conceivable kind, (Clark, 2004). Here, thought can be described by variables governed by a set of non-linear differential equations and an agent behaviour can be generated from the complex dynamical evolution of stable states and their instabilities in a non-linear dynamical system, (Schöner, 2008).

The proposed framework is meant to allow:

- For an operator to teach and demonstrate to the robot the motion of a task skill it must reproduce.
- To build a database with the knowledge of the learned skills allowing for its storage, classification and retrieval.
- To adapt and generate learned models of a skill, to new context, for compliance with the current task constraints.

Our propose framework is illustrated in Fig. 2.

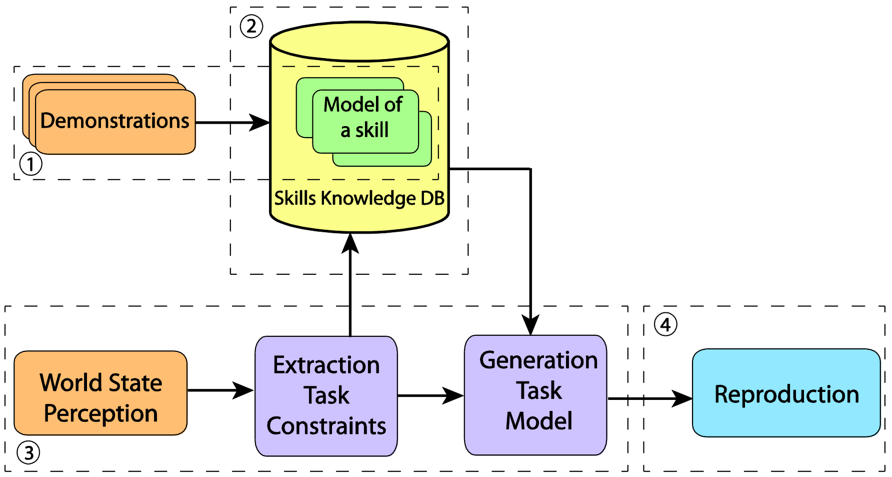


Fig. 2. Propose framework for a cognitive model for adaptation of learned models of robot skills for complying with task constraints. A database of the skills knowledge (2) is built with the models of the skills learned by demonstrations (1). The constraints of a requested task are extracted from the perception of the world state. With the current task constraints and the models of a skill retrieved from the knowledge database an adapted task model (3) is generated for reproduction (4).

4 Learning Models of a Skill from Demonstrations

The robot skills motions would be learned as a time independent model of the motion dynamics estimated through a set of first order non-linear multivariate dynamical systems, as in the proposed approach of (Ijspeert, et al., 2002). To teach and learn the robot skills a Robot Programming by Demonstration (RPbD) or Learning from Demonstration (LfD) framework is implemented. Adopting an imitation learning approach provides intuitive and user-friendly methods to teach tasks to a robot by demonstrating the skills, and they don't require for the user to have expert programming skills.

Adopting non-linear dynamics systems theory has become an increasingly accepted practice in several branches of sciences. The field of neural control of movement has long suggested to model movement phenomena with dynamical systems, (Kelso, 1995). Encapsulating the dynamics of the movement into a dynamical system encoding is a promising approach to learning movement trajectories, (Billard, et al., 2008).

A Dynamical Systems (DS) approach to skill learning can offer a fast, simple and powerful formulation for representing and generating movement plans, learned from demonstration. The DS framework allows complying with the attractor dynamics of the desired behaviour, modulating it with a set of non-linear dynamic systems that form an autonomous control policy for motor control. Statistical learning techniques can be used to arbitrarily shape the attractor landscape of the control policy for encoding within the desired trajectory, going from an initial state to an end state driven by the attractor dynamics. DS are intrinsically robust and can adapt its trajectories instantly in the face of spatio-temporal perturbations, (Khansari, et al., 2010). The DS do not explicitly depend on time indexing and provide closed loop control and are able to model arbitrary non-linear dynamics, (Gribovskaya, et al., 2010). The DS can also be easily modulated to generate new trajectories that have similar dynamics, performing in areas that were not covered during demonstrations, (Khansari, et al., 2011).

4.1 Stable Estimator of Dynamical Systems

(Khansari, et al., 2011) proposed a learning method, called Stable Estimator of Dynamical Systems (SEDS), to learn the parameters of the DS that ensure for all motions to closely follow the demonstrations dynamics. The approach formulates the encoding as a control law that is driven by a first-order autonomous non-linear ODE with Gaussian Mixtures.

The state of the robotic system ξ is assumed to be governable by an autonomous dynamical system, with a single equilibrium point. And the set of N-dimensional demonstrated data points to be represented as $\{\xi^i, \dot{\xi}^i\}_{i=1}^{D}$. A probabilistic framework is employed to build an estimate $\hat{f}$, of the non-linear state transition map f, based on the set of demonstrations. The dynamics of the motion are learned thus by modelling the estimate $\hat{f}$ via a finite mixture of Gaussian functions, f is define as a non-linear combination of a finite set of Gaussian kernels using the GMM, (Gribovskaya, et al., 2010). A mixture model of K components is defined by a probability density function,

$$p(\xi) = \sum_{k=1}^{K} p(k)p(\xi|k) \qquad (1)$$

where ξ is a data point, $p(k)$ is the prior probability and $p(\xi|k)$ is the conditional probability. The GMM computes a joint probability density function for the input and the output so that the probability of the output conditioned on the input are a Mixture of Gaussian.

The GMM define a joint probability distribution $p(\xi^i, \dot{\xi}^i)$ of the training set of demonstrated trajectories as a mixture of the K Gaussian multivariate distributions N^k, with π^k, μ^k, and Σ^k, respectively the prior, mean and covariance matrix, parameters of the Gaussian component k. Therefore the parameters in Eq. 1 become,

$$p(k) = \pi^k$$
$$p(\xi|k) = N(\xi;\, \mu^k, \Sigma^k) \qquad (2)$$

The mixture of Gaussian functions would estimate the non-linear function f, thus the unknown parameters θ of f, are define by the prior, π^k, the mean, μ^k, and the covariance matrix, Σ^k, of the K Gaussian functions, such that $\theta = \{\pi^k, \mu^k, \Sigma^k\}$.

To generate a new trajectory from the GMM, one then can sample from the probability distribution function $p(\xi, \dot{\xi})$, this process is called Gaussian Mixture Regression (GMR). So it is possible after training, to recover the expected output variable $\hat{\dot{\xi}}$, given the observed input in ξ. This process is called Gaussian Mixture Regression. The GMR process takes the conditional mean estimate of $p(\dot{\xi}|\xi^*)$, the estimate of our function can be expressed by,

$$\hat{\dot{\xi}} = \sum_{k=1}^{K} h^k(\xi^*)\left(\Sigma^k_{\dot{\xi}\xi}(\Sigma^k_{\xi})^{-1}(\xi^* - \mu^k_{\xi}) + \mu^k_{\dot{\xi}}\right) \qquad (3)$$

$$where, h^k(\xi) = \frac{p(\xi; \mu^k, \Sigma^k)}{\sum_{k=1}^{K} p(\xi; \mu^k, \Sigma^k)}$$

$$with\ h^k(\xi) > 0\ and \sum_{k=1}^{K} h^k(\xi) = 1$$

The process for encoding the dynamics of a motion through GMM, and GMR, is illustrated on Fig. 3.

To build a globally asymptotically stable DS a set of sufficient stability conditions is established. It is need then to determine the unknown parameters, $\theta = \{\pi^k, \mu^k, \Sigma^k\}$, of the estimate of f, such that, by starting the motion from any point in the state space the energy of the system decreases until it reaches the target. Learning the parameters of the GMM proceeds as a constraint optimization problem, ensuring that the model satisfy global asymptotic stability of the DS at the target, (Khansari, et al., 2010). For the optimization objective a function based in the log-likelihood, SEDS-likelihood, is used as a means to quantify the accuracy of estimations, computes the optimal values of θ by solving,

$$\min_{\theta} J(\theta) = -\frac{1}{T}\sum_{i=1}^{D}\sum_{t=0}^{T^i} \ln p\left((\xi^{t,i}; \dot{\xi}^{t,i})|\theta\right) \qquad (4)$$

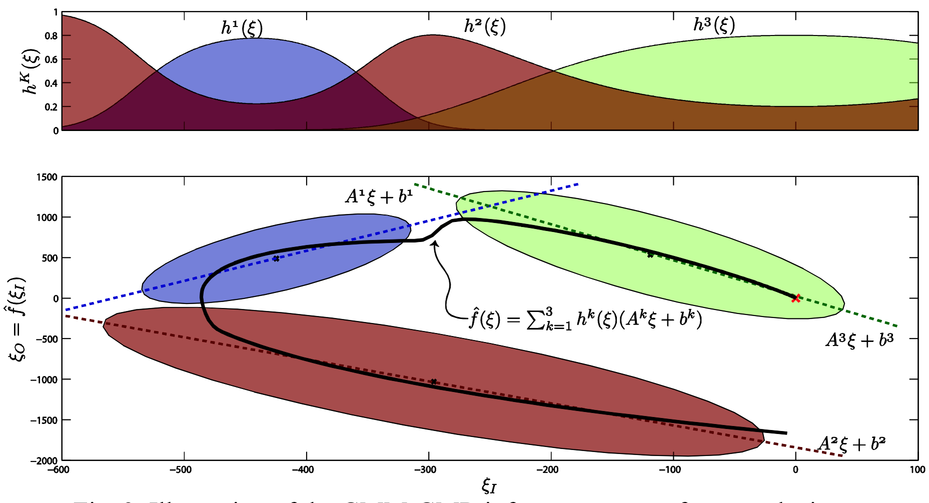


Fig. 3. Illustration of the GMM-GMR inference process for reproducing learned trajectories.

The approach provides a sound ground for the estimation of non-linear DS which is not heuristic driven and, thus, has the potential for much larger sets of applications. Also, by presenting the properties of being time-invariant and globally asymptotically stable at the target, the DS estimated with SEDS are able to respond immediately and appropriately to perturbations that could be encountered during reproduction of the motion.

5 Representation of Skills Knowledge

For a robotic system to perform different skills and task in a changing and unstructured scenario it is important to develop a structure in which to organize its acquired knowledge in a manner that allows it to retrieve it in order to use it to deal with the current context constraints.

The aim is to populate a Knowledge database of the available robot skills for reproduction, learned as multivariate dynamical systems from demonstrations. Dynamical system parameters, attractors, trajectories, bifurcations, can be regarded with a representational status, storing knowledge which can influenced behaviour, (vanGelder, et al., 1995). The knowledge database needs to hold all necessary information for reproduction of the skills.

Before one can start to deal with the issues of building a representation of the world, and the commitments it must ascribe to for the representations of knowledge and the process of reasoning to work, a key decision must be made on which aspects of the world one will focus on and which aspects of the world one will choose to ignore, and how the knowledge about the world would be structured. For any representational system the question of what is needed to be modelled and what can be ignored or abstracted away is a fundamental issue, (Anderson, 2003). The abstractions are necessary because no system can possible manage a world model that includes the whole of the world.

Traditional representations in artificial intelligence have focused on the symbolic discrete representation of objects and actions, (Geib, et al., 2006). The objects-actions dichotomy is an important abstraction for the performance of robots as embodied cognitive situated agents. A majority of approaches in cognitive architectures focus on skill knowledge about how to generate or execute sequences of actions, while often relegating equally important conceptual knowledge dealing with categories of objects, situations or other concepts, (Langley, et al., 2009).

A robot task behaviour will be considered to be of the form <*robot pick blue ball*>, <*robot place cup on plate*>, etc. in which an action is describe requesting an operation upon an object for a goal oriented skill task behaviour. Therefore the Skills Knowledge database must represent elements in two principal directions, of objects and task actions. Since for a single behaviour there could be more than one pairing <*object, task model*> the addition of at least one more dimension could be required in order to prevent ambiguities.At least one more direction for representations would seem necessary, like a description of the state of the environment. To resolve this problems it is suggested to considered two more representational directives, one for the task goal, and one for the configuration of the current state of the world, mainly objects position and relations with themselves, the robot and a human operator.

Much of an agent's knowledge must consist of skills, concepts and facts about the world. The importance of dealing with objects when developing robotic agents which must perform in the world seems quite evident, since most of a robot's operations in the environment would be bound to the manipulation of an object. It seems clear that the representational attributions must be oriented to dealing with objects in the environment and the actions that can be executed on them. In order to deal with changing dynamic environments representations must also have the ability to handle different situations or events. Recognizing different events or situations in the environments and the objects and actions that pertain to the current configuration of the environment is a crucial ability that the robotic systems discussed here must be able to possess. A robot must be able to extract from the world the relevant information of the objects, and the actions they afford, and relate them to an event of its task, and must be able to express this knowledge in representational terms, in order to perform and take action in the environment. Fig. 4 shows the representation of the skills in the knowledge database.

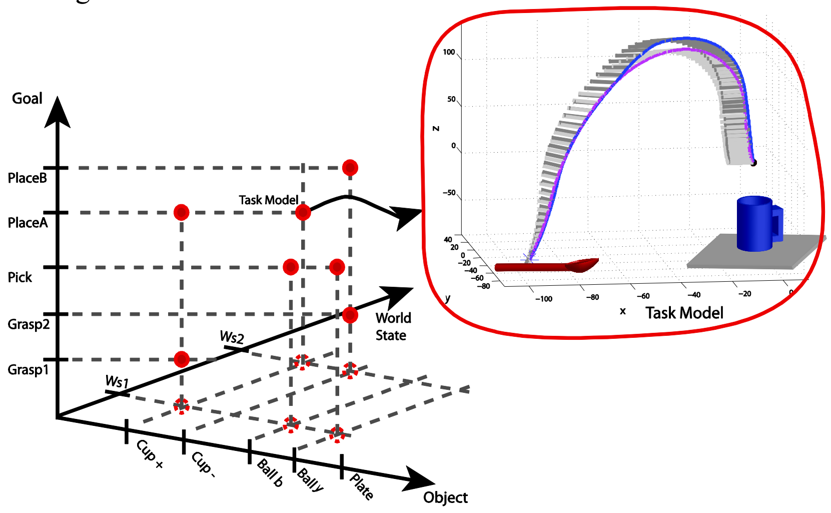


Fig. 4. Illustration of the GMM-GMR inference process for reproducing learned trajectories.

In this way, a task behaviour could be represented by the for ***"Do a Task (A), To an Object (X), For achieving Goal (Z), When current State of the World (W)"***. Therefore, the tuple formed by by *<Do = Task(A), To = Object(X), For = Goal(Z), When = World State(W)>* holds all necessary information for the reproduction of the task behaviour.

For example, in *<robot pick blue ball>* we'll have, *<Do = Pick_r, To = Ball_b, For = *, When = + >*, where the World State, +, could perhaps help infer that right hand is the closest to the object, and therefore the best model to choose would be *Pick_r*, or alternative, it could have the meaning that the right hand path has obstacles and is better to use *Pick_l*.

6 Generation of Robot Skills

When receiving order commands for reproduction of a skill, there could be several instances in which an adaptation of previously learned models or generation of new task models becomes necessary. The robotic system would be able to retrieve the closer models of a task from the knowledge base by finding the answer to the phrase ***"Do Action (A) …"*** for its current task constraints when being presented with the triple <*Object, Goal, World State*>. Using both, the already learned model of a skill, and the extracted constraints information of the current task, the model of the skill is adapted to reproduce the new task. The robot would be given from the different models of perception and interaction the required appropriate commands ordering the reproduction of a skill. Therefore the process for reproduction of a skill is reduced to a search in the Knowledge base for the appropriated Task Model satisfying the desired command and reproducing the obtained skill behaviour, given by <*Task, Object, Goal, World State*>.

The process for automatic generation of a new skill model can be summarized as follows. First, identifying the closest models similar to requested task <*Object, Goal, World State*>. Compare the models and classified them. If they are similar merge them and update the database representation of Model MRS* When no similarity can be found and the models are too different, combine the Models as a Mixture of Experts, identifying their regions of "expertise", where each model contribution would be more dominant. And reproduce the skill behaviour given by <*NewModel, Object, Goal, World State*>.

The idea behind is that different components of the learned models of skills can model the new skill distribution in different regions of input space, and will look to determine the functions that decide which components are dominant in which region. Fig. 5 illustrates the process.

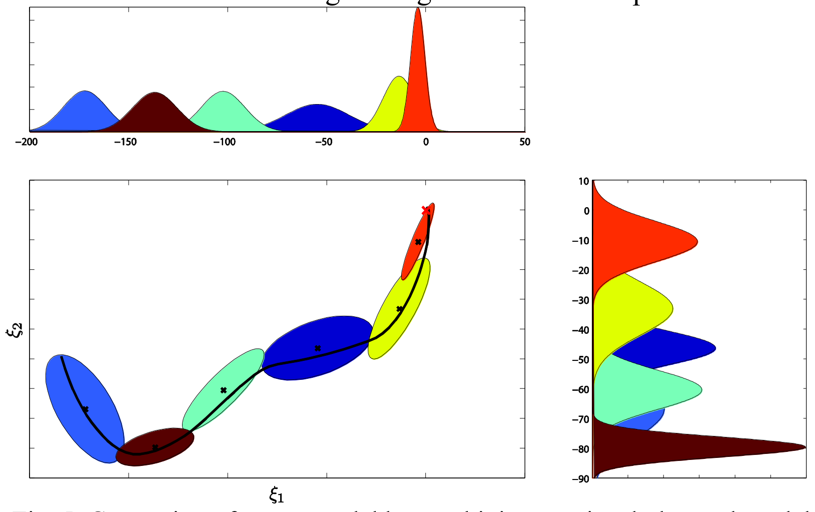


Fig. 5. Generation of a new model by combining previously learned models.

7 Discussion and Future Work

This work is centred on the aspiration of building humanoid robots capable of interacting with humans in their homes, workplaces, and communities, providing support in several areas, and collaborating with humans in the same unstructured working environments. The aspiration is to have humanoid robots acting as robot companions and co-workers sharing the same space, tools, and activities. This paper focus is on topics concerning the learning, representation, generation and adaptation, and reproduction of robot skills.

The main contribution of this work is the proposition of a framework for the generation and adaptation of learned models of a skill for complying with task constraints, Fig. 2. In which a database of the skills is built with the models of the skills learned through demonstrations. During execution the constraints of a requested task are extracted from the perception of the world state and the models of an appropriate skill are retrieved from the skills knowledge database. With all available information a new adapted task model is generated for reproduction.

A Learning from Demonstration algorithm have been studied and implemented in teaching and learning with the robot the different sets of skills employed in the rest of the framework.

A knowledge database of skills has been developed and implemented. The knowledge database allows for the storage, classification and retrieval of learned models of skills. A knowledge database is populated with the robot available skills, learned by demonstration, for latter reproduction.

Also, modalities that allows for the adaptation and generation of new skill models based on the already learned models of skills stored in the knowledge database has been developed and implemented. To adapt and generate a task model, the closest models of a skill are combine as a mixture of experts, identifying the regions of ``expertise" where each model contribution would be more dominant.

Acknowledgements

This work has been funded by the CICYT-MICINN Project Arcadia DPI2010-21047 and the Community of Madrid Project RoboCity2030 S2009/DPI-1559.

References

Albus, J.S. 1991. Outline for a theory of intelligence. Systems, Man and Cybernetics, IEEE Transactions on. Vol 21. Num 3. Pages 473-509.

Albus, J.S., Barbera, A.J. 2005. RCS: A cognitive architecture for intelligent multi-agent systems. Annual Reviews in Control. Vol 29. Num 1. Pages 87-99.

Ambrose, R.O., Aldridge, H., Askew, R.S., Burridge, R.R., Bluethmann, W., Diftler, M., Lovchik, C., Magruder, D., Rehnmark, F. 2000. Robonaut: NASA's space humanoid. Intelligent Systems and their Applications, IEEE, Vol 15. Num 4. Pages 57-63.

Anderson, M. L. 2003. Embodied cognition: a field guide. Journal Artificial Intelligence. Elsevier Science Publishers Ltd. Pages 91-130.

Beer, R. D. 2000. Dynamical approaches to cognitive science. Trends in Cognitive Sciences. Vol 4. Num 3. Pages 91-99.

Billard, A., Calinon, S., Dillmann, R., Schaal, S. Robot Programming by Demonstration. In Handbook of Robotics. Editor Siciliano, B., Khatib, O. Springer. Pages 1371-1394.

Brooks, R.A. 1990. Elephants Don't Play Chess. Robotics and Autonomous Systems. Vol 6. Pages 3-15.

Brooks, R.A. 1996. Behavior-based humanoid robotics. Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. Vol 1. Pages 1-8.

Clark, A. 2004 Mind and Causality. In Embodiment and the Philosophy of Mind. Editor Peruzzi, A. John Benjamins Pub.

Clark, A., Grush, R. 1999. Towards a cognitive robotics. Journal of Adaptive Behavior. Vol 7. Num 1. Pages 5-16.

De Silva, L., Ekanayake, H. 2008. Behavior-based Robotics And The Reactive Paradigm A Survey. Computer and Information Technology, 2008. ICCIT 2008. 11th International Conference on. Pages 36-43.

Gat, E. 1997. On three-layer architectures. In Artificial intelligence and mobile robots. Editors Kortenkamp, D, Bonnasso, R. P., Murphy, R. AAAI Press. Pages 195-210.

Geib, C., Mourao, K., Petrick, R., Pugeault, N., Steedman, M., Krueger, N., Wörgötter, F. 2006. Object Action Complexes as an Interface for Planning and Robot Control. International Conference on Humanoid Robots.

Gribovskaya, E, Zadeh, Khansari, M. S., Billard, A. 2010. Learning non-linear multivariate dynamics of motion in robotic manipulators. International Journal of Robotics Research.

Ijspeert, A., Nakanishi, J., Schaal, S. 2001. Trajectory formation for imitation with nonlinear dynamical systems. In IEEE international conference on intelligent robots and systems. Pages 752-757.

Ijspeert, A., Nakanishi, J., Schaal, S. 2002. Movement Imitation with Non-linear Dynamical Systems in Humanoid Robots. In IEEE International Conference on Robotics and Automation. Pages 1398-1403.

Kelso, J.A.S. 1995. Dynamic Patterns: The Self-Organization of Brain and Behavior. Mit Press.

Khansari-Zadeh, S.M., Billard, A., 2011. Learning Stable Nonlinear Dynamical Systems With Gaussian Mixture Models. Robotics, IEEE Transactions on. Vol 27. Num 5. Pages 943-957.

Khansari-Zadeh, S.M. and Billard, A. 2010. Imitation learning of globally stable non-linear point-to-point robot motions using nonlinear programming. Intelligent Robots and Systems, International Conference on. Pages 2676-2683.

Langley, P., Laird, J. E., Rogers, S. 2009. Cognitive architectures: Research issues and challenges. Cognitive Systems Research. Vol 10. Num 02. Pages 141-160.

Legg, S., Hutter, M. 2006. A formal measure of machine intelligence. In Proc. 15th Annual Machine Learning Conference of Belgium and The Netherlands. Pages 73-80.

Levesque, H. Lakemeyer, G. 2008. Cognitive Robotics. Chapter 23. In Handbook of Knowledge Representation. Editor van Harmelen, F., Lifschitz, V. Porter, B. Elsevier.

Mataric, M.J. 1997. Behavior-Based Control: Examples from Navigation, Learning, and Group Behavior. Journal of Experimental and Theoretical Artificial Intelligence. Vol 9. Pages 323-336.

Murphy, R.R. 2000. An Introduction to AI Robotics. MIT Press.

Nicolescu, M., Mataric, M.J. 2002. A Hierarchical Architecture for Behavior-Based Robots. In Proc., First International Joint Conference on Autonomous Agents and Multi-Agent Systems.

Poole, D., Mackworth, A., Goebel, R. 1998. Computational Intelligence: A Logical Approach. Oxford University Press, USA.

Simmons, R.G. 1994. Structured control for autonomous robots. IEEE Transactions on Robotics and Automation. Vol 10. Num 1. Pages 34-43.

Schöner, G. 2008. Dynamical Systems Approaches to Cognition. In Cambridge Handbook of Computational Cognitive Modeling. Editor Sun, R. Cambridge University Press.

Stoytchev, A., Arkin, R.C. 2001. Combining deliberation, reactivity, and motivation in the context of a behavior-based robot architecture. In Computational Intelligence in Robotics and Automation, 2001. Proceedings 2001 IEEE International Symposium on. Pages 290-295.

van Gelder, T., Port, R. F. 1995. Mind as motion. Massachusetts Institute of Technology. Cambridge, MA, USA.

Wooldridge, M., Jennings, N.R. 1995. Intelligent Agents: Theory and Practice. Knowledge Engineering Review. Vol 10. Pages 115-152. Pages 227-233.

CAPÍTULO 5

OBJECT TAGGING FOR HUMAN-ROBOT INTERACTION BY RECOLORIZATION USING GAUSSIAN MIXTURE MODELS

M. GONZÁLEZ-FIERRO[1,2], M. A. MALDONADO[2], J. G. VICTORES[1], S. MORANTE[1,2] and C. BALAGUER[1]

[1]Robotics Lab, Universidad Carlos III de Madrid; [2]Samsamia Technologies S.L. [1]{mgpalaci,jcgvicto,smorante,balaguer}@ing.uc3m.es; [2]{miguelgfierro, miguelmaldonado, santimorante}@samsamia.com

In this work, we present a method to tag objects by applying a color model learned from another source object. We learn the statistical color model of objects using Gaussian Mixture Models and Expectation-Maximization algorithm. The source model is transferred to the target object to be tagged by matching the Gaussian distribution that best describe the color structure. This makes the target gain the color model of the source while maintaining its initial appearance. This algorithm can be used in Human-Robot Interaction to visually tag objects for selection, targeting or discrimination. We perform some experiments to test our proposed method.

1 Introduction

To allow robots to share their living space with humans, they must be able to understand the environment and act intelligently. One of the first steps to accomplish this behavior is enabling robots to identify objects or people. However, in many cases, this is a complex task for the robot alone. Humans can help the robot to understand the environment by helping it to select and tag targets with which to perform a desired task. This can be done by teleoperation (Pierro, 2009), interaction through gestures (Bueno, 2012) or Learning from Demonstration (Argall, 2008).

Object tagging is an area of research that has many applications in social networks and in the Internet in general, and it is widely addressed in computer vision. In (Bergman, 2011), an automatic method for object tagging is presented, where objects like skin, the sky or foliage are automatically tagged. Another popular method is the "bag of words" (Csurka, 2004), where a bag of features treated like words is computed, and then classified to visually categorize objects.

Despite these efforts, there is still a huge gap between what a human is capable of tagging and automatic selection, as (Pavlidis, 2009) defends. There is a growing interest on relying on humans to solve this difficult computer vision tasks (Sigala, 2004). A widely known method is the reCAPTCHA of Google (Von Ahn, 2008), which aims to digitalize old texts with the help of millions of users throughout the web. The first of the two words that appears in a reCAPTCHA is used for security reasons, to find out if the user is a human or a machine. The second one is a word that Google is not capable of recognizing using OCR methods, and is thus presented to the user for human identification.

In this paper we propose a supervised method to tag objects using color substitution. In a first step a color model of the source object is learned using Gaussian Mixture Models (GMM). This color is then applied to the target object, but maintaining the shape and visual structure of the target. As a result, the object changes its color but it is still identifiable by the human. Some examples of color substitution are (Pitie, 2005), that performs histogram matching, or (Tai, 2005), that makes parametric matching. Our work is based on (Huang, 2009), that proposed a method of image recoloring to help colorblind people to recognize objects. The authors swap colors that people with color vision deficiencies may have difficulties in perceiving with colors that they may identify more easily. The final color model application is performed through the method described in (Saphira, 2009).

The aim of this work is to make it easier for a human to interact with a robot in a cluttered environment. Usually, object tagging is performed using a bounding box that surrounds the target object, and at times an attached text. Our proposed method allows the recoloring of objects from a determined source, avoiding the use of occluding bounding boxes. Additionally, different classes of objects may be tagged with user friendly and easily recognizable patterns. Fig. 1 presents a side-by-side view of the bounding box method and the proposed method.

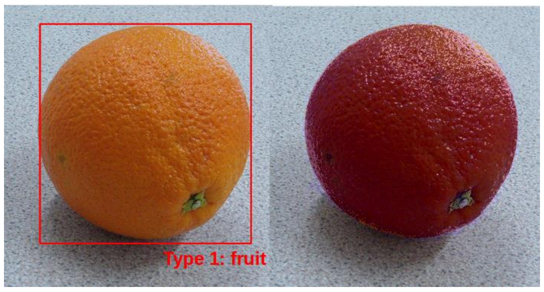


Fig. 1. On the left side: the usual tagging system, the object is tagged by a bounding box and a text where the type is expressed. On the right side: our proposal, the object is tagged using a determined source color model, in this case red. As it can be seen, occlusion of the environment by the bounding box (left) is avoided through the use of the presented recoloring mechanism.

The document is ordered as follows: Section 2 outlines the basic mathematical used tools, Section 3 explains how the objects are tagged by color substitution, Section 4 presents the experiments, and Section 5 provides several conclusions.

2 Basic tools

This section reviews some of the most relevant algorithms that the authors use to perform the presented segmentation and tagging recoloring process.

2.1 Gaussian Mixture Model

A Gaussian Mixture Model (GMM) is a parametric probabilistic model for representing subpopulations in training datasets of points. It can be expressed as a weighted sum of **M** multivariate Gaussian distributions. As explained in (Reynolds, 2008), this model can be expressed as:

$$p(x|\lambda) = \sum_{i=1}^{M} w_i g(x|\mu_i, \Sigma_i) \tag{1}$$

Where x is a D-dimensional data vector, w_i are the mixture weights, and g is the multivariate Gaussian probabilistic density function. This density function is defined by:

$$g(x|\mu_i, \Sigma_i) = \frac{1}{(2\pi)^{D/2} |\Sigma_i|^{1/2}} exp \left\{ -\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1} (x-\mu_i) \right\} \tag{2}$$

where μ_i is the mean vector and Σ_i is the covariance matrix. This is a general model that can express several specific scenarios, i.e. a single Gaussian model (where **M**=1), a univariate Gaussian model (where the mean and covariance are actually scalars), or a case where the information among the axes is non-correlated (resulting in a diagonal covariance matrix instead of a full one).

A number of algorithms exist to determine the numerical values of the parameters of a GMM such that it correctly predicts the values of a training dataset *x*. The quality of this prediction is equal to the likelihood of the dataset *x* given the parameters λ. This is,

$$L(\lambda : x) = p(x|\lambda) = \prod_{t=1}^{T} p(x_t|\lambda) \tag{3}$$

where the parameters λ for a GMM are M, *w*, μ and Σ. Thus, the problem of determining the best predictor is equivalent to finding the parameters that optimize the likelihood. This gives name to the family of Maximum Likelihood Estimation (MLE) algorithms. While MLE for simple distributions is trivial, the GMM case presents a non-linear function of the parameters. The MLE of a GMM may solved through iterative methods, such as the Expectation-Maximization (EM) algorithm.

The EM algorithm is initialized with a set of *a priori* parameters λ. At each iteration, the algorithm looks for a set parameters λ' such that $p(x|\lambda') \geq p(x|\lambda)$. This process is repeated until convergence within a specified threshold reference. The following set of equations guarantee a monotonic increase of the likelihood thus enabling the advance towards an optimal model. They update the expectation of the Gaussian moments and thus compose the Expectation (E step) of the EM algorithm:

Mixture weights:

$$w_i' = \frac{1}{T} \sum_{t=1}^{T} p(i|x_t, \lambda) \tag{4}$$

Means:

$$\mu_i' = \frac{\sum_{t=1}^{T} p(i|x_t, \lambda) x_t}{\sum_{t=1}^{T} p(i|x_t, \lambda)} \tag{5}$$

Variances:

$$\sigma_i'^2 = \frac{\sum_{t=1}^{T} p(i|x_t,\lambda)x_t^2}{\sum_{t=1}^{T} p(i|x_t,\lambda)} - \mu_i'^2 \tag{6}$$

where $w_i^{'}$, $\mu_i^{'}$, and $\sigma_i^{'}$ refer to arbitrary elements of their respective vectors. The Maximization (M step) of the EM algorithm is performed by computing the *a posteriori* distribution. For component *i*, this is given by:

$$p(i|x_t,\lambda) = \frac{w_i g(x_t|\mu_i,\Sigma_i)}{\sum_{k=1}^{M} w_i g(\mu_k,\Sigma_k)} \tag{7}$$

2.2 GrabCut Algorithm

We make use of the GrabCut algorithm (Rother, 2004) to segment the image. GrabCut uses Gaussian Mixture Models and Expectation Maximization to find globally optimal segmented solutions.

The Grabcut algorithm includes two parts, hard segmentation and border matting. In the hard segmentation phase, the algorithm estimates the foreground and the background of the image by using an iterative version of graph-cut optimization (Boykov, 2001). Then, in the border matting phase, alpha values are obtained in a narrow region in the surroundings of the segmentation boundary.

2.3 Kullback–Leibler divergence

The Kullback–Leibler divergence (KL) is a method for determining the similarity between two probabilistic distributions (Kullback, 1951). Usually, one of the distributions is the real data (P) and the other (Q) is the distribution model that you want to use as an interpretation of your data. KL is a measure of how much information your model gives you about the data you are modeling. Formally, for discrete systems, KL is defined as:

$$D_{KL}(P||Q) = \sum_i ln(\frac{P(i)}{Q(i)})P(i) \tag{8}$$

which can be described as the sum of the likelihood of observing one data with the distribution P if the particular model Q actually generates the data. The lower the KL distance is, the more similar the distributions are. In other words, lower KL results indicate that the statistical model Q, assumed for interpret the real data P, is good explaining it. Notice that this measure is distinct when talking about $D_{\mathrm{KL}}(P||Q)$ and $D_{\mathrm{KL}}(Q||P)$. This is the reason why it is considered a "non-symmetric" distance.

3 Tagging objects by color substitution

The full process of target object recoloring is depicted in Fig. 2.

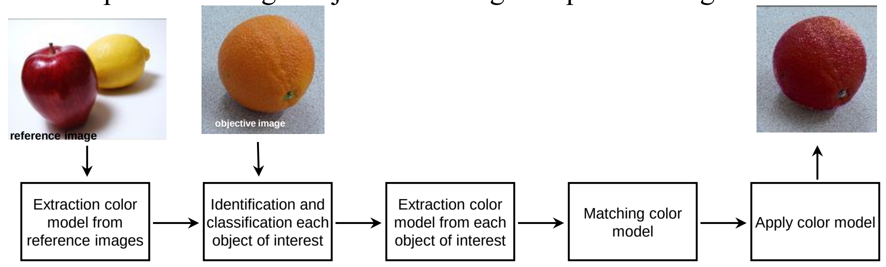

reference image
objective image
Extraction color model from reference images
Identification and classification each object of interest
Extraction color model from each object of interest
Matching color model
Apply color model

Fig. 2. Full process.

It basically consists in the following 5 distinct steps.

1) We extract the color model of the reference images; each of them represents a category. This process is actually three-fold:

- Selection of the images from which to extract the color model.
- Segmentation of each object using the GrabCut algorithm, as explained in Section 2.1.
- Estimation of its GMM color model using the EM algorithm, both explained in Section 2.2.

2) In the target image, we perform a supervised identification of objects of interest. For this purpose we select the area of each object of interest and classify them within the categories available.

3) We segment each of the identified objects and extract their color model.

4) Using the KL distance described in Section 2.3, we match the Gaussians of the target object with that of the source object.

5) We apply the color model to tag the object (recoloring) through the method described in (Saphira, 2009).
As a result, we obtain a target image tagged with the source color. The next step would be to perform a task like monitoring, searching, tracking or targeting.

4 Experiments

Some results of our proposal are shown in Fig. 3, Fig. 4 and Fig. 5.

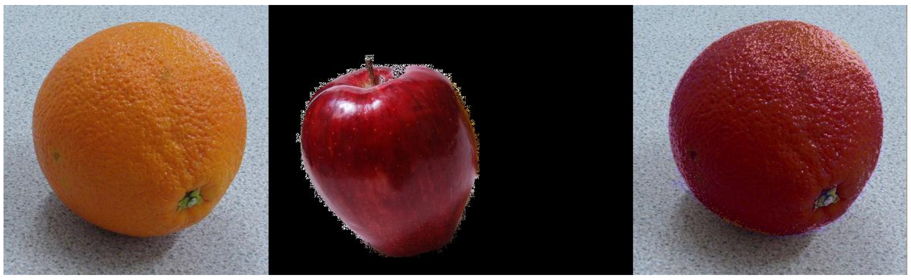


Fig. 3. On the left side: an orange. In the center: an apple. On the right side: an orange recolored like an apple.

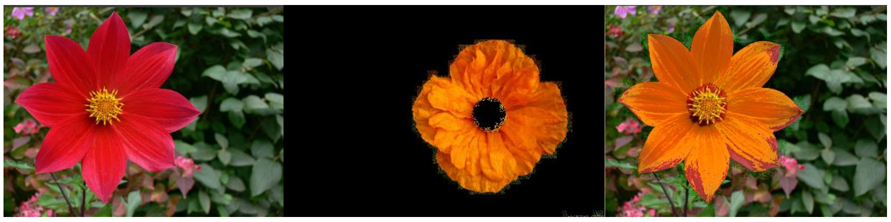


Fig. 4. Process of image color substitution: The target object is selected (left image), extraction of color model of source object (center image), application of color model to the target object (right object).

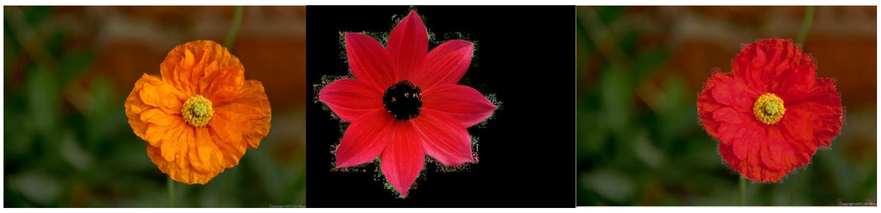


Fig. 5. Process of image color substitution, swapping the source and the target.

Initially we select GMM with 3 components to define RGB values, one component to describe each channel. However, instead of choosing three components, we could have actually chosen any number of Gaussian components. The results with several different choices are shown in Fig. 6.

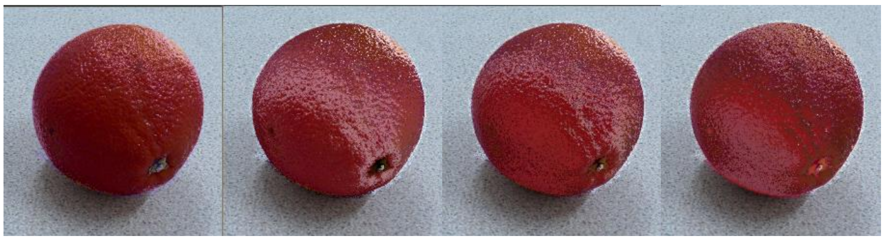


Fig. 6. From left to right, object recoloring using 3, 6, 9 and 12 Gaussians.

One of the drawbacks of our application is that only a uniform color model can be learned at the same time. To use 2 or more colors for tagging, the process should be repeated for every color.

5 Conclusions

This paper proposes a tagging system for objects to be used in human-robot interaction for selection, discrimination or targeting. Target objects are tagged by color recoloring instead of the classical bounding box mechanism, thus avoiding environmental occlusion (especially relevant in cluttered environments) and the possibility of applying user-friendly color patterns for labeling. This process can be repeated for any object that we need to tag. In a cluttered environment, where it is difficult to distinguish between objects, our tagging system could be useful since it is friendlier from a visual point of view.

As a future work we propose to create an interface where our system is integrated with the vision of the robot, then a human operator can interact with the environment easily and benefit from the advantages of our tagging system.

Acknowledgements

The research leading to these results has received funding from the ARCADIA project DPI2010-21047-C02-01, funded by CICYT project grant on behalf of Spanish Ministry of Economy and Competitiveness and from the RoboCity2030-II project (S2009/DPI-1559), funded by Programas de Actividades I+D en la Comunidad de Madrid and cofunded by Structural Funds of the EU.

References

Argall, B. D., Chernova, S., Veloso, M., & Browning, B. 2009. A survey of robot learning from demonstration. Robotics and Autonomous Systems, 57(5), 469-483.

Bergman, R., & Nachlieli, H. 2011. Perceptual segmentation: combining image segmentation with object tagging. Image Processing, IEEE Transactions on,20(6), 1668-1681.

Boykov, Yuri Y., and M-P. Jolly. 2001. "Interactive graph cuts for optimal boundary & region segmentation of objects in ND images." Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on. Vol. 1. IEEE.

J. G. Bueno; M.G.Fierro; L.Moreno; C.Balaguer. 2012. Facial Gesture Recognition using Active Appearance Models based on Neural Evolution . 2012 Conference on Human-Robot Interaction (HRI 2012). Boston. USA.

Csurka, G., Dance, C., Fan, L., Willamowski, J., & Bray, C. 2004. Visual categorization with bags of keypoints. In Workshop on statistical learning in computer vision, ECCV (Vol. 1, p. 22).

Huang, J. B., Chen, C. S., Jen, T. C., & Wang, S. J. 2009. Image recolorization for the colorblind. In Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on (pp. 1161-1164). IEEE.

Kullback, S., & Leibler, R. A. 1951. On information and sufficiency. The Annals of Mathematical Statistics, 22(1), 79-86.

Pavlidis, T. 2009. Why meaningful automatic tagging of images is very hard. In Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on (pp. 1432-1435). IEEE.

P.Pierro; D.Hernandez; M.G.Fierro; C.Balaguer; L. Blasi; A. Milani. 2009.A human-humanoid interface for collaborative tasks. Second workshop for young researchers on Human-friendly robotics. Sestri Levante. Italy. Dec.

Pitie, Francois, Anil C. Kokaram, and Rozenn Dahyot. 2005. N-dimensional probability density function transfer and its application to color transfer. Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on. Vol. 2.

Reynolds, D. 2008. Gaussian mixture models. Encyclopedia of Biometric Recognition, 2(17.36), 14-68.

Rother, Carsten, Vladimir Kolmogorov, and Andrew Blake. 2004. Grabcut: Interactive foreground extraction using iterated graph cuts. ACM Transactions on Graphics (TOG). Vol. 23. No. 3. ACM.

Shapira, L., Shamir, A., & Cohen Or, D. 2009. Image Appearance Exploration by Model- Based Navigation. In Computer Graphics Forum (Vol. 28, No. 2, pp. 629-638). Blackwell Publishing Ltd.

Sigala, M. 2008. Web 2.0, social marketing strategies and distribution channels for city destinations: enhancing the participatory role of travelers and exploiting their collective intelligence. Information communication technologies and city marketing: Digital opportunities for cities around the world, 220-244.

Tai, Yu-Wing, Jiaya Jia, and Chi-Keung Tang. 2005. Local color transfer via probabilistic segmentation by expectation-maximization. Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Vol. 1.

Von Ahn, L., Maurer, B., McMillen, C., Abraham, D., & Blum, M. 2008. reCAPTCHA: Human-based character recognition via web security measures.Science, 321(5895), 1465-1468.

CAPÍTULO 6

DISTINGUISHING BETWEEN SIMILAR OBJECTS BASED ON GEOMETRICAL FEATURES IN 3D PERCEPTION

J. GARCÍA BUENO[1,2], A. MARTÍN CLEMENTE[1,2], M. GONZÁLEZ-FIERRO[1,3], L.MORENO[1] and C. BALAGUER[1]

[1]Robotics Lab, Universidad Carlos III de Madrid; [2]beMee Technology S.L.; [3]Samsamia Technologies S.L.,
{jgbueno, aimartin, mgpalaci, moreno, balaguer}@ing.uc3m.es

In this study, we propose a meta-classifier to identify and distinguish between objects with confusable geometry, similar in appearance, colors and dimensions. The suggested classifier is able to differentiate successfully between 3D point clouds with a high degree of similarity extracting parameters such as 3D features from the surface of the cluster, absolute geometrical boundaries and also color distributions. Objects are until now detected based on its Viewpoint Feature Histograms (VFH) with the downside of being scale independent, making difficult the segmentation of items with simile geometry. Furthermore, this paper discusses several distance metrics to choose which one fits most with VFH descriptors. We present several experiments that validate the model and corroborate the meta-classifier here proposed.

1 Introduction

During the last years, object recognition has turned into a prior research line in robotics. The idea of designing robots capable of detecting and recognizing objects in their vicinity make more natural and prosperous their integration in humanlike scenarios such as offices, kitchens (Rusu, 2010) or living rooms (Bueno, 2012; Liu, 2001). With the aim of detecting and interacting with these objects, robot perception abilities have to be mostly enhanced. Time-Of-Flight technology and structured-light have become decisive in perception sensorial devices such as Kinect camera or Asus Xtion Pro Live. Those instruments are capable of capturing RGB-D point clouds at high frame rates allowing robots to interpret and analyze with clarity what is coming about (Rusinkiewicz, 2002) and consequently react to these perturbations.

Objects are intended to be detected mostly for grasping. This fact leads to a consequent challenge: items have to be not only successfully detected but also favorably located in order to determine the path to reach and grasp them (Bueno, 2011). This requirement demands two different goals: object pose by means of its 6 DOF and object labeling based on its inherent features such as color, texture, shape, 3D key-points, geometry, center of gravity, etc. For the first purpose, there exist multiple strategies such as 3D recognition based on correspondence grouping using SHOT32 features, ICP-like algorithms (Rusinkiewicz, 2001) or 3D matching (Bueno, 2012; Scovanner, 2007; Knopp, 2010). For the last, 3D features are necessary. Some of the perception features used nowadays are SIFT 3D (Scovanner, 2007), NARF (Bueno, 2012), SURF 3D (Knopp, 2010; Bay, 2006) or FPFH (Rusu, 2009) among others. Without exception, they extract consistent features from a 3D point cloud cluster such as edges, normal variations on the surfaces or intrinsic geometric models such as lines, planes or spheres.

This study is focused on FPFH (Rusu, 2010), a histogram of values that represents with a single histogram the variations and most remarkable variations for a specific surface. FPFH histogram might be used to search similarities among different objects with scale invariability. This feature becomes an issue when databases are composed by scaled versions of similar objects at different scales (this circumstance is quite common for kitchen items such as spoons, mugs, glasses or dishes). Classifiers are always supported by a distance metric in charge of comparing candidates with the observed guest. The selection of a correct metric can affect directly to the recognition rates as discussed in (Martin, 2009).

Subsequent sections of this paper are organized as follows. Section 2 is a formal description of FPFH histogram. Then section 3 presents the proposed meta-classifier algorithm to detect similar histogram. Afterwards in section 4 the experimental results are discussed and later section 5 addresses the highlighted conclusions.

2 Point Feature Histogram

There exist multiple alternatives to represent and interpret surfaces. As a general rule those techniques analyze point cloud curvatures and normals along the surfaces to reduce a high dimensional problem into something more manageable. Most scenes will contain 3D clusters that may represent dissimilar objects or not, making challenging to create a wide representation for 3D point clouds into a single feature list. (Rusu, 2009; Rusu, 2010) proposes a novel point features representation named Point Feature Histogram (PFH) based on the curvature and normal orientation between neighbors in a certain point cloud. Since surface normals and curvature estimations are able to capture precise details of the geometry around a point, most algorithms base their intention focusing on them to deal with false correspondences.

In order to enhance recognition approaches, there exist many different feature representations for a surface. In general it would be ideal to represent each point with an information label that contains the geometry class it belongs to: edge point, spherical surface point, etc.

Following the above, there is a need of finding a multi-dimensional feature space which separates surfaces in different categories. In terms of point-wise analysis, the concept of a dual-ring neighborhood is introduced for any point $p_i \in \wp$ as

$$(\exists) r_1, r_2 \in \Re, r_1 < r_2 \text{ such that} \begin{cases} r_1 \Rightarrow \wp^{k_1} \\ r_2 \Rightarrow \wp^{k_2} \end{cases}, \text{with } 0 < k_1 < k_2$$

where $\wp$ represents the complete set of 3D points. Both radii r_1, r_2 have a specific target. While r_1 represents the surface normal at the query point p_i, obtained from the Principal Component Analysis of the neighborhood patch $\wp^{k_1}$. The radius r_2 delimits the PFH representation itself.

As it has been stated before, the main goal of the PFH formulation is to encode the $\wp^{k_2}$ neighborhood's geometrical properties by generalizing the mean curvature around p_i using a multi-dimensional histogram of values. Assuming that normal values of neighbors of p_i have been already computed, it is possible to state that having two different points p_i and p_j the relative difference between them can be defined as follows:

$$\overrightarrow{p_{ij}} = p_i - p_j, \overrightarrow{p_{ji}} = p_j - p_i$$

$$if\ \cos^{-1}(\overrightarrow{n_i} \cdot \overrightarrow{p_{ji}}) \leq \cos^{-1}(\overrightarrow{n_j} \cdot \overrightarrow{p_{ij}}) \rightarrow \begin{cases} p_s = p_i, n_s = n_i \\ p_t = p_j, n_t = n_j \end{cases}$$

$$else \rightarrow \begin{cases} p_s = p_j, n_s = n_j \\ p_t = p_i, n_t = n_i \end{cases}$$

being p_s the source and p_t the target. As the above condition takes place, p_s is chosen such that the angle between its associated normal and the line connecting the two points is minimal. It is then defined a Darboux coordinates frame at one of the points as shown in Fig. 1.

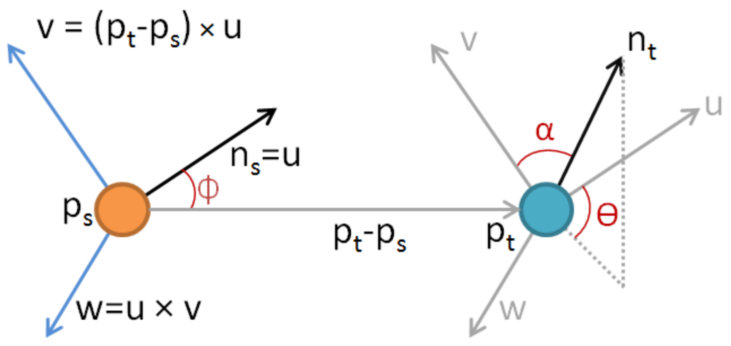


Fig. 1. Darboux coordinates frame at one of the point when computing PFH

where $u = n_s, v = u \times \frac{(p_t - p_s)}{\|p_t - p_s\|_2}$ and $w = u \times v$. With this in mind, the difference between the two normals n_s and n_t is defined with

$$\alpha = v \cdot n_t \tag{1}$$

$$\phi = u \cdot \frac{(p_t - p_s)}{\|p_t - p_s\|_2} \tag{2}$$

$$\theta = tg^{-1}\left(\frac{w \cdot n_t}{u \cdot n_t}\right) \tag{3}$$

The quadruplet $\langle \alpha, \phi, \theta, d \rangle$ reduces the 12 values x, y, z, n_x, n_y, n_z (for both points) to 4 with $d = \|p_t - p_s\|_2$. The influence region diagram is represented in Fig. 2 where p_q represents a query point and its k neighbors in a 3D sphere of radius r_1 are referred as p_{ki} .

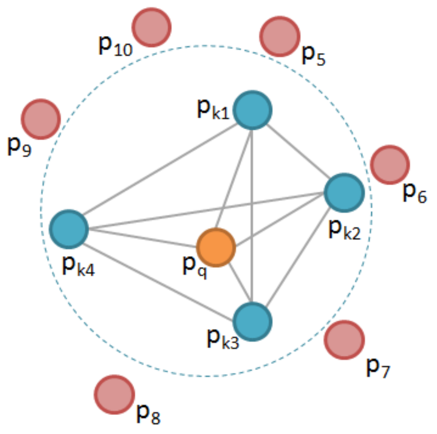


Fig. 2. Influence region diagrams for 3D neighbors in PFH

In order to encode the quadruplet for the whole point cloud, a histogram is created. The quadruplet values are divided on bins forming a histogram including on each bin the number of coincidences for this interval. Seeing that three of the four values are angles it is straightforward to divide the resulting values into equi-spaced bins.

2.1 Fast Point Feature Histogram

A first improvement for PFH consist on taking into account the histogram of the enclosing neighbors for each point and take them into account creating an averaged histogram averaged,

- For each point p_q a set of tuples $\langle \alpha, \phi, \theta \rangle$ are computed between a query point and its neighbors using the equations described above. This is called SPFH (Simplified PFH).
- then, for each point p_q its histogram is recomputed taking into account a weighted value of neighbors SPFH

$$FPFH(p_q) = SPFH(p_q) + \frac{1}{k}\sum_{i=0}^{k}\frac{1}{w_i}\cdot SPFH(p_k) \qquad (4)$$

where w_i represents a distance between the query point p_q and a neighbor point p_k in some given metric space (for this study euclidean distance metric will be computed). FPFH includes additional point pairs outside the r_1 radius sphere with a re-weighting strategy that smooth local discontinuities and define a better geometry around the query point.

2.2 View Point Feature Histogram

The second enhancement applied to FPFH concerns to the view point of the object. FPFH is view point dependent because it does not take into account the position of the sensor and therefore it is not suitable for dynamic scenarios such as mobile robot manipulators. To convert the feature estimator view point independent this viewpoint information has to be extracted from the quadruplet somehow. (Rusu, 2009) proposes to maintain viewpoint invariance while retaining invariance to scale computing additional statistics between the viewpoint direction and the normals estimated at each point.

The best way to remove this dependence is straight mixing the viewpoint direction directly into the relative normal angle calculation in the FPFH. A histogram of the angles that the viewpoint direction makes with each normal is computed. This is done from the viewpoint direction at the centroid of the cluster point cloud achieving the feature scale invariant. The histogram VFH then is created as in Fig.3 where there is a viewpoint component and then the FPFH component previously computed.

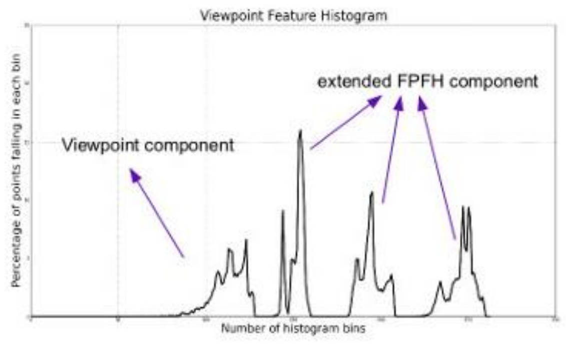


Fig. 3. Example of a VFH histogram with the extended PFH component

3 Proposed algorithm

As it has been stated in the previous section, VFH is scale invariant. In order to classify correctly objects with similar surfaces at different scales, it has been proposed a meta-classifier that not only takes into account the VFH histogram but also the absolute geometry of the object (height, width and depth) and color texture (Red, Green and Blue). The following Fig. 4 contains some of the items included in the database created specifically for this study.

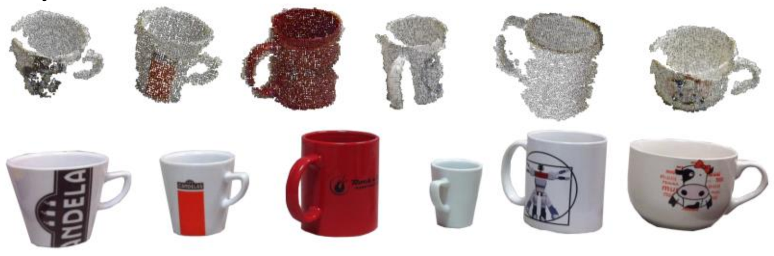


Fig. 4. 3D and color representations for the different objects that form the database. Notice that most of the items are geometrically similar.

As it can be noticed, there exist objects with similar surface shape but different size or color. This becomes uncertain and miscalculated if only FPFH features are taken into account during the categorization. For this reason, a meta-classifier has been proposed as it is represented in the following Fig.5.

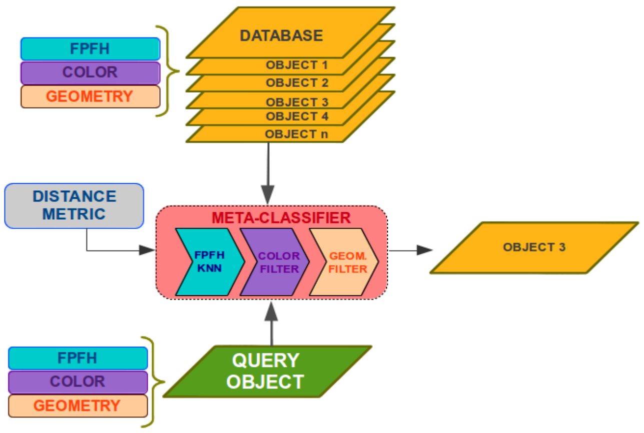


Fig. 5: Diagram of the classifier proposed for this study. Distance metric can be changed before filtering is done. The meta-classifier takes into account 3D surface descriptors, color and geometry for each candidate.

Geometry features contains absolute measures in the three axes giving more significance to the height value. That condition is imposed to maintain the view-point invariance since width and depth depend on the rotation of the object while height maintains constant relative to the supporting plane, maintain detached from yaw angle.

The next subsections explain each filter part of the complete meta-classifier. Firstly, the VFH filter will extract those objects from the database with analogous surfaces. Secondly, geometric filter will pick out those candidates which contain similar global geometry values. Lastly color filter will be in charge of selecting the most color-like remaining candidate from the database.

3.1 Distance metric

In order to measure distances between distributions, several distance metrics have been proposed. One of the aims of this research is to determine the degree of reliability for several *f-functions* to distinguish between different VFH histograms and color histograms. The list of functions and their equations can be found in the following Table 1. It will be assumed that all the distributions are normalized in order to satisfy the metrics conditions.

Table. 1. Distance metrics used for this study. Name of the function and the discrete-form equation

Distance metric	**Distance function**
Euclidean	$d = \sqrt{\sum_{i=1}^{N} (p_i - q_i)^2}$
Manhattan	$d = \sum_{i=1}^{N} \lvert p_i - q_i \rvert$
Bhattacharyya	$d = -\ln \sum_{i=1}^{N} \sqrt{p_i - q_i}$
Kullback-Leibler	$d = \sum_{i=1}^{N} p_i \ln \frac{p_i}{q_i}$
Chi-Square	$d = \sum_{i=1}^{N} \frac{(p_i - q_i)^2}{p_i + q_i}$
Histogram Intersection Kernel	$d = \sum_{i=1}^{N} \min(p_i, q_i)$

3.2 VFH filter

The earlier filter in the classifier extracts the VFH descriptor for the observed cluster and compares it with the database. Because this operation might spend a lot of time, a KNN tree is previously built with the catalog of objects already added to the database. This advancement achieves faster execution times.

The aim of this process is to determine which the ten most similar candidates are taking into account exclusively VFH descriptors. Those aspirants could have various scales or textures and the same surfaces. Fig. 6 represents a list of the 5 nearest neighbors for a single candidate that it is also shown (first of the second row).

Candidates are sorted by the metric function distance desired and selected. Notice that distance is zero for the first guess because it is already in the training database. This will never happen due to systematic errors in observation and sensors.

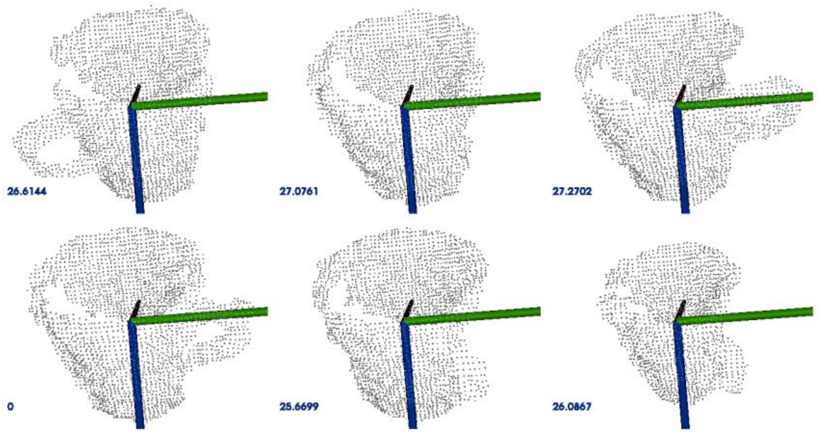


Fig. 6. Result of searching for similar objects in the database. The fourth item is the query and the VFH distance for this match is 0 because it is included in the database.

The KNN tree modeled contains, for each leaf, the VFH histogram of a single view for a specific object. VFH implementation uses 45 binning subdivisions for each of the three extended FPFH values, plus another 45 binning subdivisions for the distances between each point and the centroid and 128 binning subdivisions for the viewpoint component.

Upcoming Fig. 7 represents a set of views of the same object and their individual VFH values making it easy to understand the structure of the database.

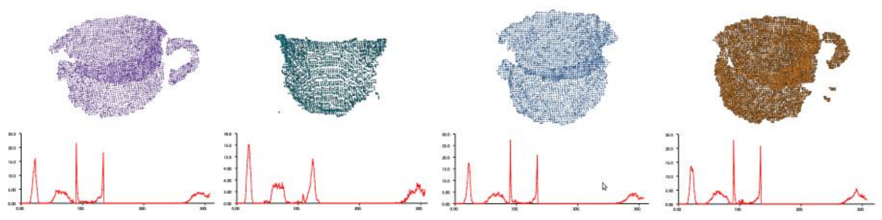


Fig. 7. Representation of several items in 3D and their corresponding View-Point feature Histogram in red. Histograms look quite similar due to their real look-alike.

3.2 Geometry filter

Subsequent filter step of the classifier is focused on geometry absolute geometric values: height, depth and width for each view. The most valuable measure is height because both width and depth are weak independent from view point and rotation angle. This makes sense because the higher part of the point cloud will always be observed by the camera sensor, and therefore successfully determine, while depth or width can be mistaken due to occlusions.

One arising problem when capturing these three values is determining the absolute position of the object according to the supporting plane global position. For this reason, it is necessary to extract the plane equation before computing the point cloud bounding box.

The parametric equation of the supporting plane has been extracted applying RANSAC over the whole point cloud and considering that only 30% of the points are outliers. Fig. 8 shows the supporting plane segmentation and object clustering and also geometry parameters (height, width and depth) computed on the projected point cloud.

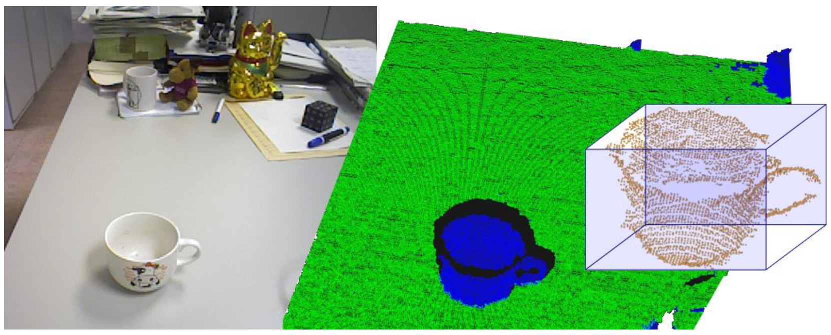


Fig. 8. Display of a real scenario with noise items in the back. Right image shows the supporting table segmentation and projected bounding box.

3.3 Color filter

Last step of the classifier is determining differences in texture between similar candidates. This last condition makes the complete architecture able to distinguish between objects with similar surface shapes and identical geometrical sizes but different colors.

There exist multiple ways of comparing textures among objects. For this paper, each color histogram R, G and B will be assumed as an independent distribution and the distance metric between objects will be chosen from one of the previously mentioned in Table 1. Each color distribution will be normalized to remove cluster size dependency.

4 Experimental results

A good way for discussing the accuracy of an object recognition algorithm with similar candidates is creating a confusion matrix that collects all of them. This kind of studies compares the whole range of classes and explains the correlation between them, intrinsic similarities and the degree of segmentation among them.

For this study, several confusion matrices will be shown. The first experiments will cover the dependency of distance matrices when comparing VFH features and RGB histograms. The last investigations will focus on the performance of the meta-classifier and its accuracy depending which single filters are applied.

4.1 Distance metric performance

In this experiment, several distance metrics have been evaluated in order to measure their performance and select the most suitable function to evaluate the distance between feature histograms. As it has been mentioned in the previous section, six classic distance functions will be used: Mahattan distance (L1), Euclidean (L2), Bhattacharyya, Kullback-Leibler divergence, Chi-Square distance and Histogram Intersection Kernel.

Their performance has been measured with a single candidate (tiny-mug) in different poses and distances relative to the sensor view point with the aim of making the trials more realistic and the experiment richer.

Table. 2. Recognition rates for different distance metrics.

L1	**L2**	**Bhattacharyya**	**KL**	X^2	**HIK**
63%	66%	46%	84%	74%	23%

The above results have been computed using the following equation

$$Performance = 100 \cdot \frac{\sum_{i=1}^{N} 1 - |o_i - r_i|}{\sum_{i=1}^{N} r_i} (\%) \qquad (5)$$

where o_i corresponds to each bin of the observed VFH histogram and r_i represents the real and stored value for this view.

Norm L1 and L2 are quite similar for this kind of distributions while HIK does not fit properly. Results show up that the best achievement is performed using Kullback-Leibler when contrasting feature histograms. This distance metric will be used in the following test.

4.2 Confusion matrices

Next experiment is focused on measuring the improvement of the filter as long as it is activated internally. That is, by means of confusion matrix it is achievable to compare detection rates for similar candidates and therefore understand the degree of confusion or precision of the filter as it is applied each of the previous stages. Three matrices have been therefore computed. Each one represents a different version of the filter. First version includes only the VFH filter, next version includes the geometric filter and the last differentiates among colors and textures.

VFH filter

Table. 3. Confusion matrix for VFH filter. Notice the high degree of confusion among scaled versions of the same objects.

	mug -tiny	**mug -medium**	**mug -big**	**mug-robot-white**	**mug-robot-red**	**mug-cow**	**teddy**
mug-tiny	**7**	41	3	19	30	0	0
mug-medium	0	**29**	26	18	27	0	0
mug-big	0	0	**84**	1	8	5	2
mug-robot-white	0	0	0	**26**	74	0	0
mug-robot-red	0	0	0	49	**51**	0	0
mug-cow	0	0	34	0	0	**59**	7
Teddy	0	0	0	0	0	0	**100**

The average recognition rate for this filter is 50.85%. Notice that it takes uniquely in account the 3D surface shape of each object, confusing objects with similar surfaces such as mug-robot-white and mug-robot-red, which are highly mixed up. The same is happening between mug-medium and mug-big, where the recognition rates are below 35%.

VFH+ Geometry filter

Table. 4. Confusion matrix for VFH+Geometry filter. Objects with similar geometry but different textures are still confused

	mug -tiny	mug -medium	mug -big	mug-robot-white	mug-robot-red	mug-cow	teddy
mug-tiny	**49**	29	0	10	12	0	0
mug-medium	0	**40**	11	11	38	0	0
mug-big	0	0	**99**	1	0	0	0
mug-robot-white	0	0	0	**77**	23	0	0
mug-robot-red	0	0	0	30	**67**	0	0
mug-cow	0	0	1	0	0	**99**	3
teddy	0	0	0	0	0	0	**100**

Second experiment takes into account not only the surface features VFH but also the global geometry of the object. With this addition, it is expected to decrease the confusion error between objects with similar shape but different size such as the mug-tiny, mug-medium and mug-big, where confusion has been demonstrated to decrease markedly. Notice that mug-cow and mug-big are now much better segmented due to their distinction. The average recognition rate for this test has been raised to 75.85%.

VFH+ Geometry + Color filter

Table. 5. Confusion matrix for VFH+Geometry+Color filter. Confusion matrix is higher diagonal stating the degree of success for the filter.

	mug -tiny	mug -medium	mug -big	mug-robot-white	mug-robot-red	mug-cow	teddy
mug-tiny	**84**	12	0	4	0	0	0
mug-medium	20	**60**	0	20	0	0	0
mug-big	3	0	**96**	0	0	1	0
mug-robot-white	0	0	0	**100**	0	0	0
mug-robot-red	0	0	0	11	**88**	0	1
mug-cow	0	0	0	0	0	**100**	0
Teddy	0	0	0	0	0	0	**100**

Last filter compares VFH features, geometric features and color texture for each object with respect to the observation. In this case, confusion matrix's diagonal is strictly higher and therefore correlation between candidates is well diminished. Mug-robot-red and mug-robot-white, which are geometrically identical but in different color, are well segmented and recognized while the rest of the distinctions rates are maintained constant as expected. The average recognition rate for this last experiment, including the three filter steps is notoriously higher reaching 89.71%.

5 Conclusions

This paper proposes an improvement of View-Point Feature Histogram based on geometrical and color texture features for 3D objects. Several experiments have been performed concluding Kullback-Leibler divergence as the most suitable selection for VFH comparisons. Furthermore, confusion matrices for the three states of the filter have been studied, correcting from the initial 50.85% of recognition rate for single VFH features to 89.71% for with confusable 3D objects.

Acknowledgements

The authors would like to thank the Spanish Government for the funds provided by the HYPER CONSOLIDER-INGENIO 2010 project and the FPU grant provided by the Spanish Education Ministry AP2009-3445. The research leading to these results has received funding from the ARCADIA project DPI2010-21047-C02-01, funded by CICYT project grant on behalf of Spanish Ministry of Economy and Competitiveness and from the RoboCity2030-II project (S2009/DPI-1559), funded by Programas de Actividades I+D en la Comunidad de Madrid and cofunded by Structural Funds of the EU.

References

Bay, H., Tuytelaars, T., & Van Gool, L. 2006. Surf: Speeded up robust features. In *Computer Vision–ECCV 2006* (pp. 404-417). Springer Berlin Heidelberg.

Bueno, J.G.; Slupska, P.J.; Burrus, N.; Moreno, L. 2011. "Textureless object recognition and arm planning for a mobile manipulator," *ELMAR, 2011 Proceedings* , vol., no., pp.59,62, 14-16.

Bueno, J. G.; Fierro, M. G.;Moreno, L.; Balaguer, C. 2012. Facial Gesture Recognition using Active Appearance Models based on Neural Evolution . *2012 Conference on Human-Robot Interaction (HRI 2012)*. Boston. USA. Mar.

Knopp, J., Prasad, M., Willems, G., Timofte, R., & Van Gool, L. 2010. Hough transform and 3D SURF for robust three dimensional classification. In Computer Vision–ECCV 2010 (pp. 589-602). Springer Berlin Heidelberg.

Liu, Y., Emery, R., Chakrabarti, D., Burgard, W., & Thrun, S. 2001. Using EM to learn 3D models of indoor environments with mobile robots. In-*MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE*-(pp. 329-336).

Martin, F., Munoz, M. L., Garrido, S., Blanco, D., & Moreno, L. 2009. L1-norm global localization based on a Differential Evolution Filter. In *Intelligent Signal Processing, 2009. WISP 2009. IEEE International Symposium on* (pp. 229-234). IEEE.

Rusinkiewicz, S., & Levoy, M. 2001. Efficient variants of the ICP algorithm. In*3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on* (pp. 145-152). IEEE.

Rusinkiewicz, S., Hall-Holt, O., & Levoy, M. 2002. Real-time 3D model acquisition. In *ACM Transactions on Graphics (TOG)* (Vol. 21, No. 3, pp. 438-446). ACM.

Rusu, R. B., Marton, Z. C., Blodow, N., Holzbach, A., & Beetz, M. 2009. Model-based and learned semantic object labeling in 3D point cloud maps of kitchen environments. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on* (pp. 3601-3608). IEEE.

Rusu, R. B., Blodow, N., & Beetz, M. 2009. Fast point feature histograms (fpfh) for 3d registration. In Robotics and Automation, 2009. ICRA'09. IEEE International Conference on (pp. 3212-3217).

Rusu, R. B., Bradski, G., Thibaux, R., & Hsu, J. 2010. Fast 3d recognition and pose using the viewpoint feature histogram. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on* (pp. 2155-2162).

Rusu, R. B. 2010. Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments. KI-Künstliche Intelligenz, 24(4), 345-348.

Scovanner, P., Ali, S., & Shah, M. 2007. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th international conference on Multimedia* (pp. 37-360). ACM.

Steder, B., Rusu, R. B., Konolige, K., & Burgard, W. 2010. NARF: 3D range image features for object recognition. In *Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)* V. 44

CAPÍTULO 7

AUTONOMOUS ROBOT FOR NAVIGATION AND INSPECTION IN OIL WELLS

G. POLETTI[1], G. EJARQUE[1], C. GARCIA[1], R. SALTAREN[1], R. ARACIL[1], and M. URDANETA[1]

[1] Centre for Automation and Robotics (CAR CSIC-UPM), Madrid, Spain
gabrielpoletti@outlook.com; gejarque@etsii.upm.es;
cecilia.garcia@upm.es; rsaltaren@etsii.upm.es; rafael.aracil@upm.es;
urdanetatoo@gmail.com

This paper proposes a novel robotic system that is able to move along the outside of the oil pipelines used in Electric Submersible Pumps (ESP) and Progressive Cavity Pumps (PCP) applications. This novel design, called RETOV, proposes a light weight structure robot that can be equipped with sensors to measure environmental variables avoiding damage in pumps and wells. In this paper, the main considerations and methodology of design and implementation are discussed. Finally, the first experimental results that show RETOV moving in vertical pipelines are analyzed.

1 Introduction

As a result of the increasing interest on climbing robots around the world, different types of climbing robots were developed for climbing over flat or curved surfaces.

Research topics on the pipe climbing robots that can move along the walls or pipes in many environments, including buildings and large ships are the most reported in the literature. These kinds of robots stick to the desired surfaces and move up or down using electromagnets (Eich and Vögele, 2011, Chung et al. 2011, Yukawa et al. 2006).

Robots for climbing inside pipes or ducts were also developed (Kwon et al, 2011, Xu et al, 2011, Choi et al, 2010) and the mobility of pipeline wheeled robots are also studied in (Park et al, 2009 & 2011, Lee et al, 2010).

In the petroleum industry there are some robotics devices to take measures of process variables inside of pipeline called Pipeline Inspection Gauges (PIG). These kinds of robots travel inside the pipe through the fluid (Hu and Appleton, 2005).

One of the most important equipment used in the oil extraction are the pumps. Submersible pumps are divided into two types:

- Surface pumps (PCP) where the motor is in the surface and the pump is inside the well (Fig.1a), and;
- Electro-submersible pumps (ESP) where the motor and pump are inside the well (Fig.1b).

These technologies have two critical issues, the first is the extreme condition under this equipment operates (high temperatures, high pressure, acid attack, etc.) and the second is the delicate balance that exists between the productions levels and the integrity of the well (extract more oil than the well can produce, could result in damages).

From Fig.1, it can be seen that a typical oil installation has a deep well made with reinforced concrete, usually called casing, and a straight pipeline is inside of it. Due to the length of the internal tube, some connectors must be used to link the pieces of pipelines.

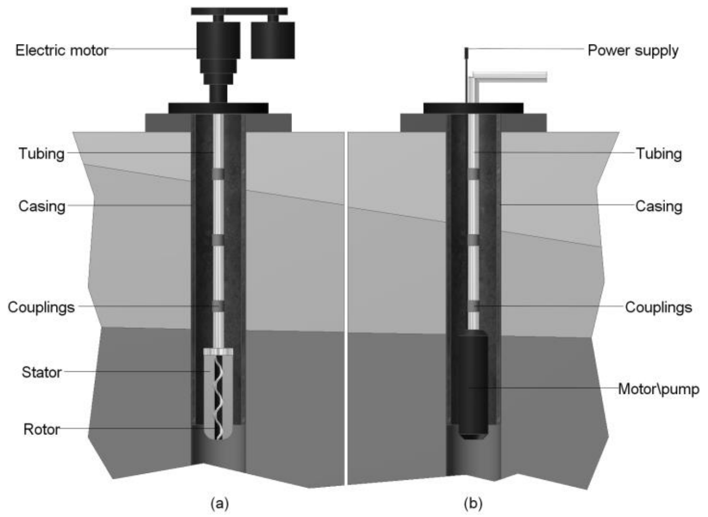


Fig. 1. (a)Progressive cavity pump (PCP); (b) Electrical submersible pump (ESP)

It is important to remark that inside of well, the environmental conditions lead to reduce useful life of the instruments and equipment (around three months). However, if internals well variables like pressure, temperature, flow of oil or oil level are not known, the maintenance tasks cannot be performing in a good way. Furthermore, the maintenance tasks in ESP and PCP systems can be made more frequently to keep them in good conditions. Therefore the maintenance tasks are expensive, tedious and dangerous.

This paper presents a novel design of a sliding robot to perform inspection and maintenance tasks in any kind of straight pipelines. Particularly, this development is carried out like an optimal solution to measurement problems and maintenance tasks in the oil industry.

Fig.2 shows a photograph of the first developed prototype. Because of the novel mechanical design, this robot can move between the casing and the oil tube. One of the most powerful advantages of this closed structure is its light weight and also its capacity to adapt around any pipeline diameters because of its modular components.

Moreover, the RETOV robot can navigate along the tube with three types of different movements: circular, straight and spiral. Using suitable combinations of them, the robot can avoid obstacles, like edges of welded joints, using a suspension mechanism located in each wheel, or change the direction of movement through its wheels.

The mechanical structure was also designed to hold the most important sensors needed for a good maintenance of the oil installations.

This article is organized as follow. Section 2 describes the most important aspects taken into account in the mechanical design of the robot. Also the hardware used in the first developed prototype is presented. Section 3 shows a kinematics model under screw theory and some simulations results that allows validating the model. Experimental results are detailed in Section 4 and; finally, conclusions and futures advances are presented.

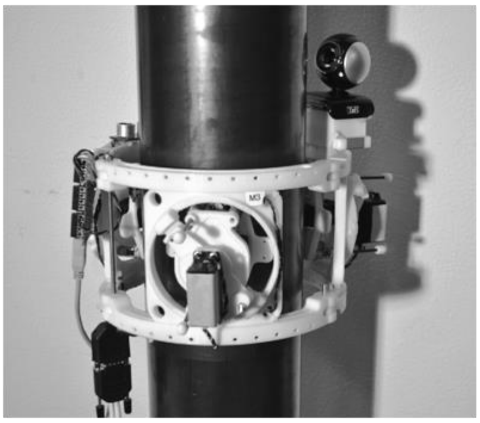


Fig. 2. Functional prototype of RETOV

2 Development of the first prototype

2.1 Design considerations

There are many restrictions to be taken into account in the mechanical design of this robot. Undoubtedly, the distance between tubing and casing (Fig.1) is the meaningful. In (Urdaneta et al, 2012) a complete analysis about all design considerations must be found. Table 1 summarizes the most common distances between the oil pipeline (tubing) and the walls of the well (casing). This space is irregular and the robot must move in-between. Therefore, the developed robot must be designed to change the movement direction toward the maximum free space.

Table 1. Annular distance between tubing and casing*

Ø TUBING [mm]	Ø CASING [mm]	ANNULAR [mm]
88.9 (3-1/2")	139.7 (5-1/2")	25.4
88.9 (3-1/2")	177.8 (7")	44.5
114.3 (4-1/2")	244.5 (9-5/8")	65.1
139.7 (5-1/2")	244.5 (9-5/8")	52.4

*Data provided by BCP Venezuela C.A.

Based on Table 1, the minimum space between tubing and casing can be determined in 25.4 mm. Therefore, no collision between robot and tubing-casing shapes is warranted when the maximum robot diameter is equal or less than 244.5 mm. This value represents a diameter of a virtual cylinder below which no mechanical interference occurs between the robot and the casing.

2.2 Mechanical Hardware

The RETOV prototype is made by two articulated rings that fix three drive wheel systems uniformly distributed along the circumference of the tubing (120º spaced, in order to have optimal force conditions).

Each drive wheel system is charged to orientate and rotate a polypropylene wheel by actuating the servo and micro motors respectively. Also a suspension formed by four compression-springs was installed in this drive system to allow obstacles avoidance like weld beads or pipe couplings.

Fig.3 shows a 3D mechanical model of this novel prototype realized in Autodesk Inventor Pro® 2012.

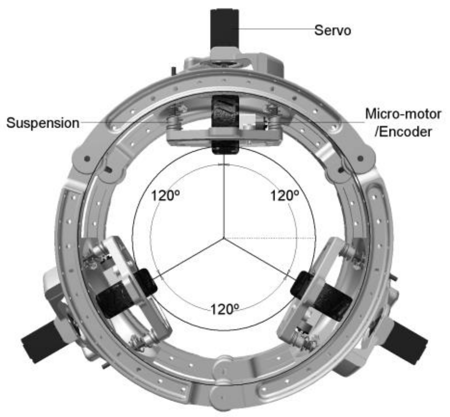


Fig. 3. 3D mechanical model of RETOV (top view)

This prototype is able to change its movement direction describing three different types of motion. Pure rotational and translational motion can be obtained independently when all three drive wheel systems are horizontally and vertically oriented respectively (servomotor at 0º and 90º respectively). Pitch variable helical movements are also possible when the same acute orientation angle is defined for all the three wheel systems.

RETOV parts were made by a Dimension Elite® 3D printer in ABS plastic. The parts of this robot are lightweight by construction but rigid by design. To accomplish the design considerations, the robot ring diameters were fixed in 220 mm and the length between two rings in 134 mm. RETOV prototype was designed to freely move over the pipe and to avoid pipeline obstacles with a height less than 8mm. The height of obstacles which the robot can handle depends on the size of the diameter of wheels used in this robot. The maximum robot diameter is, in this prototype, equal to 240 mm. This value represents a diameter of a virtual cylinder below which no mechanical interference occurs between the robot and the casing. The robot dimensions mentioned above allow it to inspect an Ø114.3 mm tubing and Ø244.5mm casing pipeline.

Table 2. Physical characteristics

Weight	1 kg
Load Capacity	0.5 kg
Maximum Size	110 x 135 mm (cylindrical r x h)
Materials	ABS plastic (first prototype)

Based on Table 2, some of the most important characteristics of the system are a total weight of 1 kg and a maximum payload capacity of 0.5 kg.

Materials used in the parts of the robot depends on the application, for the first RETOV prototype ABS plastic was used for its lightweight and construction facility. Some aluminum alloys could be used for hard environments.

This mechanical architecture allows a set of different kinds of sensors and its electronics if it is necessary. The ABS plastic rings are fabricated with a circular hole-pattern to allow the modular assembly of its components.

2.3 Control and inspection hardware

Control hardware manages the information concerning the status of the robot. The rotation and orientation of each wheel are actuated by micro-motors and servomotors respectively. Micro-motors are driven by a shield card (Ardurobotronica) which is commanded by an Arduino UNO® that controls its H-bridges. Servomotors receive a PWM signal directly from Arduino UNO®. Both cards are connected to a User-PC via USB port.

Three permanent-magnet electric micro-motors (Pololu®) with high torque planetary gearboxes (298:1), which generate a torque of 6.5 kg/cm at 6V and speed of 100 rpm were used to move the attached polypropylene wheels with high traction used on radio control cars. The motors have a quadrature encoder which are connected to the motor controller and can be used for position or velocity feedback.

Three Hitec® HS-645MG servomotors drive the orientation of the RETOV wheels. This motor can switch motion with three possibilities: 0^o for rotation around the tubing, 90^o for translational displacement, and pitch variable helical displacement of the robot when the angle varies between 0^o and 90^o degrees.

Robot position along the vertical axis (Z-displacement) is measured by a Parallax® PING sonar. The robot orientation angle (ϴ-orientation) is measured by a quadrature encoder coupling in the micro-motor shaft. An Inertial Measurement Unit (IMU) was installed to verify the real orientation in a case of wheel slipping. The connectivity of all these components is showed in Fig.4.

The inspection hardware acquires information related to the environmental conditions inside the oil well. Different types of variables as temperature, humidity, pressure and others can be sampled by sensors mounted on the structure of RETOV. According to this statement, a cordless digital camera was installed on the robot to show the conditions of the tubing surface in the user interface.

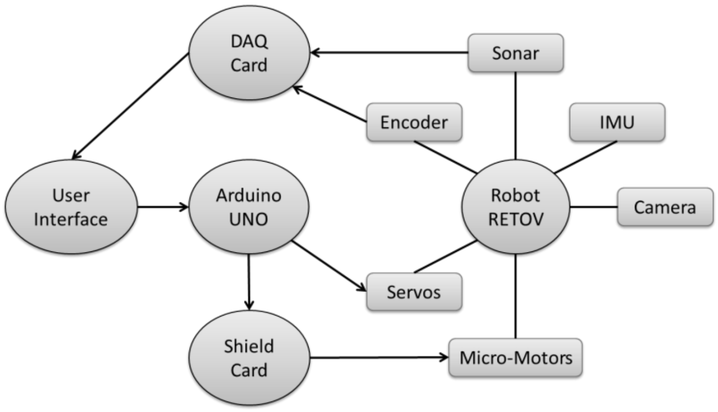


Fig. 4. Control and sensing architecture

A real deployed view of the robot is shown in Fig.5 with the mechanical structure equipped with all the embedded sensors installed on RETOV for this work.

From Fig.5, it can be observed that the robot is completely modular; that is a new segment can be easily added and then the structure can slide by pipelines with higher diameters.

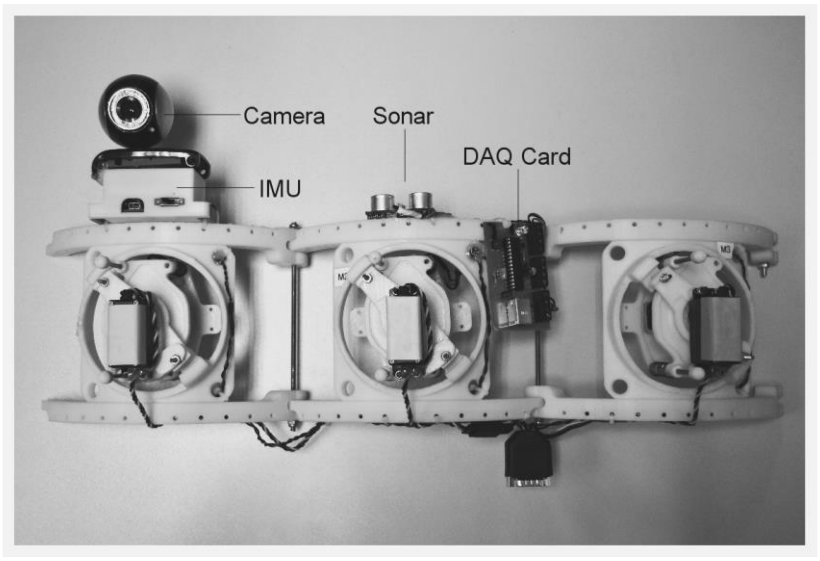


Fig. 5. Deployed view of modular RETOV

3 Kinematics model

The basic movements of the robot are rotation and translation along the same axis. These movements can be described perfectly by the screw theory, initially developed by Sir Robert S. Ball.

The kinematics model assumes constant angular velocity in the three wheels at all time. It also assumes that there is no wheel slip with respect to the tubing. It is possible to anticipate the approximate trajectory of the robot along the pipe by using the kinematics model. Therefore, wheel revolutions can be estimated base on the final location of the robot.

A screw can be defined with eight parameters which are a unit vector, a position vector, a rotated angle and a displacement along the axis. Below are the equations that describe the motion of the robot:

$$s = [s_x, s_y, s_z] \qquad (1)$$

$$s_o = [s_{ox}, s_{oy}, s_{oz}] \qquad (2)$$

$$\theta = \frac{d}{R \tan\varphi} \qquad (3)$$

Where:

s: unit vector along the direction of the screw axis.

s_o: position vector of a point on the screw axis.

θ: rotation angle about the pipe.

d: translational distance along the pipe.

φ: wheel angle.

R: radius of the pipe.

Using the formula of Rodrigues (Tsai, 1999) for a spatial displacement of a rigid body, a homogeneous transformation can be obtained as:

$${}^{A}\hat{p} = A\,{}^{B}\hat{p} \qquad (4)$$

Where:

${}^{A}\hat{p}$: a vector with the calculated location.

A : is a 4x4 transformation matrix.

${}^{B}\hat{p}$: a vector with the desired position.

To obtain the inclination that the robot must perform about the pipe, the follow equation is calculated:

$$\varphi = \arctan\left(\frac{d}{\theta R}\right) \qquad 0 \le \varphi \le \frac{\pi}{2} \tag{5}$$

Replacing the screw parameters in (4), the transformation matrix results in:

$$A = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

3.1 Kinematics model validation by simulation

A MATLAB® program has been performed to implement the kinematics model. Then for a given Z-displacement and the number of turns around the tubing, it calculates the wheel orientation angle to develop the desired maneuver.

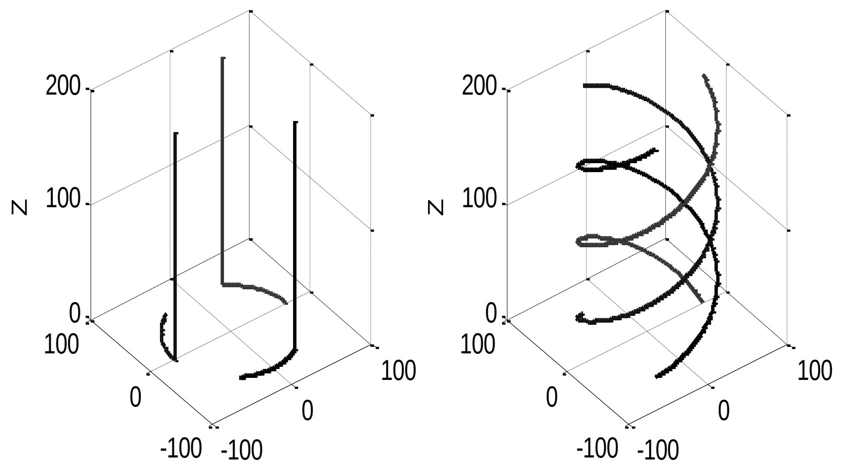


Fig. 6. Kinematics model validation results from different types of movements

Fig.6 (left) shows two basic movements that this prototype can execute independently (pure rotation and translation). Fig.6 (right) shows a pitch-variable helical movement by combining rotation and translation movements. Avoiding obstacles is possible combining these kinds of movements. Fig.7 shows a 3D model graph for better understanding of one particular wheel displacement.

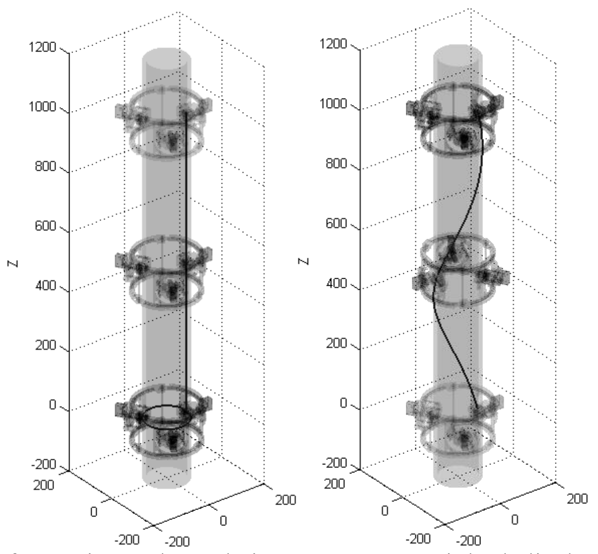


Fig. 7. Left: rotation and translation movements. Right: helical movement

4 Experimental results

4.1 User interface

To perform experimental tests, a Graphical User Interface (GUI) was developed. This application has been done under LabView® 2010.

The aim of the GUI is to get the information given by measurement system and to process it.

The GUI structure consists in five main parts, which are a 3D View, a Cam View, Environment graph, Control elements and Indicator elements. Fig.8 shows a screenshot of the GUI and its components are described below.

3D View permits to observe the route before launching the robot into the pipe and also allows visualizing the instantaneous estimated location in the pipe. CamView shows at each moment images from the environment through a cordless camera what is happening inside the well. Environment graph shows a selected variable like temperature, pressure or humidity. Control elements are buttons and sliding bars that provide interaction between operator and robot, permitting to select the type of movement (translation, rotation, helical) change speed, angle of orientation, moving direction of RETOV and start-stop execution. Finally, Indicator elements allow the operator to know about the status of the robot.

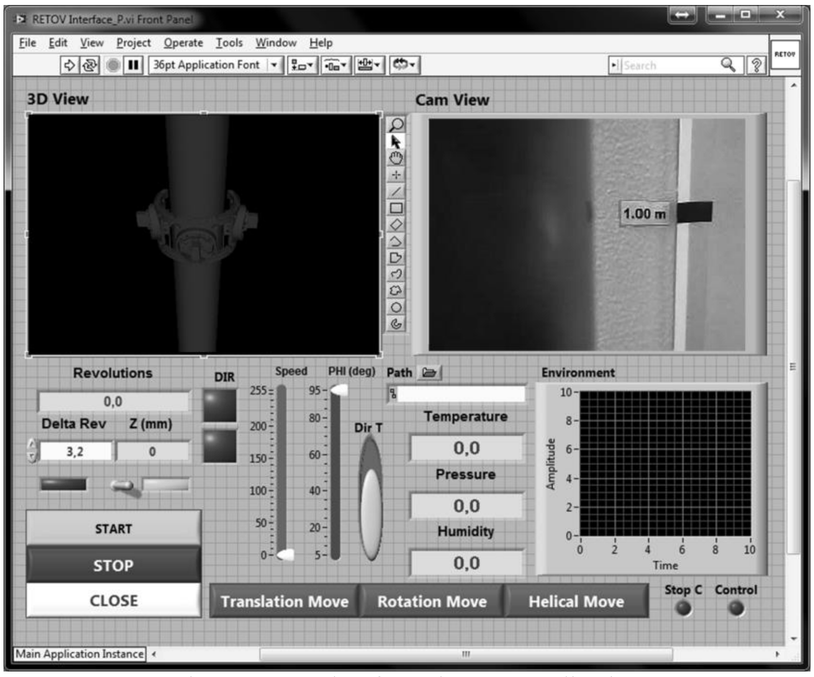


Fig. 8. Screenshot from the GUI application

4.2 Motion test

Some tests were made in a two-and-a-half-meters-high oil pipe installed at the research center. The tests show the capabilities of RETOV to develop two basic types of movement stated before and also demonstrate that the combination of these basic motions can generate a helical movement.

The following conditions were assumed: constant velocity for all the three wheels, maximum Z-displacement and ϴ-orientation were established in a meter and 360º respectively.

Three phases of navigation were performed. In the first phase, the robot described a pure rotation of 360º around the tubing. The second phase consisted in a pure counter-gravity translation of a meter. Finally a combined motion was described by doing a complete counter clockwise revolution and descending a meter at the same time. The data of three-logged tests at the three phases of motion are showed in Fig.9.

Theoretical odometry values were calculated for the three phases of movements by using the robot kinematics shown in Section 3 and programmed in MATLAB®. Practical odometry values were registered for each test in order to compare them with the theoretical ones. This information is listed in the Table 3 and the best result of the three tests is compared with theoretical calculations.

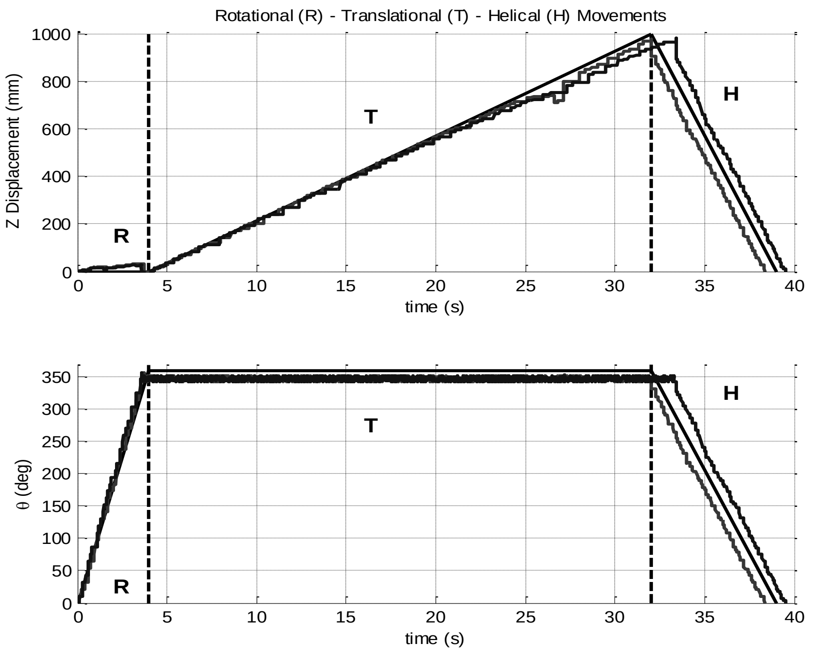


Fig. 9. Z-displacement and Θ-orientation of RETOV during three types of motion along the tubing

Table 3. Odometry results

Odometry	**Type of Movement**		
	Rotation	***Translation***	***Helical***
Theorethical	3.32 rev (360.0°)	7.58 rev (1.0 m)	8.28 rev (360.0° in 1.0 m)
Experimental	3.30 rev (357.4°)	8.30 rev (1.1 m)	7.10 rev (328.9° in 0.9 m)

Clockwise rotational movement is well tracked by the robot because rolling resistance is favorable in this situation. Counter-gravity translational movement shows a slight difference between theoretical and practical odometry values due to the reduction in the rolling resistance produces some slipping in the robot wheels. Helical descending movement presents a combination of rotational and translational wheel slipping characteristics.

The drive wheel suspension system pressed the polypropylene wheel over the tubing during the movements developed on this experiment. This pressure generated a frictional force that avoided wheel slipping. At the same time, wheel suspension adapted to unknown coaxiality of the pipe along the followed path. A snapshot sequence of a rotational movement shows the robot performance in Fig.10.

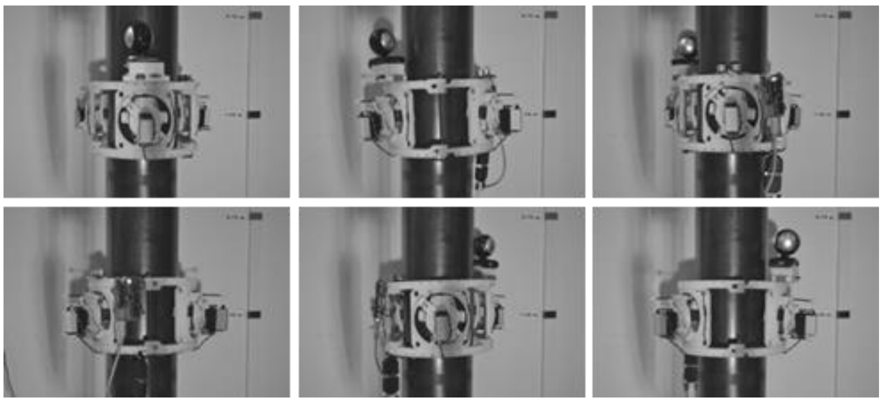


Fig. 10. Snapshot sequence of RETOV during a rotational movement

5 Conclusions

This paper presented the first developed prototype of RETOV. This robot is a patented novel design (Garcia, et al, 2012) for performing inspection and maintenance tasks in any kind of straight pipe. However this prototype is an optimal solution for service tasks in oil industry.

Because the robot can navigate between two pipes, inspection and maintenance tasks can be performed in both interior and exterior tubes. Then, just one robot can repair or inspect both tubes at the same time.

The mechanical structure allows getting all the measurement system onboard then the most critical variables in an oil well can be observed in real time. Furthermore, the mechanical design is completely modular, that is the diameter of the robot can be increased by adding new modules, then pipelines with different dimensions can be inspected by this robot.

The robot can slide on the tube because a drive wheel system that allows to perform three different types of movements. A suitable combination of them turns the robotic structure good for avoiding obstacles or reaching some objective point very fast.

The kinematics model was developed and validated by simulation first and then confirmed by experimental results.

Finally, the designed climbing robot can move, carry and measure the environment variables in the well. The potential applications of this robot are the inspection, exploration, and maintenance of the oil pipeline. Future work will primarily focused in design a robot capable of moving along horizontal, vertical and curved piping and avoiding various kinds of obstacles.

Acknowledgements

The authors are thankful for the financial support of Spanish Government CICYT Project Ref. DPI2009-08778 and also Comunidad de Madrid who support the project ROBOCITY2030-II Ref. P2009/DPI-1559.

References

Choi, C., Park, B., Jung, S. 2010. The Design and Analysis of a Feeder Pipe Inspection Robot With an Automatic Pipe Tracking System. IEEE ASME Trans. On Mechatronics. C-5(15): 736–745.

Chung, W.K., Li, J., Chen, Y., and Xu, Y. 2011. A Novel Design of Movable Gripper for Non-enclosable Truss Climbing. In IEEE International Conference on Robotics and Automation, Shanghai, China, pp. 519–525.

Eich, M. and Vögele, T. 2011. Design and Control of a Lightweight Magnetic Climbing Robot Vessel Inspection. In IEEE Mediterranean Conference on Control and Automation, Corfu, Greece, pp. 1200–1205.

García, C., Saltaren, R., Urdaneta, M. 2012. Aparato para la inspección de tuberías. Patent No ES2370897.

Hu, Z. and Appleton, E.. 2005. Dynamic Characteristics of a Novel Self-Drive Pipeline Pig. IEEE Trans. On Robotics. C-4(31): 781–789.

Kwon, Y., Lee, B., Kim, W. and Yi, B. 2011. A Flat Pipeline Inspection Robot with Two Wheel Chains. In IEEE International Conference on Robotics and Automation, Shanghai, China, pp. 5141-5146.

Lee, M.P., Ma, S., Li, B., Wang, Y. and Liu, Y,. 2010. Self-Rescue Mechanism for Screw Drive In-pipe Robots. In IEEE/RSJ International Conference on Intelligent Robot and Systems, Taipei, Taiwan, pp. 2844-2849.

Park, J., Kim, T., Yang, H.. 2009. Development of an actively adaptable in-pipe robot. In IEEE International Conference On Mechatronics. Malaga, Spain.

Park, J., Hyun, D., Cho, W., Kim, T. and Yang, H.. 2011. Normal-Force Control for an In-Pipe Robot According to the Inclination of Pipelines. IEEE Transaction On Industrial Electronic. C-12(58): 5304–5310.

Tsai, L.W. 1999. Robot Analysis: The Mechanics of Serial and Parallel Manipulators, Wiley: New York, USA.

Urdaneta, M., García, C., Saltaren, R., Contreras, G. and Ugarte, R. 2012. Estructura Robótica Pre-Tensada para Robot en Tuberías Petroleras. Revista Iberoamericana de Automática e Informática Industrial. C-2(9): 135–143.

Xu, F.Y., Wang, X.S. and Cao, P.P. 2011. The Design and Analysis of a Feeder Pipe Inspection. In IEEE International Conference on Robotics and Automation., Shanghai, China, pp. 4910-4914.

Yukawa, T., Suzuki, M., Satoh, Y., Okano, H. 2006. Design of Magnetic Wheels in Pipe Inspection Robot. In IEEE International Conference on Systems, Man, and Cybernetics, Taipei, Taiwan, pp. 235-240.

CAPÍTULO 8

ROBOTS MÓVILES PARA TAREAS DE DESMINADO HUMANITARIO

MANUEL ARMADA[1], ROEMI FERNÁNDEZ[1], HÉCTOR MONTES[1, 2], JAVIER SARRIA[1], CARLOTA SALINAS[1]

[1]Centro de Automática y Robótica CAR (CSIC-UPM); [2]Facultad de Ingeniería Eléctrica – Universidad Tecnológica de Panamá
manuel.armada@csic.es

La eliminación de minas antipersona es un problema de dimensión internacional que requiere del empleo de nuevas tecnologías tales como el desarrollo de sensores avanzados y el empleo eficiente de robots móviles. Organizaciones tales como el *International Advanced Robotics Programme* (IARP) han constituido grupos de trabajo (*Working Group HUDEM*) focalizados en el complejo problema del desminado humanitario. En este trabajo se presenta una breve revisión de diversas contribuciones y de investigaciones previas encaminadas a la solución de este problema que han sido propuestas por Grupo de Trabajo HUDEM de IARP. El trabajo recoge y analiza una amplia selección bibliográfica de trabajos de interés en este campo de desminado humanitario.

1 Introducción

El *International Advanced Robotics Programme* (IARP) es un proyecto internacional cuyo origen se remonta al *Versailles Economic Summit* de 1982. Todos los países miembros de IARP han acordado el siguiente objetivo general: "…*to foster international cooperation aiming to develop robot systems able to dispense with human exposure to difficult activities in harsh, demanding or dangerous conditions or environments". El programa IARP ha constituido un Grupo de Trabajo en desminado humanitario (*Working Group* HUDEM) en el año 2000 focalizado en este importante problema con la siguiente finalidad: "*All members of the WG*

Hudem have agreed to foster international cooperation aiming to develop performing techniques and robotics systems speeding up the demining of infested countries." A partir de un Workshop de IARP que tuvo lugar en Toulouse (Francia), una reunión en la que se dio a varios científicos la oportunidad de presentar sus trabajos en el dominio de los robots móviles de exteriores, se siguió una propuesta formal al Dr. Tom Martin (Alemania) para la constitución del grupo de trabajo HUDEM, quien organizó un primer workshop en Zimbabwe. Estos workshops iniciales fueron seguidos sistemáticamente por los siguientes workshops internacionales de IARP HUDEM (2002, Vienna; 2003 Pristina; 2004 Brussels; 2005 Tokyo; 2006, Cairo; 2007, Sousse; y desde 2009 hasta la actualidad en Sibenik, Croacia). El Prof. Yvan Baudoin, de la *Royal Military Academy* (Bélgica) ha sido el *Chairman* del *IARP Working Group HUDEM* desde 2002.

La detección y eliminación del terreno de minas antipersona es, actualmente, un problema grave con una dimensión política, social y económica notable. Hay un claro interés en la comunidad científica internacional por resolver este problema, lo que se aborda desde diferentes perspectivas y con diferentes metodologías. La detección y eliminación de terrenos infestados de minas antipersona de estos artefactos es reconocido como un problema a nivel mundial (Baudoin *et al*., 1999). Minas antipersona (AP), explosivos remanentes de guerra (ERW), y explosivos improvisados (IED) son un legado de situaciones de conflicto. Estos dispositivos pueden permanecer activos durante décadas, no son consecuentes con las negociaciones o de los tratados de paz, y no distinguen entre personal militar o civil.

Existen todavía minas AP y explosivos no explotados (UXO) de la Segunda Guerra Mundial en la mayor parte de los países de Europa y del norte de África (El-Qady *et al.,* 2005). En 2010, se han contabilizado un total de 4191 nuevas víctimas de minas, un 5% más que en 2009, y un total de 27 estados así como siete áreas en conflicto han sido confirmados o como sospechosos de estar afectados por minas (*Landmine Monitor Report*, 2011). La mejor solución a este problema, aunque quizás no la más rápida, podría ser aplicar sistemas completamente automáticos. Sin embargo, e independientemente de los últimos avances en la materia, esta solución parece estar lejos de ser posible.

Se requieren, en primer lugar, sensores más eficientes, detectores y sistemas de posicionamiento, para detectar, localizar, y si es posible, identificar las minas. En segundo lugar, el desarrollo de vehículos apropiados es de la mayor relevancia para llevar a bordo dichos sensores y desplazarlos sobre los campos infestados y, de esta manera, alejar al operador humano del riesgo directo. Dado que los sistemas completamente automáticos para esta aplicación de desminado humanitario son muy complejos y difíciles de conseguir, una solución intermedia puede ser el empleo de teleoperación y de colaboración persona-robot en el lazo de control.

Cualquier tipo de vehículo puede, en principio, llevar a cabo sensores sobre un área infestada de minas: vehículos con ruedas, con orugas e incluso vehículos con patas, pueden llevar a cabo tareas de desminado con eficiencia y seguridad. Los robots con ruedas son los más sencillos y de menor coste; los robots con orugas presentan una excelente capacidad de desplazamiento sobre casi cualquier tipo de terreno; los robots con patas exhiben asimismo un muy interesante potencial para esta actividad.

Este trabajo presenta una breve revisión de diversas contribuciones específicamente seleccionadas y de investigaciones previas relacionadas con el empleo de robots móviles para desminado humanitario encaminadas a la solución de este problema que han sido propuestas por IARP (Baudoin *et al.*, 2011; González de Santos *et al.*, 2004; Marques *et al.*, 2002). Para ello, el trabajo recoge y sintetiza la información de anteriores workshops de IARP.

2 Robots móviles para desminado humanitario

Varios workshops de IARP (Marques *et al.*, 2012) han mostrado cómo el empleo de la robótica puede mejorar de forma notable la eficiencia y la seguridad en operaciones de desminado humanitario, y cómo los sistemas robóticos de diversos tipos pueden suponer una herramienta de trabajo muy prometedora en esta dirección.

Hay tres clases principales de robots móviles investigados por la comunidad científica internacional para HUDEM: con ruedas, con orugas y con patas (también hay algunos trabajos con tracción híbrida ruedas-patas (Caltabiano *et al.*, 2004; Bostater *et al.*, 2004; Amati *et al.*, 2004)). La idea de emplear robots con patas para desminado humanitario se ha desarrollado, al menos, en los últimos 15 años, y se conocen diversos prototipos que han sido probados experimentalmente. TITAN VIII (Hirose *et al.*, 1998), AMRU-2 (Baudoin *et al.*, 1999a) y RIMHO2 (González de Santos *et al.*, 1995) son algunos de los ejemplos de plataformas de prueba para desminado humanitario que emplean robots caminantes. El robot COMET-1 ha sido quizás el primer robot con patas que se ha desarrollado específicamente para tareas de desminado (Nonami *et al.*, 2000). Estos cuatro robots se basan en configuraciones de patas tipo insecto, pero se han desarrollado otras configuraciones diferentes de robots caminantes para desminado, tales como sistemas deslizantes (Habumuremyi *et al.*, 1998), (Marques *et al.*, 2002a). El robot SILO6 del proyecto DYLEMA (González de Santos *et al.*, 2007; González de Santos *et al.*, 2002) ha sido diseñado específicamente para desminado humanitario y ha sido probado con excelentes resultados. La siguiente figura (Fig. 1), en forma de tabla, recoge algunos de los sistemas de robots móviles más relevantes en relación con la tarea de desminado humanitario.

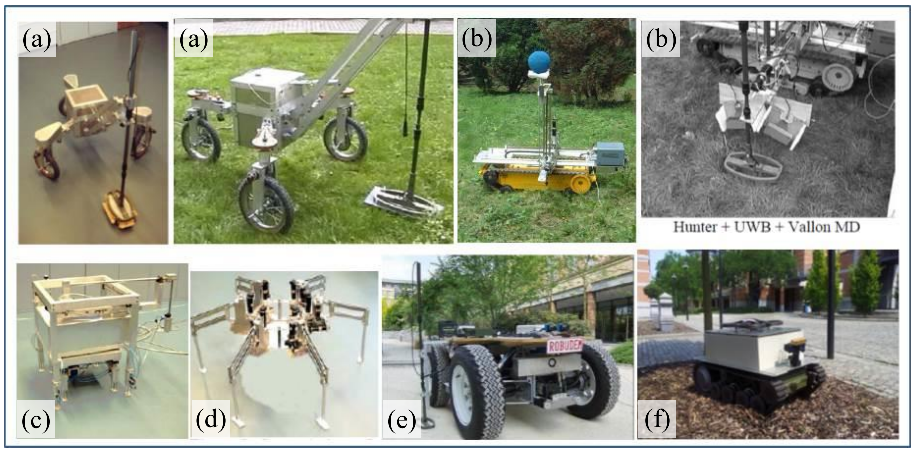


Fig. 1. (a) TRIDEM, (b) Hunter, (c) AMRU4 y (d) AMRU5, (e)ROBUDEM y (f) TEODOR/MAV. Department of Applied Mechanics. Royal Military Academy.30 Renaissance Av. 1000 Brussels, Belgium; Department of Electronics and Information Processing VUB-ETRO-IRIS, Vrije Universiteit Brussel, Pleinlaan 2, B-1050 Brussels; Service des Systèmes Logiques et Numériques, Université Libre de Bruxelles 50, Av. F. Roosevelt, B-1050 Bruxelles.

La necesidad de emplear robots caminantes en ciertas situaciones de desminado humanitario donde los sistemas con ruedas o con orugas no pueden operar de forma adecuada o presentan bajo rendimiento, ha sido investigada por la *Royal Military Academy* (RMA) y por la Universidad Libre de Bruselas. Se muestran en la Fig. 1 los prototipos de robots con patas desarrollados para llevar a bordo sensores de detección con la finalidad de probarlos en campos minados. Se muestran en dicha figura los robots TRIDEM, AMRU4 y AMRU5. Robots con orugas como el HUNTER y con ruedas como ROBUDEM también realizados por la RMA (Baudoin *et al*., 2011; Baudoin *et al*., 1999; Habumuremyi, 1998; Habumuremyi, 1998a; Habumuremyi *et al*., 2002; Baudoin, *et al*., 2003; Habumuremyi *et al*., 2004; Baudoin, *et al*., 2005; Marques *et al*., 2012).

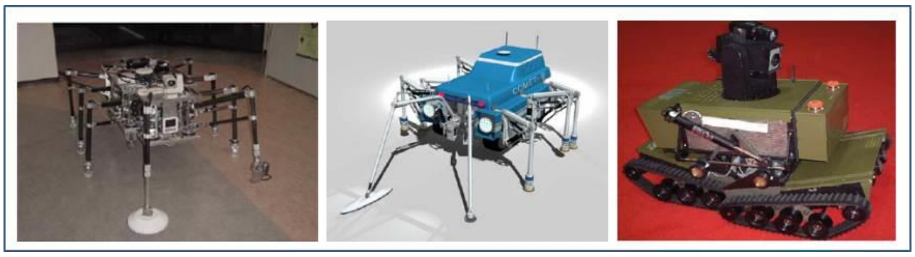


Fig. 2. COMET II, COMET III (*Dept. of Electronics and Mechanical Engineering, Chiba University, Japan*) y ROBHAZ-DT (*Advanced Robotics Research Center, KIST & Department of Mechanical Engineering, KAIST, Korea*).

La Fig. 2 muestra una vista del robot hexápodo COMET II (actuado eléctricamente) y la versión final del sistema completamente autónomo COMET III (actuado hidráulicamente), para la detección de minas antipersona, que son los dos sistemas de robots caminantes investigados por la Universidad de Chiba (Nonami, 2002). A la derecha de la Fig. 2 se muestra ROBHAZ-DT, un robot móvil con doble sistema de orugas (Munsang *et al*., 2002).

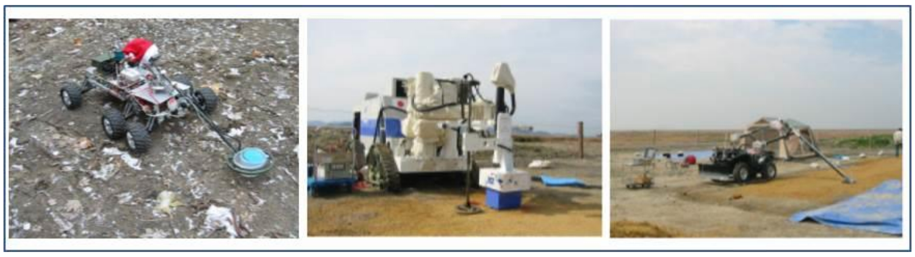


Fig. 3.- Concepto de locomoción para HUDEM (*Vienna University of Technology Division Intelligent Handling and Robotics,* Viena, Austria); y sistema para ensayar diversos detectores basados en GPR, MD, y el sistema dual ALIS (*Tohoku University*, Sendai 980-8576, Japón).

La Fig. 3 presenta un concepto de locomoción para HUDEM (Kopacek y Silberbauer, 2008) realizado por la Universidad Tecnológica de Viena, y el *Mine Hunter Vehicle*, empleado para la evaluación de GPR (*array-antenna* GPR-SAR) y para la evaluación de diversos tipos de detectores de minas: *Metal Detector* y el sistema dual ALIS (GPR+MD) realizados por la Universidad de Tohoku (Sato *et al*., 2005; Sato *et al*., 2005a).

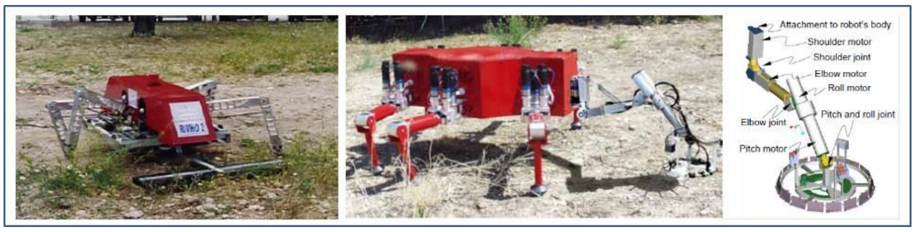


Fig. 4.- RIMHO 2 y SILO 6 (proyecto DYLEMA). Detalle del diseño del manipulador embarcado (Cento de Automática y Robótica CAR (CSIC-UPM). Ctra. Campo Real, Km. 0,200- La Poveda. 28500 Arganda del Rey, Madrid, Spain).

El Centro de Automática y Robótica (CSIC-UPM) ha llevado a cabo diversas investigaciones relacionadas con el desminado humanitario. La Fig. 4 muestra el robot cuadrúpedo RIMHO 2 y el robot hexápodo SILO 6, un sistema totalmente autónomo equipado con un brazo de cinco grados de libertad para el barrido de áreas infestadas y dotado de sensores diversos (entre ellos un detector de minas). El proyecto DYLEMA (MICINN) ha tenido como finalidad el desarrollo de robots caminantes que integren las tecnologías relevantes en locomoción, manipulación y sensores para desminado humanitario (González de Santos *et al*., 2004; González de Santos y Jimenez, 1995; Galvez *et al*., 2000; Cobano *et al*., 2008; González de Santos *et al*., 2007; Sanz *et al*., 2012; González de Santos *et al*., 2002; Armada *et al*., 2005; Marques *et al*., 2012).

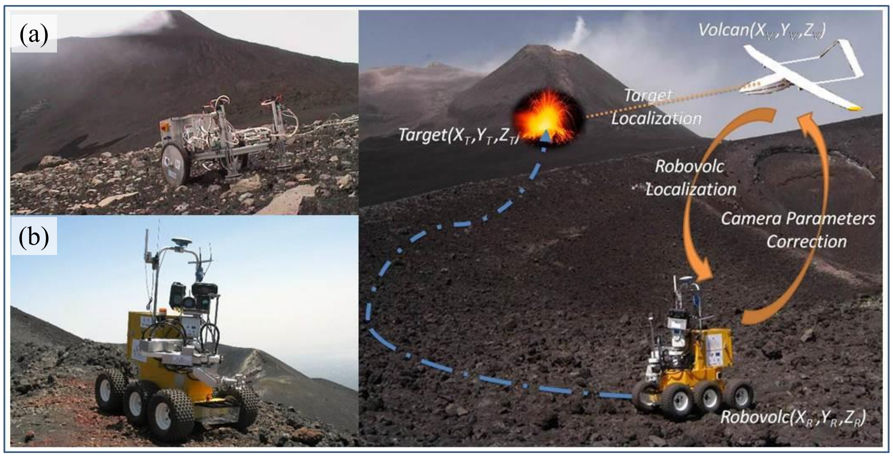

(a)
(b)
Volcan(X_V,Y_V,Z_V)
Target Localization
Target(X_T,Y_T,Z_T)
Robovolc Localization
Camera Parameters Correction
Robovolc(X_R,Y_R,Z_R)

Fig. 5. (a) WHEELEG y (b) ROBOVOLC (DIEES Università degli Studi di Catania, viale A. Doria 6, 95125 Catania, Italy).

La Fig. 5 muestra los robots WHEELEG, un robot móvil híbrido que usa ruedas y patas para la locomoción, y el robot ROBOVOLC durante la realización de pruebas en el cráter del volcán Etna. A la derecha se muestra la cooperación de robots terrestres y aéreos (UAV VOLCAN). La cooperación entre robots heterogéneos para intervenciones en medios hostiles ha sido particularmente investigada por la Universidad de Catania (Caltabiano *et al*., 2004; Bruno *et al*., 2012).

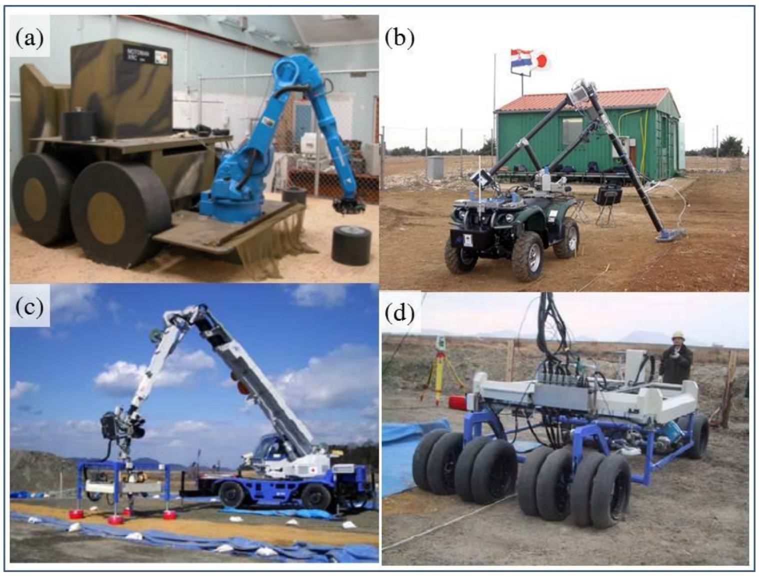


Fig. 6. (a) Semi-Autonomous Vehicle for Mine Neutralisation , (b) Humanitarian Demining Robot Gryphon, (c) Advanced Mine Sweeper, (d) Teleoperated Buggy Vehicle and Weight Balanced Arm (Systems Sciences Laboratory and Land Operations Division. Systems. DoD, Australia; Department of Mechanical and Aerospace Engineering, Tokyo Institute of Technology, 2-12-1 Ookayama, Meguro-ku, 152-8550 Tokyo, Japan; Dept. of Micro-Nano Systems Engineering, Nagoya University; Dept. of Intelligent Interaction Technologies, University of Tsukuba; Dept. of Bioengineering and Robotics, Tohoku University; National Institute of Advanced Industrial Science and Technology (AIST); Mitsui Engineering & Shipbuilding CO., LTD; Tokyo Institute of Technology, 2-12-1 Ookayama, Meguro-ku, Tokyo 152-8552, JAPAN)

La Fig. 6 muestra los robots *Semi-Autonomous Vehicle for Mine Neutralisation* (Ide *et al*., 2004), *Humanitarian Demining Robot Gryphon-An Objective Evaluation* (Cepolina y Snobar, 2008), *Environment-Adaptive Antipersonnel Mine Detection System-Advanced Mine Sweeper* (Fukuda *et al*., 2005), y *Teleoperated Buggy Vehicle* y *Weight Balanced Arm for*

*Mechanization of Mine Detection and Clearance Task*s (Fukushima *et al*., 2005). Estos robots (entre otros) han sido el resultado de las investigaciones financiadas por el Gobierno de Japón en esta área de desminado humanitario.

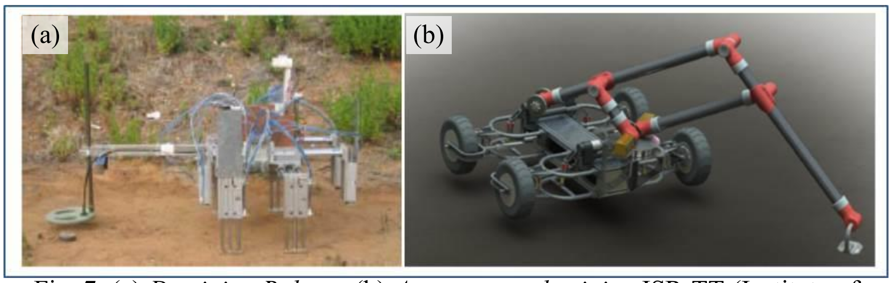


Fig. 7. (a) *Demining Robot* y (b) *Autonomous demining* ISR-TT (Institute of Systems and Robotics, University of Coimbra, Polo II, 3030-290 Coimbra, Portugal)

La Fig. 7 muestra los resultados de algunas de las investigaciones en desminado humanitario con diversas plataformas en el ISR (Instituto de Sistemas Robóticos) de la Universidad de Coimbra (Marques *et al*., 2002; Marques *et al*., 2002a; Larionova *et al*., 2011; Larionova *et al*., 2004; Marques *et al*., 2012).

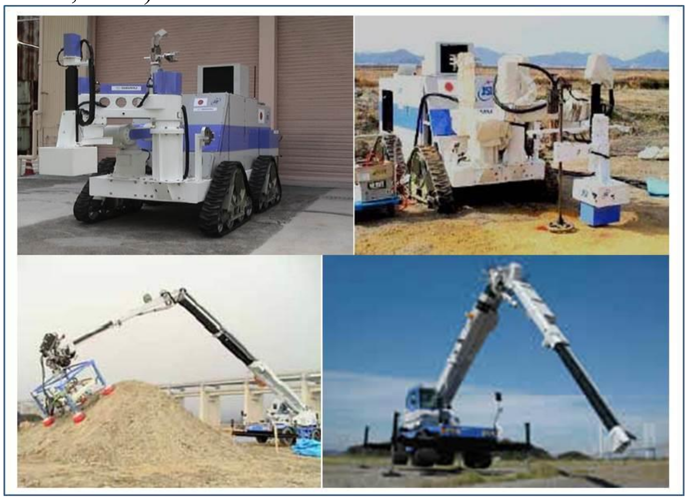


Fig. 8.- Pruebas de evaluación del empleo de GPR sobre vehículo robóticos (Japan Science and Technology Agency, Shibuya-Property-Tokyu Bldg., 10F, 1-32-12 Higashi, Shibuya-ku, Tokyo 150-0011, JAPAN; Tokyo Denki University, Ishizaka, Hatoyama-machi, Hiki-gun, Saitama 350-0394, Japan; University, 1-33 Yayoi-cho, Inage-ku, Chiba 263-8522, Japan; Fuji Heavy Industries Ltd. , 1-1-11 Yonan, Utsunomiya, Tochigi, 320-8564, Japan)

La Fig. 8 muestra las pruebas llevadas a cabo para evaluar los sistemas GPR para la detección de minas antipersona (AMS) (Ishikawa *et al*., 2005), y el robot Mine Hunter (Nonami *et al*., 2000; Nonami y Aoyama, 2005).

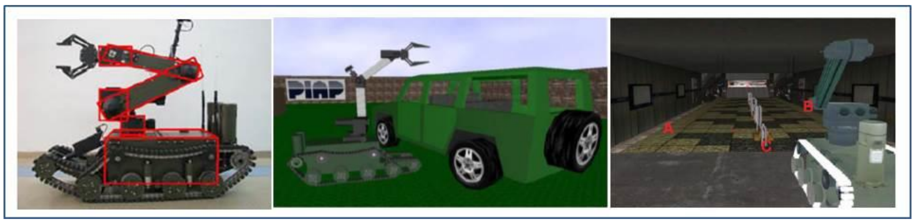


Fig. 9.- Pruebas de evaluación de sistemas de teleoperación y de simulación (*Industrial Research Institute for Automation and Measurements PIAP, Warsaw, Poland; Institute of Mathematical Machines, Warsaw, Poland; Institute of Automation and Robotics, Warsaw, Poland.*)

La Fig. 9 muestra el robot INSPECTOR empleado en tareas en entornos hostiles y detección de explosivos. La importancia de los sistemas de simulación y de entrenamiento se investigan de forma amplia en el *Institute of Mathematical Machine* y en el *Institute of Automation and Robotics* (Varsovia) (Bedkowski y Maslowski, 2008; Będkowski *et al*., 2008; Będkowski *et al*., 2012; Bedkowski y Maslowski, 2011).

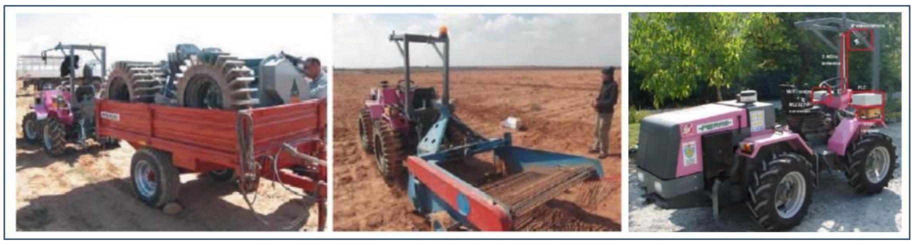


Fig. 10.- LOCOSTRA, pruebas en Jordania.

La Fig. 10 muestra las pruebas de evaluación de LOCOSTRA, sistemas para desminado humanitario que se basan en el empleo de tractores para agricultura. Estos sistemas son investigados por la Universidad de Génova, PIERRE y SNAIL AID (Cepolina y Snobar, 2008; Cepolina *et al*., 2011).

3 Conclusión

El desarrollo de robots móviles para desminado humanitario ha sido sujeto de una gran atención por el Grupo HUDEM de IARP en los últimos 10 años. En este artículo se ha presentado un resumen de los principales resultados obtenidos en este campo, referidos exclusivamente al caso de los robots móviles. Es de esperar que esta revisión sea útil para una mejor comprensión de los desarrollos más recientes en esta aplicación tan compleja. De la literatura revisada se desprende que la configuración de robots móviles más utilizada es la que emplea ruedas para su desplazamiento (Marques *et al.*, 2002a; Larionova *et al.*, 2011; Caltabiano *et al.*, 2004; Larionova *et al.*, 2004; Sato *et al.*, 2005; Fukuda *et al.*, 2005; Fukushima *et al.*, 2005; Kopacek y Silberbauer, 2008; Freese *et al.*, 2008; Bedkowski y Maslowski, 2008; Będkowski *et al.*, 2008; Będkowski *et al.*, 2012; Bruno *et al.*, 2012; Bedkowski y Maslowski, 2011; Cepolina *et al.*, 2011; Sato *et al.* 2005). La configuración de robots caminantes (Baudoin *et al.*, 2011; González de Santos *et al.*, 2004; Marques *et al.*, 2002; Baudoin *et al.*, 1999a; González de Santos y Jiménez, 1995; Nonami *et al.*, 2000; Galvez *et al.*, 2000; Cobano *et al.*, 2008; González de Santos *et al.*, 2007; Sanz *et al.*, 2012; Nonami, 2002; González de Santos *et al.*, 2002; Armada *et al.*, 2005) ha sido investigada como segunda opción, quizás a causa de la versatilidad y de las propiedades específicas de este tipo de robots. Sorprendentemente, los desarrollos basados en el empleo de vehículos con orugas han sido los menos empleados (Nonami *et al.*, 2000; Munsang *et al.*, 2002; Nonami y Aoyama, 2005).

Agradecimientos

Los autores desean reconocer la financiación recibida de la Comisión Europea (REA) en el 7PM (TIRAMISU Grant Agreement Nº 284747), la financiación de la Comunidad de Madrid en el programa Robocity2030 S-0505/DPI-0176, y especialmente la financiación recibida de la CICYT (MEC) en el proyecto Detección y localización eficiente de minas antipersonas (DYLEMA) (DPI-2001-1595). Agradecimiento especial para el Prof. Y. Baudoin (RMA) que ha organizado el *Working Group HUDEM* y los workshops de IARP. Se extiende el agradecimiento a todos los autores que han contribuido a los workshops de IARP en los últimos 10 años.

Referencias

Baudoin, Y., et al. 1999. EC Brite/Euram TN on Climbing and Walking Robots, including the Support Technologies for Mobile Robotic Machines, (CLAWAR), Year 2 Report: TASK 9, Humanitarian demining.

El-Qady, G., Sato, M., and Ushijima, K. 2005. Mine problem in Egypt: Demand for new technology. Sixth IARP Workshop HUDEM'2005, Tokyo.

Landmine Monitor Report 2011. Landmine and Cluster Munition Monitor.

Miles, R. B., Dogariu, A., and Michael, J. B. 2012. Bringing bombs to light. IEEE Spectrum, 49(2).

Baudoin, Y., Habib, M.K., Doroftei, I. 2011. Mobile robotics systems for humanitarian de-mining and risky interventions. Using robots in hazardous environments. Woodhead Publishing Limited.

Gonzalez de Santos, P., Garcia, E., Cobano, J.A., and Guardabrazo, T. 2004. Using Walking Robots for Humanitarian De-mining Tasks. In Proc. 35th ISR, Paris, Francia.

Marques, L., Rachkov, M., and Almeida, A.T. 2002. Mobile pneumatic robot for demining. In Proc. IEEE Int. Conf. On Robotics and Automation (ICRA 2002), Washington DC, pp. 3508-3513.

Hirose, S. and Kato, K. 1998. Quadruped walking robot to perform mine detection and removal task. In Proc. of the 1st International Conference on Climbing and Walking Robots, Brussels, Belgium, pp. 261-266.

Baudoin, Y., Acheroy, M., Piette, M. and Salmon, J.P. 1999a. Humanitarian Demining and Robotics. Mine Action Information Center Journal, 3(2).

Gonzalez de Santos, P. and Jimenez, M.A. 1995. Generation of discontinuous gaits for quadruped walking machines. Journal of Robotics Systems, 12(9): 599-611.

Nonami, K., Huang, Q.J., Komizo, D., Shimoi, N., and Uchida, H. 2000. Humanitarian mine detection six-legged walking robot. In Proc. of the 3rd International Conference on Climbing and Walking Robots, pp. 861-868, Madrid, Spain.

Habumuremyi, J.C. 1998. Rational designing of an electropneumatic robot for mine detection. In Proc. of the 1st International Conference on Climbing and Walking Robots. Brussels, Belgium, pp. 267-273.

Marques, L., Rachkov, M., and Almeida, A.T. 2002a. Control system of a demining robot. In Proc. of the 10th Mediterranean Conference on Control and Automation. Lisbon, Portugal.

Galvez, J.A., Estremera, J., and Gonzalez de Santos, P. 2000. SILO4-a versatile quadruped robot for research in force distribution. In Proc. 3rd International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines, Professional Engineering Publisher, U.K., pp. 371-383.

Cobano J, Ponticelli R, y Gonzalez de Santos P. 2008. Mobile robotic system for detection and location of antipersonnel land-mines: field tests. Industrial Robot: an International Journal, 35(6):520-527.

Gonzalez de Santos P, Cobano J, Garcia E, Estremera J, Armada M. 2007. A six-legged robot-based system for humanitarian demining missions. Mechatronics, 17: 417–430.

Sanz Merodio, D., Garcia, E., y Gonzalez de Santos. 2012. Analyzing energy-efficient configurations in hexapod robots for demining applications. Industrial Robot: An International Journal, 39(4).

Habumuremyi, JC. 1998a. Rational designing of an electro-pneumatic robot for mine detection. CLAWAR'98, First International Symposium, Brussels, Belgium.

Larionova, S., Almeida, A.T., Marques, L. 2011. Sensor Fusion for Automated Landmine detection on a mobile robot, Using Robots in Hazardous Environments. Woodhead Publishing.

Munsang K., et al. 2002. Development of a Mobile Robot with Double Tracks for Hazardous Environment Applications (ROBHAZ-DT). IARP WS on Robots for Humanitarian Demining. Vienna, Austria.

Kenzo N. 2002. Development of Autonomous Mine Detection Six-Legged Walking Robot for Humanitarian Demining. IARP WS on Robots for Humanitarian Demining. Vienna, Austria.

Gonzalez de Santos, P., Garcia, E., Estremera, J., and Armada, M.A. 2002. Silo6: design and configuration of a legged robot for humanitarian demining. IARP WS on Robots for Humanitarian Demining. Vienna, Austria.

Habumuremyi, J-C., Doroftei, I., y Baudoin. Y. 2002. Interest of walking robots in humanitarian demining project. IARP WS on Robots for Humanitarian Demining. Vienna, Austria.

Baudoin, Y., et al. 2003. Robotics for Humanitarian Demining: a need ? Second on-site IARP Workshop on Humanitarian demining, Prishtina, Kosovo.

Habumuremyi, J-C., Kool, P., and Baudoin, Y. 2004. Adaptive Neuro-Fuzzy Control of AMRU5, a six-legged walking robot. IARP International Workshop Robotics and Mechanical assistance in Humanitarian Demining and Similar risky interventions, Brussels-Leuven, Belgium.

Caltabiano, D., Longo, D., Muscato, G., Prestifilippo, M., Spampinato, G. 2004. The Outdoor Robotics: challenges and requirements, promising tools and dual-use applications. IARP International Workshop Robotics and Mechanical assistance in Humanitarian Demining and Similar risky interventions. Brussels-Leuven, Belgium.

Bostater, C. R., Ghir, T., Bassetti, L. 2004. Remote Sensing System for Robotics and Aquatic Related Humanitarian Demining and UXO Detection. IARP International Workshop Robotics and Mechanical assistance in Humanitarian Demining and Similar risky interventions, Brussels-Leuven, Belgium.

Amati, N., Bona, B., Canova, A., Carabelli, S., Chiaberge, M., Genta, G. 2004. Light Hybrid Robotic Platform for Humanitarian Demining. IARP International Workshop Robotics and Mechanical assistance in Humanitarian Demining and Similar risky interventions, Brussels-Leuven, Belgium.

Ide, K., Jarvis, B. J., Beattie, B., Munger, P., and Yen, L. 2004. Towards a Semi-Autonomous Vehicle for Mine Neutralisation. IARP International Workshop Robotics and Mechanical assistance in Humanitarian Demining and Similar risky interventions, Brussels-Leuven, Belgium.

Larionova, S., Marques, L., and Almeida, A.T. 2004. Feature-Level Sensor Fusion for a Demining Robot. IARP International Workshop Robotics and Mechanical assistance in Humanitarian Demining and Similar risky interventions, Brussels-Leuven, Belgium.

Baudoin, Y., et al. 2005. Mobile robotics systems for humanitarian demining – RMA Projects. IARP International workshop on Robotics and Mechanical Assistance in Humanitarian Demining, Tokyo, Japan.

Sato, M., et al. 2005. Evaluation of an array-antenna GPR system (SAR-GPR). IARP International workshop on Robotics and Mechanical Assistance in Humanitarian Demining, Tokyo, Japan.

Fukuda, T., et al. 2005. Environment-Adaptive Antipersonnel Mine Detection System - Advanced Mine Sweeper. IARP International workshop on

Robotics and Mechanical Assistance in Humanitarian Demining, Tokyo, Japan.

Ishikawa, J., Kiyota, M., and Furuta, K. 2005. Evaluation of Test Results of GPR-based Anti-personnel Landmine Detection Systems Mounted on Robotic Vehicles. IARP International workshop on Robotics and Mechanical Assistance in Humanitarian Demining, Tokyo, Japan.

Fukushima, E. F., et al. 2005. Teleoperated Buggy Vehicle and Weight Balanced Arm for Mechanization of Mine Detection and Clearance Tasks. IARP International workshop on Robotics and Mechanical Assistance in Humanitarian Demining, Tokyo, Japan.

Cruz, H., et al. 2005. Two Sustainable and Compliant Robots for Humanitarian Demining. IARP International workshop on Robotics and Mechanical Assistance in Humanitarian Demining, Tokyo, Japan.

Jeonga, H.K., et al. 2005. Development of Double-Tracked Mobile Robot with Demining System. IARP International workshop on Robotics and Mechanical Assistance in Humanitarian Demining, Tokyo, Japan.

Nonami, K., and Aoyama. 2005. Research and Development of Mine Hunter Vehicle for Humanitarian Demining. IARP International workshop on Robotics and Mechanical Assistance in Humanitarian Demining, Tokyo, Japan.

Armada, M. A., Cobano, J., Garcia, E., and Gonzalez de Santos, P. 2005. Configuration of a legged robot for humanitarian de-mining activities. IARP International workshop on Robotics and Mechanical Assistance in Humanitarian Demining, Tokyo, Japan.

Kopacek, P., Silberbauer, L. 2008. A new Locomotion Concept for Humanitarian Demining Robots. 7th IARP International workshop on Humanitarian Demining, El Cairo, Egipto.

Doroftei, D., y Baudoin. Y. 2008. Development of a semi-autonomous demining vehicle. 7th IARP International workshop on Humanitarian Demining, El Cairo, Egipto.

Freese, M., Matsuzawa, T., Aibara, T., Fukushima, E. F., and Hirose, S. 2008. Humanitarian Demining Robot Gryphon – An Objective Evaluation. 7th IARP International workshop on Humanitarian Demining, El Cairo, Egipto.

Cepolina, E., y Snobar, B. 2008. Agricultural derived tools for ground processing in humanitarian demining operations: set up of testing facility in Jordan. 7th IARP International workshop on Humanitarian Demining, El Cairo, Egipto.

Bedkowski, J., y Maslowski, A. 2008. Cognitive Theory – Based Approach for Inspection using Multi-Mobile Robot Control. 7th IARP International workshop on Humanitarian Demining, El Cairo, Egipto.

Będkowski, J., Kowalski, G., Masłowski, A. 2008. Framework for Creation of the Simulators for Inspection Robotic Systems. 7th IARP International workshop on Humanitarian Demining, El Cairo, Egipto.

Będkowski, J., Musialik, P., Masłowski, A., Baudoin, Y. 2012. Qualitative Spatio-Temporal Representation and Reasoning Framework for RISE mobile robot's operator training planning. 10th International IARP Workshop HUDEM'2012, Šibenik, Croatia.

Bruno, C., Longo, D., Melita, D., Muscato, G., Sessa, S., Spampinato, G. 2012. Heterogeneous robot cooperation for interventions in risky environments. 10th International IARP Workshop HUDEM'2012, Šibenik, Croatia.

Bedkowski, J., Maslowski, A. 2011. Semantic simulation for mobile robot operator training. IARP HUDEM'2011 Workshop, Šibenik, Croatia.

Cepolina, E., Zoppi, M., Polentes, G., Snobar, B., Abel, F., Kasesbeh, B. 2011. In-field tests of LOCOSTRA in Jordan. IARP HUDEM'2011 Workshop, Šibenik, Croatia.

Marques, L., Almeida, A. T., Armada, M., Fernández, R., Montes, H., González, P., y Baudoin, Y. 2012. State of the Art Review on Mobile Robots and Manipulators for Humanitarian Demining. IARP HUDEM'2012 Workshop, Šibenik, Croatia.

Sato, M., et al. 2005a. Evaluation of a Hand-held GPR MD sensor system (ALIS). IARP International workshop on Robotics and Mechanical Assistance in Humanitarian Demining, Tokyo, Japan.

CAPÍTULO 9

GENERACIÓN DE COMPORTAMIENTO AUTÓNOMO PARA UN ROBOT DE INSPECCIÓN DE CULTIVOS

J. M. BENGOCHEA-GUEVARA y A. RIBEIRO

Centro de Automática y Robotica (CSIC-UPM) jose.bengochea@csic.es, angela.ribeiro@csic.es

Este artículo presenta un robot cuyo objetivo es realizar la inspección de un campo navegando de forma autónoma. La estrategia propuesta para generar los distintos comportamientos o conductas del robot se basa en una red de nodos de diferente funcionalidad: nodos perceptivos, nodos cognitivos y nodos actuadores. Las distintas conductas se forman mediante la interconexión de estos nodos, siendo su número variable en función de la mayor o menor complejidad del comportamiento. Las conductas generadas pueden ser innatas o aprendidas mediante condicionamiento instrumental, creándose comportamientos más complejos. La conducta del robot que emerge en cada momento de entre su catálogo de comportamientos depende de las metas y necesidades que el robot deba satisfacer en cada instante de tiempo. De esta forma el robot genera un comportamiento autónomo que le permite inspeccionar el campo sin intervención humana.

1 Introducción

Tradicionalmente, las prácticas de cultivo se han orientado hacia una gestión uniforme del campo, ignorando la variabilidad tanto espacial como temporal que aparece en el mismo. Esto presenta fundamentalmente dos tipos de inconvenientes: a) contaminación de la atmósfera y del suelo, con la consecuente contaminación de las aguas subterráneas y b) un notable aumento en el coste económico de la producción agrícola.

Además en los próximos 25 años será necesario doblar la producción agrícola disponiendo cada vez de menos suelo y agua lo que conduce a una situación en la que es crucial la generación de tecnología que permitan minimizar los costes de producción a la vez que se realiza una gestión respetuosa con el medio ambiente.

En los últimos años, gracias al desarrollo de tecnologías como los Sistemas de Posicionamiento Global por satélite (GPS), sensores de cosecha, sensores de humedad o de fertilidad del suelo, sensores multiespectrales, sistemas de teledetección, Sistemas de Información Geográfica (SIG/GIS) y Sistemas de Soporte a la Decisión (SSD), ha surgido el concepto de Agricultura de Precisión (AP) que propone una gestión del cultivo basada en la existencia de variabilidad en el mismo.

Dentro de la AP tienen especial importancia las técnicas orientadas a la aplicación selectiva de tratamientos, que en contraposición a los métodos agrícolas tradicionales, proponen aplicar herbicida sólo en aquellas zonas del cultivo infestadas, variando además la cantidad de tratamiento aplicado según la densidad y tipo de malas hierbas.

Para llevar a cabo una aplicación selectiva de tratamientos, el primer paso es la recogida de información en el campo que permita conocer la distribución de las malas hierbas y la variabilidad espacial de cada especie (Percepción), necesarias para calcular las necesidades, en términos de herbicida, de cada unidad de terreno (Senay, 1998). Una vez recogida dicha información el siguiente paso es tratar la misma para determinar la mejor actuación (Toma de Decisión) y finalmente el último paso es la realización de la aplicación selectiva de tratamiento (Actuación).

La etapa de Percepción no es necesaria en un tratamiento convencional en el que todo el campo se trata de forma uniforme con la misma dosis de herbicida. Por tanto es esencial automatizar esta etapa al máximo y lograr que sea lo más barata posible si se quiere llegar a implantar de forma efectiva un esquema de Agricultura de Precisión. Entre los distintos medios que se pueden idear para recoger información del campo se encuentran los vehículos autónomos con elementos de sensorización embarcados y dimensiones reducidas para disminuir el impacto sobre el cultivo y la compactación del terreno.

Este artículo presenta un robot cuyo objetivo es realizar la inspección de un campo navegando de forma autónoma. Basándose en ideas tomadas de la etología y de la psicología cognitiva, se propone una red de nodos de diferente funcionalidad que guiados por la metas y necesidades del robot, generen en conjunto un comportamiento autónomo que lleve al robot a inspeccionar el campo sin intervención humana.

2 Objetivo

El robot utilizado para la inspección del campo es un modelo comercial (mBase-MR7 de MoviRobotics) de cuatro ruedas con locomoción diferencial, sin ruedas directrices y que puede girar libremente alrededor de su eje vertical (Fig. 1.).

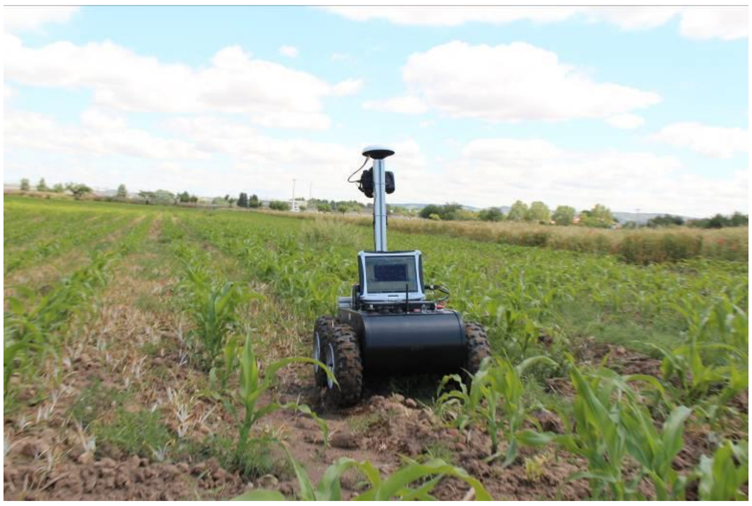


Fig. 1. Robot mBase-MR7 recorriendo el campo

La disposición típica de un campo de cultivo se puede ver en la Fig. 2. Para inspeccionar una línea de cultivo, el robot se sitúa al inicio de ella, colocándose de forma que la línea se ubique entre sus dos ruedas, y comienza a avanzar, siguiendo la línea de cultivo gracias a la cámara que lleva a bordo. El avance prosigue hasta llegar al final de la línea. Una vez recorrida esa línea, realiza las maniobras necesarias para situarse al inicio de la siguiente línea que quiere inspeccionar. El procedimiento se repite hasta que se inspeccionan todas las líneas que conforman el cultivo. Ese es el objetivo de este robot, realizar la inspección con cobertura total del campo navegando de forma autónoma.

La inspección del campo con cobertura total requiere disponer de un conjunto de conductas individuales como: 1) una conducta de seguimiento de línea, 2) una conducta de detección de fin de línea, y 3) una conducta de cambio de línea para posicionarse correctamente en la cabecera de la siguiente línea. Además debe estar dotado de 4) una conducta de detección y sorteo de obstáculos para evitar la colisión, y 5) una conducta de supervisión del nivel de las baterías, de forma que cuando detecta que la alimentación está próxima a agotarse, el robot se dirija al punto de recarga más cercano. Para lograr que el robot se dirija a un punto determinado situado en los bordes del campo de cultivo, como puede ser un punto de recarga o de inicio del recorrido de inspección, el robot debe tener una conducta que le guíe, con ayuda de un GPS, desde un punto del borde del campo (laterales y cabeceras del cultivo), hasta otro punto del mismo borde. Éstas y futuras conductas que se podrán ir progresivamente incorporando, son las conductas que debe ser capaz de generar de forma coordinada la red de nodos propuesta en este artículo.

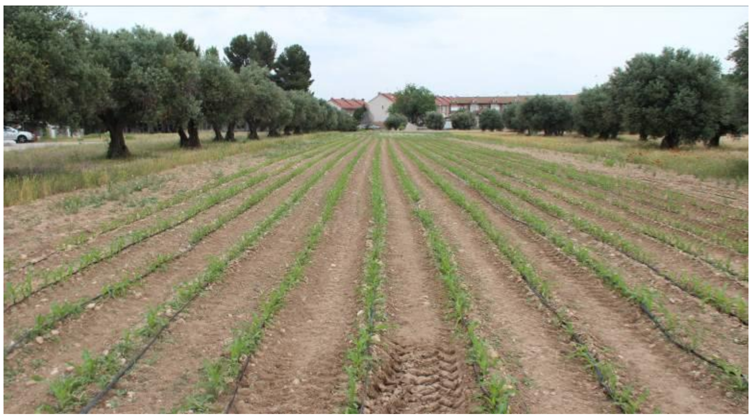


Fig. 2. Disposición típica de un campo de cultivo de maíz

3 Nodos

En esta sección se introduce el concepto de nodo, componente fundamental de la red que genera los comportamientos del robot. Un nodo es una unidad básica, con una funcionalidad determinada, cuya interconexión con otros nodos forma conductas. Existen tres tipos de nodos: nodos perceptivos, nodos cognitivos y nodos actuadores. Se ha realizado esta división basándose en los fundamentos de la psicología cognitiva, y rechazando las ideas de la psicología conductista, que no tiene en cuenta los elementos deliberativos, y explica la conducta como simples reflejos acción-reacción (Bermúdez Moreno, 2011). Los nodos perceptivos se encargan de procesar la información procedente del mundo o del propio estado del robot. Están conectados a los diferentes sensores del robot, procesando la información que éstos les facilitan, y generando como producto de este procesamiento una información que utilizan otros nodos. Por ejemplo, la conducta de seguimiento de línea de cultivo comienza con un nodo perceptivo encargado de procesar la información procedente de la cámara detectando la posición de las líneas de cultivo en la cobertura vegetal que aparece en la imagen. Esta información se transmite al siguiente nodo cognitivo. Los nodos cognitivos son los elementos deliberativos, reciben información procedente de otros nodos y deciden cómo actuar en función de esa información. Siguiendo con el ejemplo anterior, el nodo cognitivo recibe la información de la posición de las líneas de cultivo y decide cómo actuar, corrigiendo, si es necesario, la navegación del robot activando los nodos actuadores para que el robot no pise las líneas de cultivo. Los nodos actuadores son los que materializan la conducta, están conectados a los actuadores del robot. Son nodos normalmente sencillos, que llevan a cabo la actuación correspondiente cuando se activan. Para finalizar con el ejemplo, en esta conducta de seguimiento de línea se dispone de cuatro nodos actuadores: uno que incrementa la velocidad del robot, otro que la reduce, otro que actúa girando el robot hacia la izquierda, y el último que hace que gire a la derecha. Según cuál active el nodo cognitivo, se actúa de una forma u otra. En la Fig. 3 se pueden ver los nodos que forman la conducta seguir línea (de aquí en adelante, los nodos perceptivos se representarán con triángulos, los nodos cognitivos con círculos y finalmente los nodos actuadores con cuadrados).

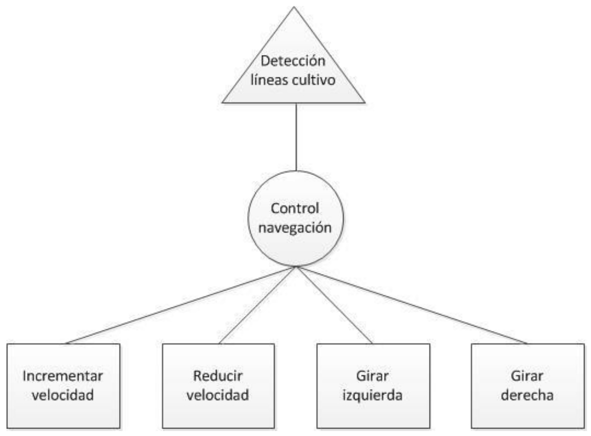


Fig. 3. Conducta seguimiento de línea de cultivo

4 Red de nodos

Para generar comportamiento autónomo en el robot, se construye una red de nodos como los presentados anteriormente. El objetivo es conseguir generar un comportamiento autónomo del robot mediante esta estructura de componentes sencillos, de forma que sea fácil interconectar unos nodos con otros y utilizar los mismos nodos para diferentes conductas. La estructura de esta red también permite que sea asequible sustituir unos nodos por otros respondiendo de esta forma a posibles cambios en la dotación sensorial del robot o a la incorporación de otras estrategias de percepción. La sencilla sustitución de nodos también responde bien cuando hay que variar la capacidad deliberativa de los nodos cognitivos, bien porque se cambian los actuadores del robot y hay que reemplazar los nodos actuadores. Otro requisito de esta red es que facilite la incorporación de nuevos comportamientos al repertorio de conductas del robot. Con el tipo de red utilizado, la incorporación se consigue de forma sencilla añadiendo nuevas conexiones a la red, entre nodos ya existentes o entre nuevos nodos que también se agregan a la red, y añadiendo esta nueva conducta a la jerarquía de motivaciones del robot tal como veremos en el apartado 5.

Las conductas se forman por la interconexión de nodos. La comunicación de información es unidireccional, los nodos perceptivos pueden suministrar información a nodos cognitivos o a nodos actuadores, pero no en sentido contrario, y los nodos cognitivos pueden enviar información a nodos actuadores, pero no al revés. Los nodos pueden estar activos o inactivos (para no perder capacidad de procesamiento). La activación de los nodos se produce a través de la interconexión que hay entre ellos. Los nodos perceptivos son capaces de activar los nodos cognitivos y actuadores que hay conectados a ellos. Asimismo los nodos cognitivos, como centros neurálgicos de la conducta, son capaces de activar los nodos perceptivos, cognitivos y actuadores conectados a ellos. En el momento en que un nodo perceptivo activo tenga información que transmitir, la manda por todas sus conexiones, activando a la vez los nodos receptores para que sean capaces de procesar la información percibida. Un nodo cognitivo activo, en cambio, tiene la capacidad de elegir qué nodos activar o a cuáles transmitir información de entre todos los nodos conectados a él.

Como ya se ha dicho, una conducta está formada por la unión de varios nodos. Esos mismos nodos, o un subconjunto de ellos, pueden formar parte de otras conductas. De esa manera, se van interconectando los nodos y se forma la red. La mayoría de las conductas estarán formadas por nodos perceptivos, cognitivos y actuadores, pero con esta estructura también se pueden implementar conductas reflejas, es decir de acción-reacción, propias de componentes reactivos y formadas únicamente por nodos perceptivos y nodos actuadores. Además es posible construir conductas plenamente deliberativas, es decir que no tengan en cuenta la percepción, y por tanto formadas por nodos cognitivos. Este tipo de conductas pueden utilizarse para por ejemplo proponer un plan que posteriormente unos nodos actuadores pueden llevar a cabo.

En una ejecución normal de la red existirán varios nodos activos ejecutándose en paralelo. Éstos pueden formar parte de la misma conducta, por ejemplo, nodos perceptivos que están captando información a la vez que se está deliberando sobre la información percibida anterior y al mismo tiempo que nodos actuadores estén actuando sobre el mundo. Pero también pueden formar parte de distintas conductas. Esto se debe a que el robot puede estar en alerta, prestando atención a distintos estímulos del entorno o de su estado interno, para poder reaccionar adecuadamente en el caso de que se presente el estímulo adecuado. Estas conductas en alerta, característicamente tienen activos únicamente los nodos perceptores correspondientes, pasando sólo a activarse el resto de la conducta cuando se perciban los estímulos adecuados. En la Fig. 4 se muestra un fragmento de una red, que incluye las conductas de seguimiento de línea de cultivo y de detección de fin de línea.

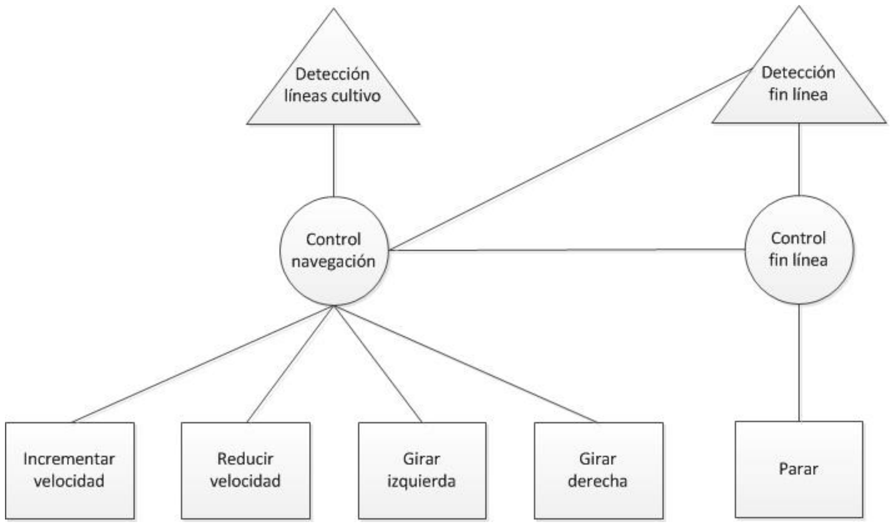


Fig. 4. Fragmento de red de nodos

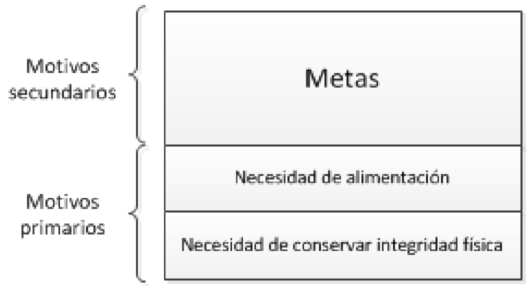
5 Motivos

Este apartado pretende dar respuesta a la cuestión de cómo emerge un comportamiento autónomo en el robot a partir de la red de nodos explicada anteriormente, sin que exista conflictos e interferencias entre las diferentes conductas, y se produzca un comportamiento coordinado y correcto.

El motivo puede definirse como una fuerza interna impulsora que activa al organismo y dirige sus acciones (Sanz Aparicio, 2011). La clasificación más básica entre motivos es la que distingue entre motivos primarios y motivos secundarios. Los motivos primarios son aquellos que están relacionados con la supervivencia de los individuos, mientras que los motivos secundarios se consideran como metas o motivos en sí mismos, que movilizan al sujeto hacia la acción. Desde este punto de vista, se entiende que los motivos secundarios son los objetivos o metas que surgen desde el interior del propio sujeto y que tienen en sí mismo un significado, una dirección y una intención o propósito.

En los animales, cuando se produce una necesidad asociada con un motivo primario, toda la motivación del sujeto se dirige a realizar de forma inmediata la conducta que satisfaga esa necesidad. Así, cuando un animal tiene hambre, desata una conducta apetitiva destinada a saciar esa necesidad. Esta forma de actuar también es la propia del robot. Cuando el robot detecte que se está quedando sin alimentación en sus baterías, dejará lo que esté haciendo y elicitará una respuesta destinada a saciar su "hambre", buscando el punto de recarga más cercano donde pueda cargar sus baterías.

Para que el robot tenga un comportamiento adecuado a sus motivaciones y no existan conflictos entre unas y otras, se propone una jerarquía de constructos motivacionales (Fig. 5.), con los motivos primarios en los niveles inferiores y los motivos secundarios en los superiores. Se organizan todos los motivos desde un nivel inferior a un nivel superior, de forma que sólo pueden satisfacerse los niveles superiores, cuando los inferiores están satisfechos. Como motivos primarios del robot se encuentran la necesidad de conservar su integridad física y la necesidad de alimentación. La primera de éstas se refiere a la necesidad de reaccionar adecuadamente cuando se encuentra algún obstáculo en el camino, puesto que si no atiende esa necesidad de forma inmediata supondrá que el robot choque con ese obstáculo y eso puede significar que el robot quede seriamente dañado. La necesidad de alimentación se refiere a la necesidad de buscar un punto donde recargar las baterías cuando se esté quedando sin alimentación. La necesidad de conservar la integridad física se sitúa en el nivel más bajo de la jerarquía, ya que es la necesidad más importante para el robot (si el robot acaba destrozado por un obstáculo, el resto de necesidades desaparecen), situándose la necesidad de alimentación en el nivel inmediatamente superior. Todos los motivos primarios tienen sus conductas en alerta, para poder prestar atención a los estímulos del entorno (obstáculos) o de su estado interno (nivel de las baterías), para poder reaccionar adecuadamente en el caso de que se presente el estímulo adecuado. Como ya se comentó anteriormente, estas conductas en alerta, característicamente tienen activos únicamente los nodos perceptores correspondientes, pasando sólo a activarse el resto de la conducta cuando se perciban los estímulos adecuados.


Motivos secundarios
Metas
Motivos primarios
Necesidad de alimentación
Necesidad de conservar integridad física
Fig. 5. Jerarquía de constructos motivacionales

Dentro de los motivos secundarios del robot, se encuentran las metas que tiene. Éstas están organizadas jerárquicamente de manera que las metas de alto nivel se persiguen mediante metas de niveles más bajos. En nuestro ejemplo, el robot tiene como máxima meta recorrer el campo de cultivo entero. Para ello debe ir recorriendo, sucesivamente, las distintas líneas que componen el cultivo, que constituyen metas de un nivel inferior. A su vez, recorrer una línea de cultivo supone una serie de metas de nivel inferior, como son posicionarse correctamente al inicio de la línea, seguir la línea sin pisar el cultivo, y llegar al final de la línea de cultivo (Fig. 6.). El fundamento está en descomponer las metas en metas cada vez más simples, de forma que según se vayan satisfaciendo las metas de los niveles inferiores, se irán gradualmente satisfaciendo las metas de niveles superiores. Dentro de cada nivel, las metas se persiguen de izquierda a derecha según están situadas en la jerarquía.

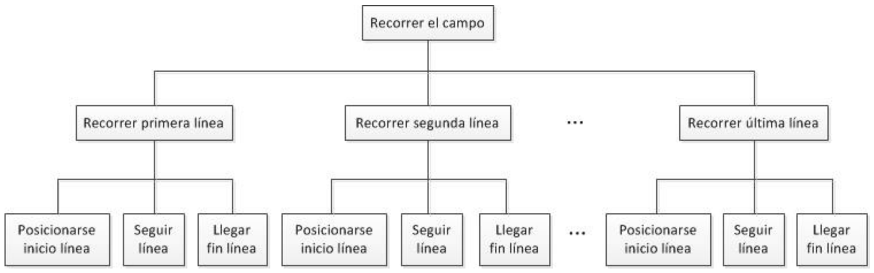


Fig. 6. Ejemplo de jerarquía de metas

Los nodos cognitivos de las conductas que se están llevando a cabo para satisfacer las metas son los encargados de poner en alerta la conducta encargada de alcanzar la siguiente meta, para que esté preparada cuando se satisfaga la meta actual, así como de desactivar su conducta cuando logra la meta que se había propuesto el robot.

Cuando se esté realizando una conducta para satisfacer una meta y aparezca una necesidad primaria, se atenderá ésta, y luego, si es posible, se continuará la meta que se estaba satisfaciendo. Por ejemplo, si aparece un obstáculo en el camino, el robot se parará, pero si pasado un tiempo determinado el obstáculo desaparece, se proseguirá con la meta que se quiere satisfacer. Si, por el contrario, no es posible continuar con la meta que se estaba satisfaciendo, se empezarán a satisfacer otra vez las metas de ese mismo nivel pertenecientes a la misma meta del nivel inmediatamente superior. Por ejemplo, si el robot mientras está siguiendo la segunda línea de cultivo, se ve obligado a abandonar la inspección para buscar un punto de recarga ya que se está quedando sin alimentación, la meta superior de recorrer la segunda línea de cultivo se habrá quedado incompleta. Entonces, cuando el robot acabe de recargarse, tendrá que volver a satisfacer la meta de posicionarse correctamente al inicio de la segunda línea para posteriormente seguirla.

Con esta jerarquía de constructos motivacionales que guían el comportamiento del robot se consigue un comportamiento autónomo del robot sin conflicto entre las conductas.

6 Condicionamiento instrumental

Como los estudios etológicos han demostrado (Abril Alonso, 2009), en los animales existen comportamientos innatos, conductas predeterminadas genéticamente que han evolucionado con la especie y les han permitido adaptarse a su entorno y facilitar su supervivencia. En el robot, conductas innatas serían todas aquellas que el programador ha desarrollado, que ha escrito en los “genes” del robot. En contraposición a estas conductas, están las conductas aprendidas, que son aquellas que el animal adquiere a través de la experiencia. Un tipo de aprendizaje es el condicionamiento instrumental, una forma de aprendizaje asociativo en el que se desarrollan nuevas conductas en función de sus consecuencias (Domjan, 2007). Se puede resumir en la ley del efecto establecida por Thorndike: “Cualquier conducta que en una situación produce un efecto satisfactorio, se hará más probable en el futuro”. Por ejemplo, si introducimos una rata en una caja con una palanca que al ser pulsada hace que aparezca alimento para el animal, al principio la rata realiza conductas aleatorias hasta que, por azar, pulsa la palanca y consigue el alimento. En un primer ensayo la rata no establece la relación entre la pulsación de la palanca y la obtención del alimento, pero a lo largo de varios ensayos, la rata aprende la relación y llega un momento en que al introducir la rata en la caja, ésta se dirige a la palanca para pulsarla y conseguir el alimento. La conducta sirve de "instrumento" para lograr un fin y se aprende a través de un mecanismo de prueba y error.

Basado en estos fundamentos, la idea que se pretende es que el robot adquiera una conducta más compleja a partir de una secuencia de conductas elementales. En concreto, se quiere probar este tipo de aprendizaje para que el robot adquiera la conducta de cambio de línea de cultivo a partir de conductas simples como son desplazarse hacia adelante, desplazarse hacia atrás, girar a la izquierda y girar a la derecha. El robot cuando termina de recorrer una línea de cultivo y llega a su final, necesita realizar una serie de maniobras para posicionarse correctamente al inicio de la línea siguiente que se quiere recorrer (Fig.7.). La distancia entre las líneas de cultivo es siempre la misma, por lo que las maniobras a realizar son siempre las mismas. Se pretende que el robot adquiera una conducta compleja a partir de un conjunto de conductas simples innatas. Para ello, se generarán secuencias aleatorias combinando estas conductas, de forma que aquellas combinaciones que logren su meta (posicionarse correctamente al principio de la siguiente línea), se verán reforzadas positivamente, siendo el refuerzo mayor cuanto menor sea el tiempo en conseguir el posicionamiento. De esta forma, cuando finalice el entrenamiento, el robot llevará a cabo la conducta de cambio de línea de cultivo que mayor refuerzo haya obtenido, pasando a formar parte del conjunto de conductas del robot. Es importante señalar que para un mismo tipo de cultivo (maíz, girasol, etc.) las líneas se siembran manteniendo siempre la misma distancia, por lo que aunque la etapa de entrenamiento pueda ser costosa está se compensa ampliamente porque la conducta aprendida será aplicable a cualquier cultivo del mismo tipo.

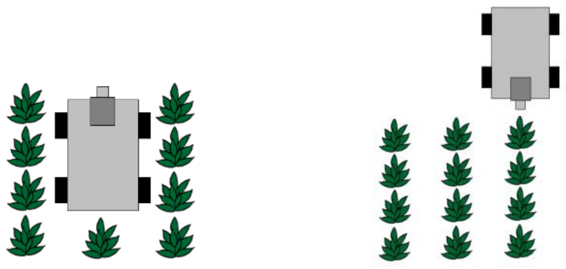


Fig.7. Posición de inicio para la maniobra (izquierda) y posición final de la maniobra (derecha). Generación de la conducta cambio de línea.

Agradecimientos

Este trabajo está financiado por el Ministerio de Economía y Competitividad a través del proyecto del Plan Nacional: AGL2011-30442-C0-02 (GroW).

Referencias

Abril Alonso, A. et al. 2009. Fundamentos de Psicobiología, Sanz y Torres: Madrid.

Bermúdez Moreno, J. et al. 2011. Psicología de la Personalidad, UNED: Madrid.

Domjan, M. 2007. Principios de Aprendizaje y Conducta, Thomson Paraninfo: Madrid.

Sanz Aparicio, M.T. et al. 2011. Psicología de la Motivación, Sanz y Torres: Madrid.

Senay, G.B. et al. 1998. “Manipulation of high spatial resolution aircraft remote sensing data for use in site specific farming”. Transactions of the American Society of Agricultural Engineers, 41, pp. 489–495.

CAPÍTULO 10

A UGV NAVIGATION SYSTEM FOR LARGE OUTDOOR ENVIRONMENTS INCLUDING VIRTUAL OBSTACLES FOR NO-GO ZONES

M. GARZON, E. FOTIADIS, P. PUENTE, A. ALACID y A. BARRIENTOS

Centro de Automática y Robótica UPM-CSIC, Calle José Gutiérrez Abascal, 2. 28006 Madrid, Spain – ma.garzon@upm.es.

This work presents a navigation system for UGVs in large outdoor environments; virtual obstacles are added to the system in order to avoid zones that may present risks to the UGV or the elements in its surroundings. The platform, software architecture and the modifications necessary to handle the virtual obstacles are explained in detail. Several tests have been performed and their results show that the system proposed is capable of performing safe navigation in complex environments.

1 Introduction

The autonomous navigation for Unmanned Ground Robots (UGV) is a very useful tool for many robotic applications; one of the areas where it can be highly exploited is the autonomous surveillance of large infrastructures, in which this work is framed. The autonomous navigation can be defined as the problem of finding and following a collision-free motion from a start to a goal position. This paper presents a scheme to perform the autonomous navigation, and to improve the safety of the system by adding a series of virtual obstacles in order to avoid zones where a potential risk, which may not be detected by the sensors on board the UGV, has been identified.

One of the main concerns regarding autonomous navigation is the safety of both the UGV performing the navigation and the elements that may be in the surroundings. In order to avoid collisions, most systems rely on the information provided by the sensors onboard the UGV – mainly on laser rangefinders – but when the environments are more complex those sensors does not provide all the information necessary, making it necessary to use more complex sensors and therefore increasing the complexity of the system and still not assuring the detection of all the obstacles. For the proposed surveillance application, most of the obstacles that cannot be detected by the laser rangefinders, such as: sidewalks, fences, benches, negative obstacles or small bushes; are placed in fixed locations, and therefore can be known *a priori*.

This work proposes a very simple solution to that problem, and it is to create virtual obstacles for those elements denoted as No-Go Zones (NGZ), and add them to the map created by the SLAM algorithm. This modified map is sent to the path planning algorithm, and will allow the navigation system to avoid such obstacles; also some modifications to the basic navigation architecture are made to prevent misbehaviors of any modules of the navigation system.

The outline of this paper is as follows: after this brief introduction, Section 2 reviews some related works. Section 3 describes the hardware components used. The software architecture for the navigation is described in Section 4. In Section 5, the Virtual Obstacles Approach and the changes to adapt the architecture to this approach are detailed. Section 6 describes the experiments that have been carried on and their respective results. Finally, Section 7 presents the conclusions and future lines of work.

2 Related Work

The Autonomous Navigation Robot world is very large, so in review of the related work, the content will be enclosed to the navigation of Unmanned Ground Vehicles (UGVs) dedicated to outdoor surveillance tasks. This is due to the huge amount of possibilities of surveillance, whether indoor or outdoor, that are carried through. For example, from using fixed cameras which help the UGVs in its task to Unmanned Aerial Vehicles (UAVs) that work cooperatively.

The main objective of any navigation system is to move safely from a start position to a goal position, both of them determined previously. That means avoiding obstacles and making the path in the lowest possible time. For example, in (Kunwar, et al., 2006), they develop a moving target interceptor in environments where obstacles may move or not. The navigation concept in here is determined by the position of the target, in which velocity and position are taken in order to intercept the target with an algorithm called rendezvous-guidance (RG). They just base their algorithm in a vision module. Other researchers have been including fuzzy logic in the navigation algorithm through the time. These method can be seen in (Zhu, et al., 2007), where they develop a fuzzy logic general navigation system which produces smooth trajectories. Here, the dead cycle problem is solved and static and moving obstacles are also avoided.

Quite a lot of sensors configuration can be arranged to perform autonomous navigation. Despite the most utilization of lasers combined with other sensors on these platforms, in another configuration is used. They develop a mobile robot whose navigation is based on GPS and Sonars. They introduce a radial basis function network (RBFN) used to locate the GPS into the robot coordinates, giving a smooth turning of the platform when avoiding obstacles. Other work bases its navigation just in vision module, as (Kunwar, et al., 2006) or (De Cubber, et al., 2009). This last one focuses its purpose on determining the terrain traversability with stereo images.

Returning to the most used method, the Laser Range Finders (LRF), in (Aliakbarpur, et al., 2009), they suggest a navigation system based on a 3D laser, a stereo camera and an Inertial Measurement Unit (IMU). In its work they develop an algorithm to calibrate these devices with the objective of being useful for surveillance and show its utility. Another LRF and camera based system is the one presented in (Kumar, et al., 2009). Here, besides the navigation system, human faces can also be detected, but is not optimal in outdoors. Other platform used in outdoor surveillance is (Yang, et al., 2010). In here, fast obstacle detection is developed based on two color cameras and helped with a 1D LRF. Regarding the navigation system, it is based on an IMU and a GPS. The innovation here is the method used to get the main ground disparity (MGD) from the V-disparity images in order to adapt the platform to intricate terrains.

3 Hardware Components Description

This section describes the hardware components used for this work. It includes a base platform, an on-board computer and series of on-board sensors that, when used together, allow the system to perform safe navigation.

3.1. Base Platform

The mobile platform is based on the Summit XL. It has skid-steering kinematics based on four high efficiency motors. The odometry is provided by an encoder on each wheel and a high precision angular sensor assembled inside the chassis. The robot can move autonomously or it can be teleoperated using images from a camera on-board. The Summit XL uses an open and modular control architecture based on ROS. An image of the robot before all the equipment was mounted is shown in *Fig. 7*

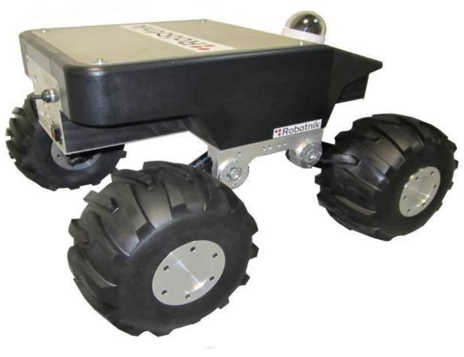


Fig. 7. Summit XL Mobile Platform

3.2. On-Board Components

On-Board Computer:

The robot has an embedded Linux PC. It is an Intel DN2800MT with mini ITX form factor located in the middle of the robot, The computer is completed with 4GB of RAM and a 2.5” SATA HDD. This computer allows having all the data processing and navigation algorithms in a fully autonomous manner.

Laser Rangefinders:

Two Hokuyo laser rangefinders are mounted on the platform: the first one is an UTM-30LX-EW, it can scan a 270º semicircular field, with a guaranteed range that goes from 0.1 to 30 meters and a maximum output frequency of 40Hz. The second laser rangefinder is an URG-04LX, it has a semicircular scanning area of 240º; the guaranteed range for this device goes from 0.06 to 4.09 meters and it can operate at a maximum frequency of 10 Hz.

Both devices are shown in *Fig. 8*. They are placed at different heights, with the 4 meters rangefinder at 10 centimeters from the floor in the front of the robot, and the 30 meters laser at 60 centimeters over the floor and in the center part of the robot.

UTM-30LX-EW URG-04LX

Fig. 8. Laser Rangefinders Mounted On-Board

Global Position System:

A Novatel OEM-4 GPS engine is used; it can offer centimeter level positioning accuracy with a frequency of 2Hz. RS232 serial communication is used to read the incoming data and send correction commands. The engine is complemented with an ANT-A72GOLA-TW GPS antenna.

Video Cameras.

There are also two cameras on-board the robot: the first one is a Logitech Sphere Camera, with motorized tracking (189° horizontal and 102° vertical) and Autofocus lens system. It provides a frame rate of up to 30 fps and a resolution of 1600 by 1200 pixels (HD quality); the second camera is a Firefly MV FMVU-03MTM, it has a frame rate of 60 fps and a resolution of 752 x 480 pixels.

The sphere camera is used mainly for teleoperation, due to its Pan-Tilt capability, while the Firefly camera is used for high level video processing because it is faster and has better optics.

Inertial Measurement Unit.

A MicroStrain 3DM-GX3 25 which is a high-performance, miniature Attitude Heading Reference System (AHRS) is mounted inside the robot. It combines accelerometers, gyroscopes and magnetometers in the three axes, with temperature sensors and an embedded processor to provide static and dynamic orientation, as well as inertial measurements.

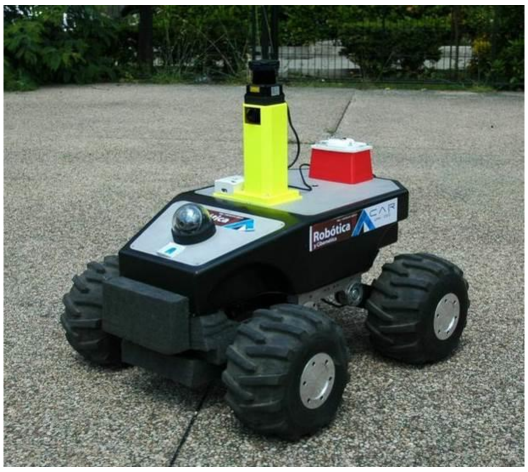


Fig. 9. Platform with all sensors and components.

The *Fig. 9* shows a picture of the complete system, the on-board computer and IMU are inside the chassis. The 30m rangefinder and one camera are mounted on a pedestal to have a better detection of the external objects, and the 4m laser rangefinder is below the robot and allows avoiding lower obstacles.

4 Software Architecture

This section describes the software architecture used for this work; it controls all the components of the UGV as well as the high level algorithms for mapping and navigation, including the proposed schema to avoid NGZ. The entirely software is supported by the software framework ROS (Robot Operating System), which comprises libraries and tools that facilitate the development of new robot applications.

First the basic components necessary to control de the robot will be mentioned. After that, the main algorithms used for the navigation will be described.

4.1 Basic Components

4.1.1 tf

To set the sensor locations and maintain the relationship between coordinate frames in a tree structure, *tf* (transform functions) package is executed. It allows keeping track of multiple coordinate frames and their corresponding transformations over time. A clear example is the sensor locations with respect to the robot body or the global position of the UGV in the reference frame. It is a very important component, because a simple deviation of a laser measurement could result in the robot hitting an obstacle.

4.1.1 summit_xl_ctrl

The *summit_xl_ctrl* package performs the low level control of the robot. This means that it receives the linear and angular velocity commands from either the teleoperation or the autonomous navigation modules. Those commands are processed using the inverse kinematics of the UGV in order to obtain the required speed for each wheel. The resulting control signals are sent directly to the control of each wheel-motor in order to obtain the desired speed. This package also receives the readings from the encoders on each wheel and from the angular sensor (gyroscope). These data are used to calculate the odometry which then is sent back to the other nodes in the system.

4.1.2 summit_xl_pad

The *summit_xl_pad* node is used to teleoperate the UGV. It uses data incoming from any joystick or control pad, and converts those data to velocity commands that are sent directly to the *summit_xl_ctrl* node, thus performing a direct control over the movements of the UGV.

4.1.3 EKF pose

The *EKF (Extended Kalman Filter) pose* package is used to estimate the three-dimensional pose (position and orientation) of the robot. Once the robot odometry, GPS and IMU readings are received and their corresponding transformations are set. It is possible to establish a measurement model that links the measurements of each sensor with the global position of the mobile robot. Those models are first combined using probability fusion techniques and then expressed as another probability density function. A detailed explanation of this process can be found in (Garzón, et al., 2013).

The models and the Kalman Filter itself are implemented using a ROS wrapper of the Bayesian Filter Library. The combined pose is sent back to the system as a ROS topic and also by setting the corresponding transformations in the *tf* topic. The estimated pose can be obtained with the data available at each that time, even if one sensor stops sending information. Also, if the information is received after a time-out, it is disregarded.

4.2 Main Navigation Algorithms

The main navigation algorithms are a series of modules that, when combined, provide the UGV with the ability of performing autonomous navigation. Here each one of the modules will be briefly described: the first module is used for creating maps using readings from the laser and the odometer (SLAM), another module is used for handling and publishing the maps, a third one is in charge of the localization within a given map, and finally, the main component performs path planning and following.

4.2.1 SLAM (gmapping)

The navigation scheme proposed is based on navigating on a known map, thus building this map is a main part of the overall process. In this work a ROS wrapper for *GMapping* is used. *GMapping* takes raw laser range data, in this case the combined odometry from the EKF, and uses it to build grid maps using a highly efficient Rao-Blackwellized particle filer (Grisetti, et al., 2007).

4.2.2 Map Server

The map server provides the map data to all other elements on the system. The maps are based on grayscale images, where each pixel represents the occupancy state of each cell of the world; white pixels represent free space, black ones represent occupied cells and gray ones represent unknown spaces. The values are loaded from the images and published in one or more occupancy grids.

4.2.3 Localization (AMCL)

This package implements the Adaptive Monte-Carlo Localization (AMCL). It works as a probabilistic localization system, using a particle filter to track the pose of the UGV in a known map. It relays on the information provided by the scans of laser rangefinder and the *tf* messages. The AMCL publishes the estimated pose of the robot in the map with its corresponding covariance, and the set of pose estimates being maintained by the filter.

4.2.4 Path Planning and Following (Move Base)

This package includes some major components for the two-dimensional navigation. It takes a goal in the world, and attempts to reach it with a mobile base. The planning is separated in two phases: global and local; each one of them uses its own cost map.

The global planner finds a minimum cost plan, from a start to an end point, using the Dijkstra's algorithm in a given global cost map. It works under the assumption that the UGV is a circular robot and provides only the list of points necessary to reach the goal, it does not provides orientations, times nor velocity commands.

The local planner provides a controller that helps the robot to follow the global path created by the global planner. It uses a smaller local occupancy map that is constantly redefined around the current pose of the robot. The local planner creates a kinematic trajectory for the robot to get from a start to a local goal pose, providing the linear and angular velocities that will allow the platform to safely traverse the local occupancy grid. The local goal pose is selected as the farthest point of the global path that is covered by the local map. It takes into account static and dynamic obstacles that may be added by the sensors on-board the robot. The velocity commands are sent directly to the robot's controller.

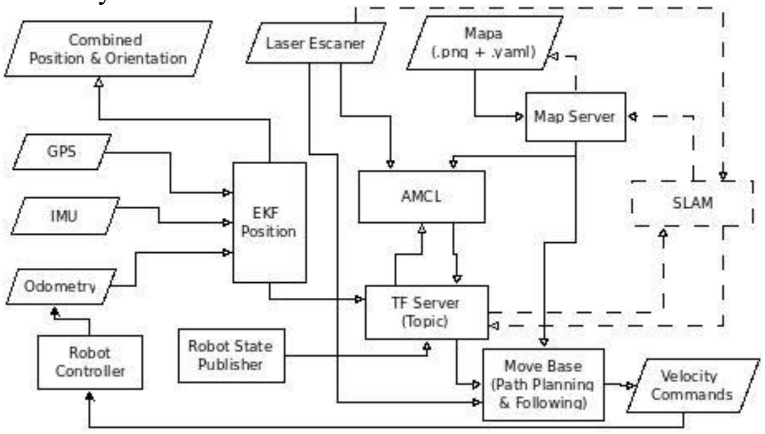


Fig. 10. Software Architecture.

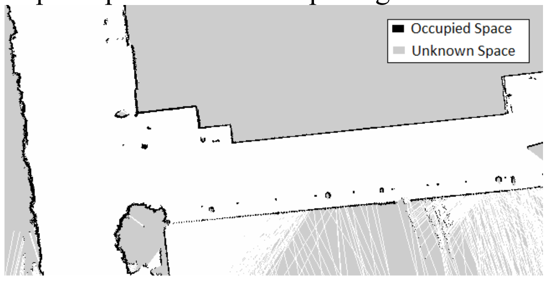
The Fig. 10 shows the complete software architecture, all important modules, inputs and outputs, as well as their respective connections are shown; the dashed lines represent the modules and connections used only for the SLAM algorithm.

5 Virtual Obstacles Approach

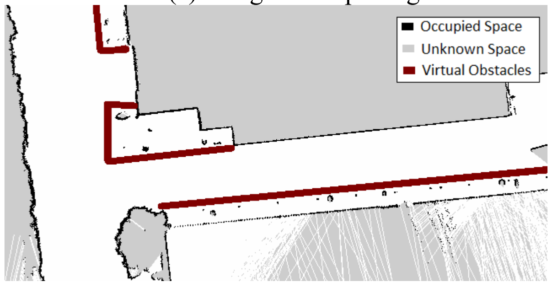
This section describes the changes and additions made to the software architecture described in Section 4, to enhance it with the ability to avoid NGZ. The basic idea is to modify the occupancy map, in order to mark the forbidden zones as if they had obstacles in them. Making changes in the map has two main drawbacks the performance of the system that need to be solved: first the localization system will not work correctly since it will no longer be able to match correctly the readings of the laser rangefinder against the given map; the second consequence is that since the laser rangefinder will not find any obstacles, it will clear the NGZ even if they were previously marked as occupied. In the next subsections, the methodology to add the obstacles will be explained, also the solutions the localization and clearing zones problem will be detailed.

5.1 Changing the Map to Add No-Go Zones

As was described in Section 4.2.2, the map is represented as an image where each pixel represents a cell and its occupancy is given by the color of the pixel. Since the location and shape of the NGZ is assumed to be known *a priori*, virtual obstacles, which represent those zones, can be added *offline* as occupied spaces into the map image.



(a) Original Map Image



(b) Inclusion of Virtual Obstacles

Fig. 11 Virtual Obstacle Addition.

The Fig. 11 shows a portion of the map before and after the virtual obstacles were added. In this example, the virtual obstacles are used to represent zones where there is a sidewalk with a height that cannot be detected by the laser rangefinder.

5.2 Solving the Issues of the Modified Map

As was mentioned before, one of the main issues of modifying the occupancy grid is that the laser scanner will no longer match the information contained in the map. This is the most problematic issue, because it affects the localization algorithms which are basic for the navigation. To solve this problem two separated map server instances were created: the first instance uses the map without any modification, thus allowing the localization algorithm to have a correct representation of the world around the UGV. The second instance uses the map with the virtual obstacles and is sent to the path planning and following module. This module reads the map and performs the planning avoiding the NGZ. In the Fig. 4 it can be seen that there is only one instance of the map server, that loads the map and sends it to both the localization and the path planning and following modules. This will be replaced by two separated modules, the first including the virtual obstacles and only communicated with the move base module; a second one without modifications that will be connected only to the AMCL module.

The second issue is that the laser scanner may clear the location where the cost map has been modified. This may cause that the planning algorithm to generate paths that may traverse the NGZ, to solve this problem a service provided by the cost map handler is used, this service reloads the original cost map from the stored image, so before the global planning is performed, the cost map is reloaded from the image. This gives as result on a path that will always avoid the NGZ.

6 Experiments and Results

This section describes the scenarios and experiments that were carried on in order to evaluate the performance of the proposed technique. The Hokuyo UTM-30-LX-EWTM 30m laser rangefinder described in Section 3.2 was used for robot localization and obstacle avoiding. It is placed in a height of about 60cm from ground level. It is evident that in real world environments, many potential obstacles lie below this limit.

The experiments where held in two different settings. The first one took place in the campus gymnasium, where normally there are no obstacles below the laser height. This relatively safe zone allows the system to perform qualitative comparisons between using or not NGZ. First the map is constructed using the SLAM technique explained in Section 4.2. In this step the robot is being moved manually around the gymnasium. Next, the virtual obstacles for NGZs are superimposed on the map such as if they were real objects. As was explained in Section 5, placing virtual obstacles in a map renders it inappropriate for navigation, since these obstacles cannot actually be detected by the robot's laser sensor. To address this, two separated instances of the map were created: one used for navigation and another for the obstacle avoidance.

In order to have real NGZs in a controlled way, training mats were placed on the floor of the gymnasium. This facilitates the visual assessment of the methodology proposed. Two different tests were made in this scenario: the first one was done using the standard configuration of the navigation system; for the second one, the virtual obstacles for NGZs proposed were included.

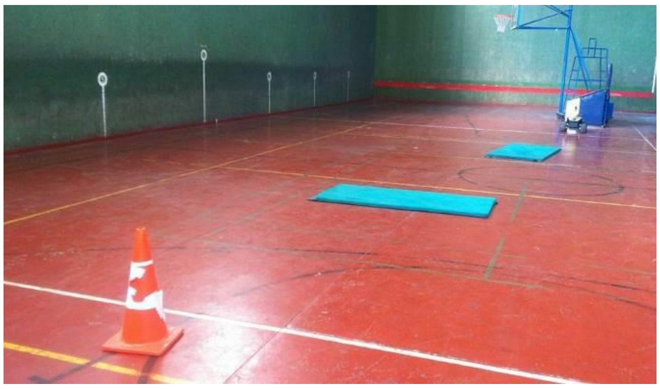
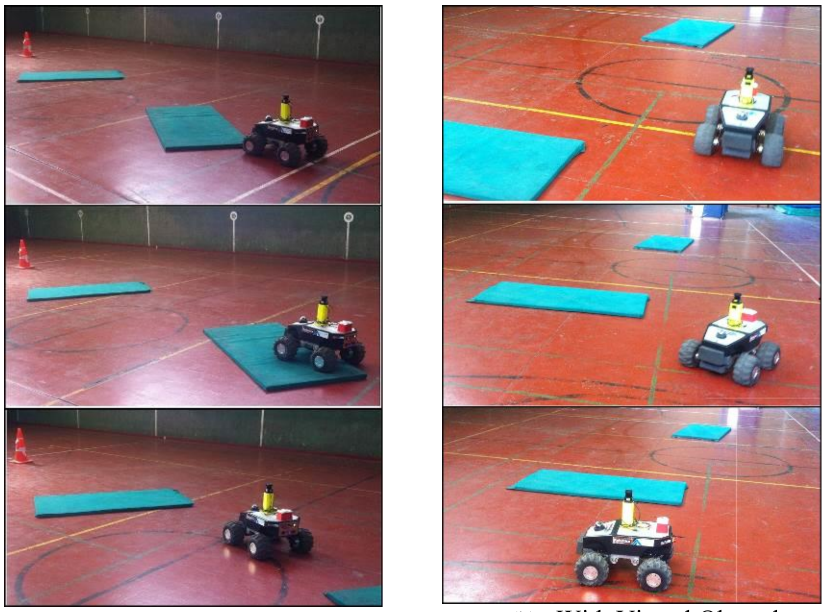


Fig. 12. Gymnasium Test Scenario.

Fig. 12 shows this test scenario. The UGV is in its starting position and the goal is represented by the orange cone, also the green mats on the floor show the relative placement of NGZs. Using this setup, the performance of the navigation system, regarding its ability to handle the presence or absence of No Go Zones, was compared. During the first test, when only the navigation map was used, the robot was unable to avoid the obstacles because the laser rangefinder could not detect them. On the other hand, when a second layer containing the virtual obstacles was used, the path performed by the UGV circumvented the NGZs by passing around them. The Fig. 13 shows images of the platform while executing the two tests.

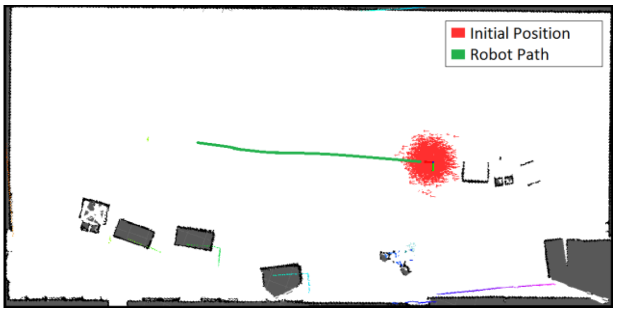


(a) Without Virtual Obstacles (b) With Virtual Obstacles

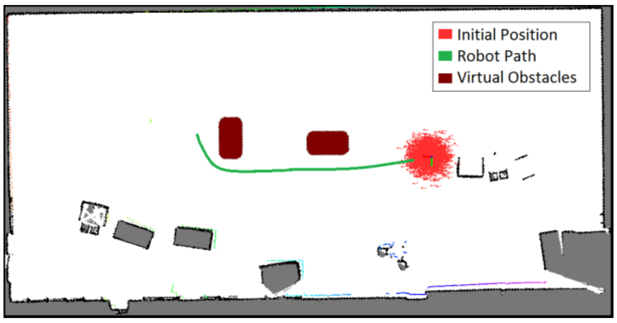
Fig. 13. Test with (b) and without (a) Virtual Obstacles for NGZ

It can be seen in Fig. 13(a) that the robot, in the first test, passed over the mats without detecting them, while in the second test, shown in Fig. 13(b), the robot makes a route around the mats and successfully avoiding the obstacles.

The result of the two first tests is more evident by comparing the path when NGZs where ignored and the path when they were taken in account. As it can be seen in Fig. 14(a), when the NGZ are ignored the path planning creates an almost straight line from the starting position to the goal. On the other hand, when the virtual obstacles are present, the path is deviated in order to avoid the mats as observed in Fig. 14(b).



(a) Ignoring NGZ



(b) Using Virtual Obstacles for NGZ

Fig. 14. Test paths with (b) and without (a) NGZ.

The second experiment was held outdoors in the university campus. For this setting, the map used was constructed in a previous experiment. In this more complex environment, a lot of objects are found below the laser height, examples of some of those elements are: sidewalks, pavements, benches, flower beds, or dustbins. Furthermore, there are other elements, such as fences, nets or sparse plants, which regardless of their height, cannot be detected correctly by the sensor. All these obstacles, which may present high risk for the robot, were marked as No-Go Zones. Therefore, corresponding virtual obstacles were manually added to the existing obstacle map. According with the process described, the original map was used for localization and the one with the modifications for the path planning and following. In order to avoid potential damages for the equipment, only the test with the NGZ map was executed.

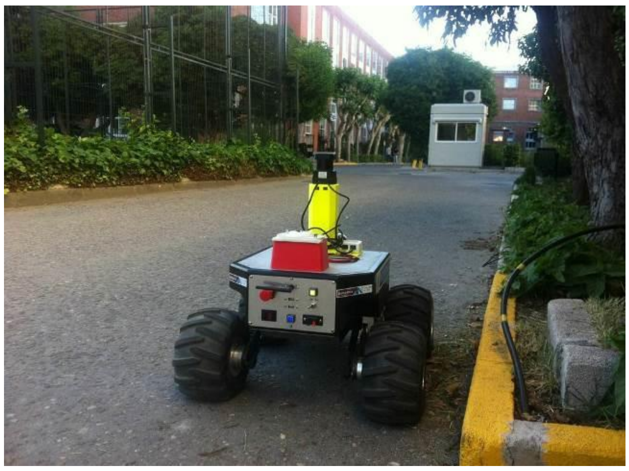


Fig. 15. Outdoor Scenario.

The scenario where the second experiment was held can be observed in Fig. 15. There is a pavement and some plants that the laser cannot detect, and a fence on the other side of the road. If the original map were used, then collision would be very probable. By using the virtual obstacles in the NGZ, the robot can effectively avoid these collisions and it drives safely towards the desired goal position.

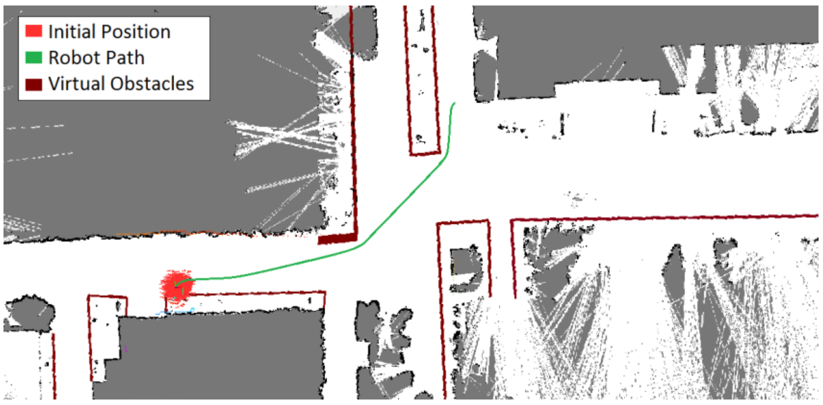


Fig. 16. Outdoor Test Using NGZ

The path that the UGV follow during this experiment is shown in Fig. 16.**Fig. 16. Outdoor Test Using NGZ** The original map and the virtual obstacles are also depicted. It is clear how the path is deformed in order to avoid the virtual obstacles as well as the normal ones, thus proving the ability of the proposed scheme to handle this complex scenario.

7 Conclusions

A system capable of navigating within simple and more complex real world environments was presented. Safe navigation in complex environments is assured by using a simple technique that allows the UGV to avoid zones where there may be risk of collisions or damage to the robot, consisting on adding virtual obstacles to the map created by the SLAM algorithm.

The main concerns of modifying a given SLAM map are studied and solved by modifying the navigation architecture, creating two instances of the map server and assuring that the virtual obstacles are taken into account in the path planning process.

Several tests were performed, first in indoor controlled scenarios and then in large and complex outdoor scenarios, all the tests show that the inclusion of the virtual obstacles enhances the navigation system, without having negative influences on the behavior of other modules of the system.

Future lines of work include the automatic inclusion of the virtual obstacles, by using an external sensor on a fixed location or onboard an mini UAV hovering over the UGV, and studying techniques to modify directly the cost map without using two instances of the map server.

Acknowledgements

This work was supported by the Robotics and Cybernetics Group at *Universidad Politécnica de Madrid* (Spain), and funded under the projects: ROTOS - Multi-robot system for outdoor infrastructures protection, sponsored by Spanish Ministry of Education and Science (DPI2010-17998), and ROBOCITY 2030, sponsored by the Community of Madrid (S-0505/DPI/000235).

References

Aliakbarpour, H., Núñez, P., Prado, J., Khoshhal, K., & Dias, J. 2009. An Efficient Algorithm for Extrinsic Calibration between a 3D Laser Range Finder and a Stereo Camera for Surveillance. *Advanced Robotics, 2009. ICAR 2009. International Conference on*, pp. 1 - 6.

De Cubber, G., Doroftei, D., Nalpantidis, L., Sirakoulis, G. C., & Gasteratos, A. 2009. Stereo-based Terrain Traversability Analysis for Robot Navigation. *IARP/EURON Workshop on Robotics for Risky Interventions and Environmental Surveillance.*

Garzón, M., Valente, J., Zapata, D., & Barrientos, A. 2013. An Aerial-Ground Robotic System for Navigation and Obstacle Mapping in Large Outdoor Areas. *Sensors, 13*(1): 1247-1267.

Kumar, S., & Awasthi, P. 2009. Navigation Architecture for Autonomous Surveillance Rover. *International Journal of Computer Theory and Engineering*, 1(3): 231 - 235.

Kunwar, F., & Benhabib, B. 2006. Rendezvous-Guidance Trajectory Planning for Robotic Dynamic Obstacle Avoidance and Interception.

Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 36(6): 1432 - 1441.

Ray, A. K., Behera, L., & Jamshidi, M. 2009. GPS and Sonar Based Area Mapping and Navigation by Mobile Robots. *Industrial Informatics, 2009. INDIN 2009. 7th IEEE International Conference on*, pp. 801 - 806.

Yang, C., Jun-Jian, P., Jing, S., Lin-Lin, Z., & Yan-Dong, T. 2010. V-disparity Based UGV Obstacle Detection in Rough Outdoor Terrain. *A Acta Automatica Sinica*, 36(5): 667 -673.

Zhu, A., & Yang, S. X. (2007). Neurofuzzy-Based Approach to Mobile Robot Navigation in Unknown Environments. *Systems, Man, and Cybernetics, Part C: Applications and Reviews,* 37(4): 610 - 621.

CAPÍTULO 11

SISTEMA DE CAPTURA DE MOVIMIENTO CON SENSOR KINECT Y ACELERÓMETROS PARA REHABILITACIÓN FUNCIONAL Y COGNITIVA DE CÁNCER DE MAMA

J. R. GARCÍA[1], R. BAREA[1], E. LÓPEZ[1], V. PRIETO[2], M.J. YUSTE[2], P. DE LA VILLA[3], L.M. BERGASA[1] y M. OCAÑA[1]

[1] Departamento de Electrónica, Universidad de Alcalá.
[2] Grupo de Investigación en Fisioterapia en los Procesos de Salud de la Mujer. Departamento de Fisioterapia. Universidad de Alcalá.
[3] Departamento de Biología de Sistemas. Universidad de Alcalá.

Email: {barea, bergasa, elena, mocana}@depeca.uah.es

El objetivo de este trabajo es el desarrollo de un sistema de monitorización inteligente de rehabilitación cognitiva y funcional. Se pretende desarrollar un sistema de valoración analítica y funcional de las extremidades superiores basado en el sensor Kinect y acelerómetros con el objetivo de aplicarlo en el proceso de rehabilitación funcional y cognitiva en mujeres tratadas con cáncer de mama. Posteriormente se pretende desarrollar un sistema robotizado que permita ayudar a estas personas en su proceso de rehabilitación.

1 Introducción

La Ingeniería de Rehabilitación (IR) es la aplicación de la ciencia y de la tecnología, con el objetivo de restituir, minimizar y/o compensar las dificultades enfrentadas por individuos con discapacidades físicas, sensoriales y/o psíquicas, al efectuar las actividades de la vida diaria. En los últimos años son numerosas las aplicaciones robóticas que han surgido dentro de esta línea de trabajo, entre ellas caben citar los robots Lokomat, Roboterapist 3D, ASIBOT, ARMIN, MIT-Manus y HAL (Lokomat 2013, Roboterapist 3D 2013, Asibot 2013, Armin 2013, Krebs et al., 2007, HAL 2013).

En medicina, la rehabilitación se refiere al proceso de atención sanitaria con miras a eliminar o reducir en lo posible las disfunciones y discapacidades de los humanos, secuelas de enfermedades y traumas, para conseguir la recuperación física, psíquica, social y laboral del paciente. Para lograr sus objetivos, la rehabilitación se vale de disciplinas tan diferentes como fisioterapia, logopedia, la terapia ocupacional y la psicología.

La valoración funcional global es el conjunto de evaluaciones que conforman el examen clínico. Estas valoraciones pueden ser analíticas y/o funcionales y pueden ser de carácter cualitativo y/o cuantitativo. Para llevarlas a cabo se recurre a medios visuales (observación de las estructuras desde un punto de vista morfológico y funcional) manuales (palpación, movilización tisular, etc.) e instrumentales (medición de magnitudes físicas y sus variaciones, de la forma más objetiva y precisa posible mediante equipos de medida). Las valoraciones funcionales estudian las posibilidades de independencia del paciente tanto en la vida privada como en la profesional. En éstas se valora al paciente de forma global, cualitativa y cuantitativamente, se busca la racionalidad del gesto, motivar del paciente, etc. Éstas son más complejas que las valoraciones analíticas y a la vez más subjetivas ya que estudian las interrelaciones existentes entre las distintas estructuras y sistemas que posibilitan la independencia en la vida privada y profesional del paciente. En general, en las valoraciones funcionales se estudia la facilidad que tienen los pacientes para realizar las diversas actividades de la vida diaria (acciones cotidianas, movimientos extremidades, marcha, etc.).

El seguimiento del Movimiento humano para rehabilitación ha sido un tema de investigación activo desde los 1980's. Ya que la rehabilitación es un proceso dinámico que utiliza las instalaciones disponibles para corregir cualquier comportamiento del movimiento no deseado con el fin de alcanzar una expectativa (por ejemplo, posición ideal) es necesario que las actividades de un paciente sean continuamente monitorizadas y posteriormente corregidas (Zhou & Hu, 2008).

Los sistemas de seguimiento de movimiento del cuerpo humano deben generar datos en tiempo real que representen de forma dinámica los cambios de pose del cuerpo humano (o una parte del mismo). Estos sistemas pueden estar basados en visión (mediante marcadores visuales o libres de marcadores) o no basados en visión (sensores de presión, goniómetros, conmutadores, magnetómetros y sensores inerciales -acelerómetros y giróscopos-). Entre estos sistemas caben citar el APAS (Apas, 2013), VICON (Vicon, 2013), BTS kinematic (BTS, 2013), NedHombro/IBV (NedHombro, 2013), etc.

En los últimos tiempos está surgiendo con fuerza la utilización del sensor Kinect en aplicaciones de Rehabilitación. Entre estos trabajos caben citar el VirtualRehab (Vrehab, 2013) es un producto para la rehabilitación de pacientes con algún grado de discapacidad física, que combina la captura del movimiento con tecnología de videojuegos. El resultado final es un producto que ayuda a mejorar la calidad de vida de estos pacientes a la vez que se divierten haciendo los ejercicios prescritos específicamente por sus fisioterapeutas. El SeeMe (SeeMe, 2013) que es un sistema de realidad virtual de captura de video proyectado donde el usuario es incrustado en una historia virtual utilizando algoritmos para de movimiento y reconocimiento de posición y de análisis. La aplicación RMT (Reflexion Measurement Tool) (RMT, 2013) permite a los profesionales médicos personalizar los planes y programas y potencialmente monitorear de forma remota a los pacientes para garantizar que sus ejercicios vayan por buen camino.

El objetivo de este trabajo es el desarrollo de un sistema de monitorización inteligente de rehabilitación cognitiva y funcional. La rehabilitación cognitiva y funcional tiene como objetivo la recuperación de aquellas personas que presenten una disminución de su capacidad física, sensorial o psíquica. Este proceso se inicia con la detección y diagnóstico de la problemática a tratar y continúa hasta la consecución y mantenimiento de la máxima funcionalidad personal posible.

En concreto en este trabajo se pretende desarrollar un sistema de valoración funcional de extremidades superiores de muy bajo coste basado en el sensor Kinect. Este trabajo se realiza en colaboración entre el Departamento de Electrónica y el Departamento de Fisioterapia de la Universidad de Alcalá y se centra en estudiar la valoración funcional de extremidades superiores de mujeres tratadas de cáncer de mama aunque puede ser extrapolable a otras valoraciones de movimientos de extremidades. Posteriormente se pretende desarrollar un sistema robotizado que permita ayudar a estas personas en su proceso de rehabilitación.

2 Motivación

El cáncer de mama constituye uno de los principales problemas de salud en los países desarrollados, encabezando la lista de tumores que padecen las mujeres con una representatividad del 20-30% (IARC, 2008). Gracias al diagnóstico precoz y al aumento de la eficacia de los tratamientos suministrados existe una tasa de supervivencia a los 5 años cercana al 80 %. Sin embargo, este tipo de terapias inducen una serie de complicaciones que deben ser tomadas en cuenta (Lacey et al., 2002): la disfunción del hombro

(Cheville & Tchou, 2007) (Kuehn et al., 2000), la alteración de la sensibilidad del miembro superior (Cheville & Tchou, 2007) (Kuehn et al., 2000), el linfedema (Bani et al., 2007) y el dolor crónico (Cheville & Tchou, 2007) (Kuehn et al., 2000). Éstas pueden provocar una serie de secuelas psíquicas de importancia que, junto con las anteriores, favorecen la disminución de la calidad de vida.

Teniendo esto presente, en el Grupo de Investigación en Fisioterapia en los Procesos de Salud de la Mujer del Departamento de Fisioterapia de la Universidad de Alcalá se están desarrollando investigaciones para comprobar si realmente existen alteraciones en los patrones musculares implicados en la articulación del hombro en mujeres tratadas de cáncer de mama. De confirmarse dicha disfunción, se establecerán investigaciones futuras para desarrollar un protocolo de fisioterapia eficaz para tratar este tipo de secuelas.

El objetivo final es estudiar si existe algún tipo de alteración en el patrón muscular (comienzo de la activación motora, número de unidades motoras reclutadas, correlación entre la fuerza muscular requerida y el reclutamiento motor en contracciones isométricas, pérdida de fuerza muscular, etc) de los movimientos de flexión, de abducción y de rotación externa del hombro y para ello será necesario desarrollar un sistema que permita la adquisición de EMG de los músculos implicados así como la amplitud articular y el registro de la trayectoria de los movimientos de la extremidad superior durante el tiempo de evaluación.

3 Arquitectura general del Sistema

Se ha desarrollado un sistema que realiza de forma protocolizada la adquisición de datos biomédicos, principalmente biomecánicos, y realiza el tratamiento de los mismos para generar informes. Estos informes a su vez sirven de base para posteriores estudios cuyo propósito es determinar la correlación existente entre las capacidades musculares y articulares de una persona afectada por una patología en contraposición con una persona sana.

El sistema consta de las siguientes partes diferenciadas:

- Un ordenador personal.
- Sensores:
    - Cámara de visión espacial *Kinect*® de Microsoft.
    - Acelerómetros *PhidgetAccelerometer 3-Axis* de Phidget.

- Dinamómetro *microFET2 Handheld Dynamometer* de Hoggan Health Industries.
- Electromiógrafo *EMG M408 Dual Bio Amp/Stimulator*® y el procesador digital *Chart5 for Windows* ®.

▪ Aplicación Informática de adquisición de procesamiento de datos.

La Fig. 1 muestra el diagrama de bloques del sistema implementado.

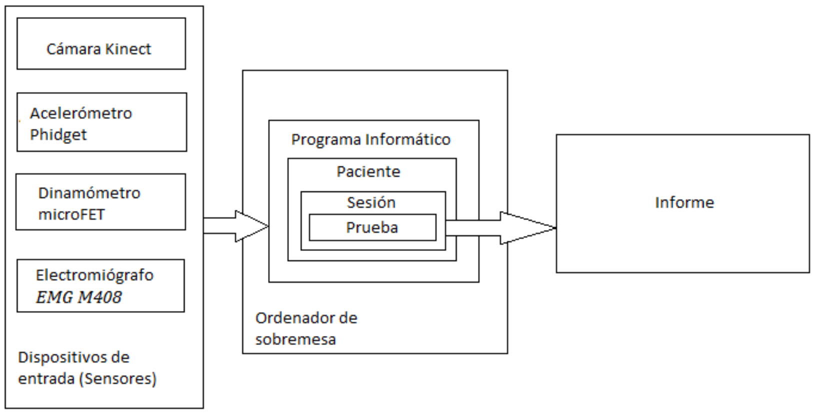


Fig. 1. Diagrama de bloques del sistema implementado.

El núcleo de la aplicación es el sistema de **captura de movimientos** mediante la cual se realiza el cálculo de la amplitud articular y el registro de la trayectoria de los movimientos de la extremidad superior durante la evaluación. La Fig. 2 muestra dicho sistema.

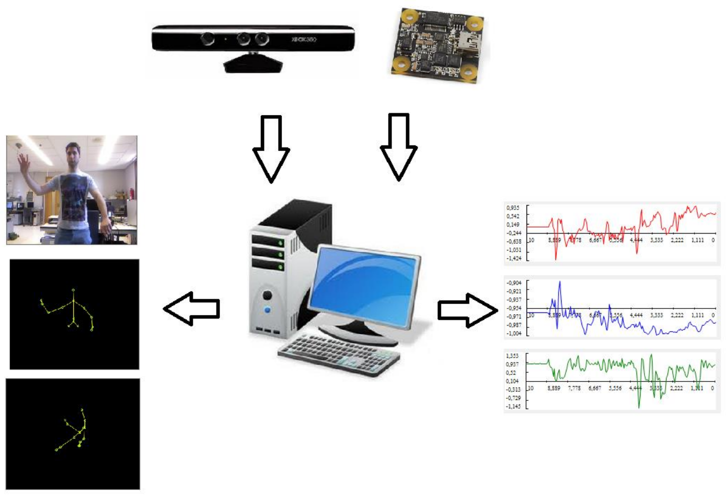


Fig. 2. Sistema de captura de movimientos.

4 Sistema de valoración funcional de extremidad superior basado en el sensor Kinect y acelerómetros

Como se ha visto el sistema se basa en acelerómetros 3D y el sensor Kinect. Los acelerómetros utilizados son PhidgetAccelerometer 3-Axis (Fig. 3) que permiten medir aceleraciones dinámicas (vibraciones) y estáticas (gravedad o inclinación). Los movimientos que detecta el acelerómetro son los de *pitch* y *roll*, y no permite medir el *yaw*. Por este motivo en función de la aplicación desarrollada es necesario colocarlos en diferentes posiciones para medir las amplitudes articulares.

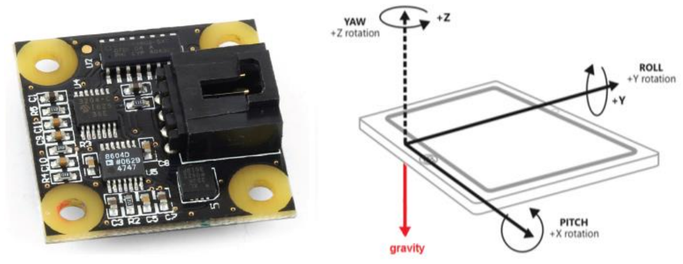


Fig. 3. PhidgetAccelerometer 3-Axis

La Kinect se compone principalmente de una cámara RGB dedicada al reconocimiento facial y la captura de video. También cuenta con un sensor de profundidad, un proyector infrarrojo con cámara VGA CMOS monocroma, un motor que gira la base y los micrófonos para reconocimiento de voz. Permite el rastreo de usuarios y posee visión de profundidad, por lo que es posible realizar un seguimiento temporal de la trayectoria tridimensional de las articulaciones de un sujeto (Fig. 4).

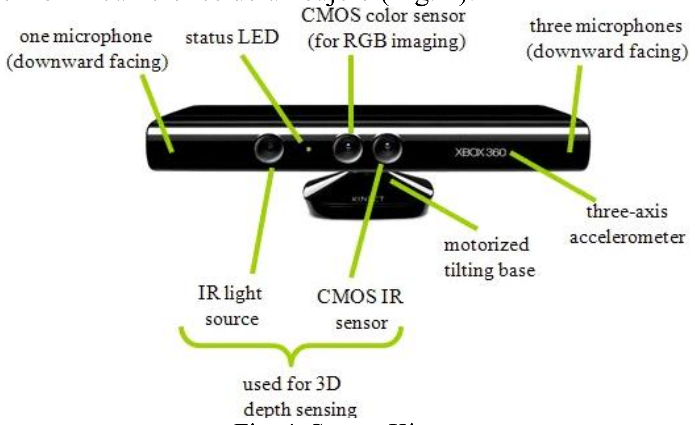


Fig. 4. Sensor Kinect.

La Kinect funciona proporcionando tres elementos de datos importantes (Ver Fig. 5):

- Color Stream: Flujo de datos de imágenes a color.
- Depth Stream: Flujo de datos de imágenes de profundidad.
- Skeleton Stream: Flujo de datos del rastreo de usuarios y esqueletos.

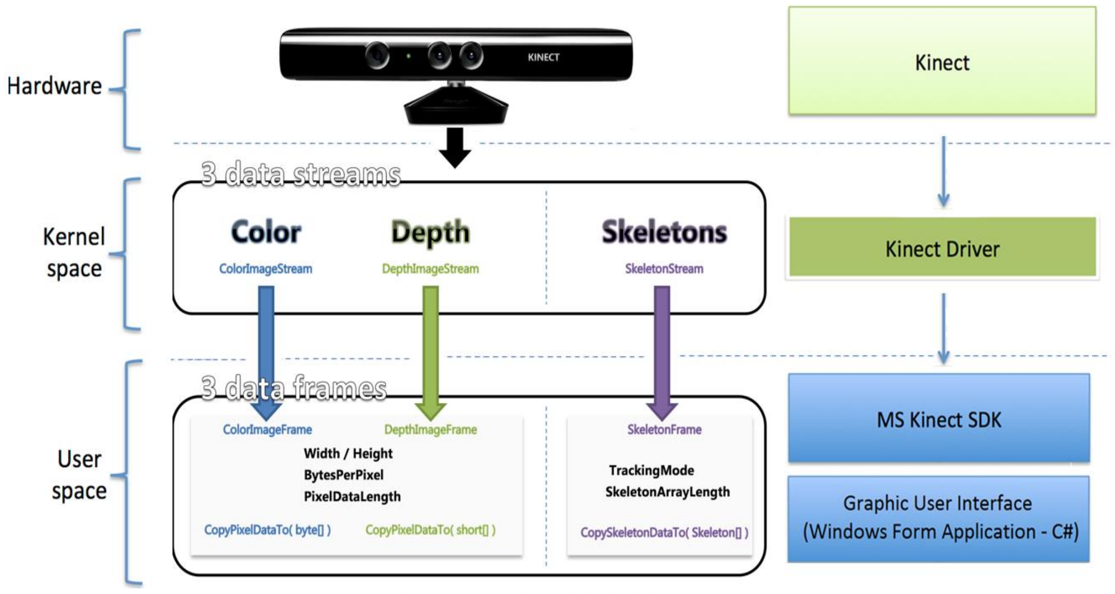


Fig. 5. Funcionalidad del SDK.

El SDK se encarga de proporcionar los drivers para gestionar estos flujos y convertirlos en imágenes. Proporciona también las librerías de usuario que permiten manipular estas imágenes en lenguaje de programación.

El corazón del funcionamiento del SDK de la Kinect son los eventos. Si se configura correspondientemente, cada vez que el SDK recibe de los flujos de datos de la Kinect suficiente información como para formar un frame (generalmente una imagen), produce una interrupción que alerta a la aplicación de que dicho frame está listo para tratarse. Estos eventos se producen a una velocidad de 30 frames/segundo, por lo que permite una alta interactividad. Principalmente, existen tres tipos de eventos de este tipo, de acuerdo a los tres tipos de flujos de datos (Fig. 6).

- **Color Frame Ready:** Es el evento encargado de la imagen capturada por la cámara RGB en el espectro visible.
- **Depth Frame Ready:** Es el evento encargado de la imagen de profundidad compuesta por el rebote de las constelaciones del proyector laser capturado por la cámara VGA en los infrarrojos.

- **Skeletal Frame Ready:** Es el evento que ofrece la matriz de esqueletos correspondientes a los usuarios detectados, que poseen a su vez la matriz de puntos de la localización de sus articulaciones.

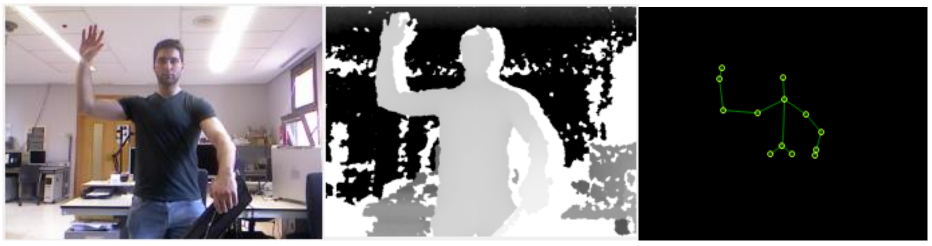


Fig. 6. Imágenes de los flujos de datos.

<u>Skeletal Frame Ready</u>

Cada esqueleto se compone de una matriz de 20 "*JointPoints*", una estructura que identifica por separado cada articulación y que guarda información referente a la posición espacial de dicha articulación así como el modo en que ha sido capturada. La Fig. 7 muestra la estructura del Skeleton Frame y la Fig. 8 las articulaciones detectadas.

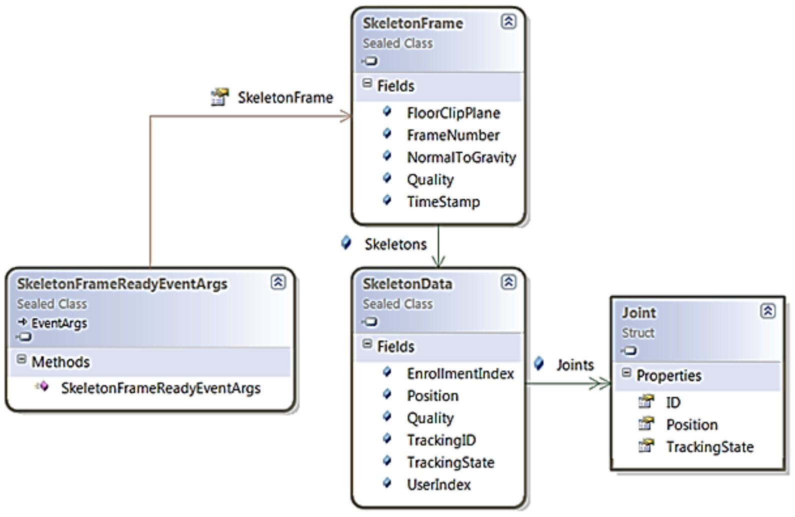


Fig. 7. Información del Skeleton Frame

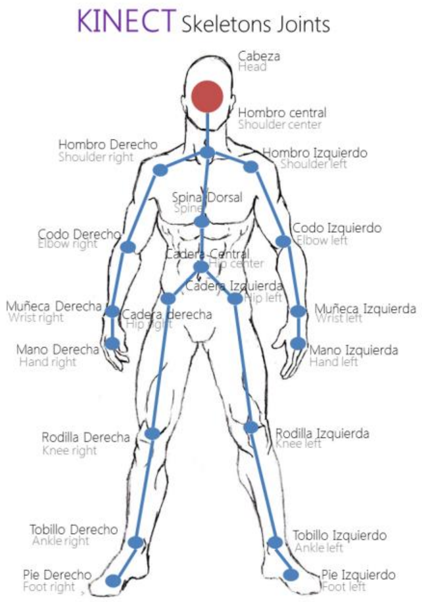
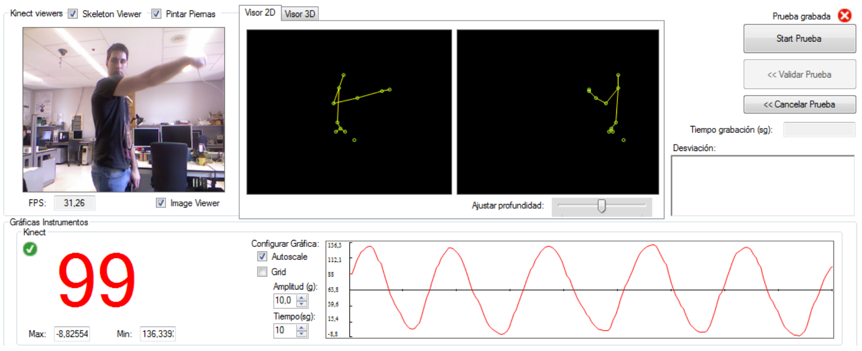


Fig. 8. Articulaciones detectadas.

4.1 Aplicación

La aplicación consta de una base de datos jerarquizada que permite configurar y almacenar datos sobre los pacientes, las sesiones y las pruebas/tests llevadas a cabo con cada uno de ellos. Permite un alto grado de configurabilidad en función de las pruebas a realizar. Además permite la generación de informes de los pacientes y evaluar su evolución a lo largo del tiempo. La Fig. 9 muestra el interfaz de prueba/test.


Kinect viewers [✓ Skeleton Viewer] [✓ Pintar Piernas]

FPS: 31,26 [✓ Image Viewer]

Gráficas Instrumentos

Kinect

99

Max: -8,82554 Min: 136,339

Configurar Gráfica: [✓ Autoscale] Grid

Amplitud (g): 10,0

Tiempo(sg): 10

Fig. 9. Interfaz de pruebas.

4.2 Tipos de Movimientos

Los movimientos articulares que se pretenden registrar y evaluar, son:

a) Flexión-Extensión: Los movimientos de flexión-extensión se producen en el plano sagital (Fig. 10).

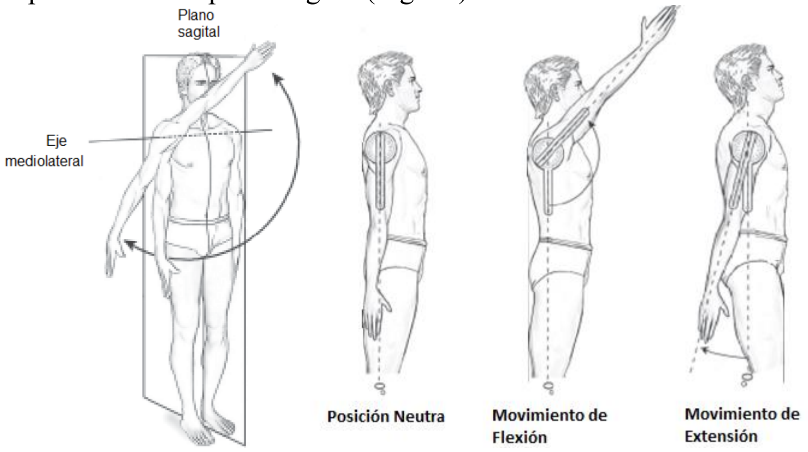


Fig. 10. Movimientos de Flexión-Extensión.

El sistema de referencia utilizado es el estándar de la goniometría, en el que de la posición neutra es aquella indicada por el brazo en reposo (posición 0).

b) Abducción-Aducción: Los movimientos de Abducción-aducción se producen en el plano coronal. Se utiliza el sistema de referencia propuesto por la goniometría, que indica que el ángulo 0, corresponde con la poción neutra, en la que el brazo se encuentra en posición de reposo (Fig. 11).

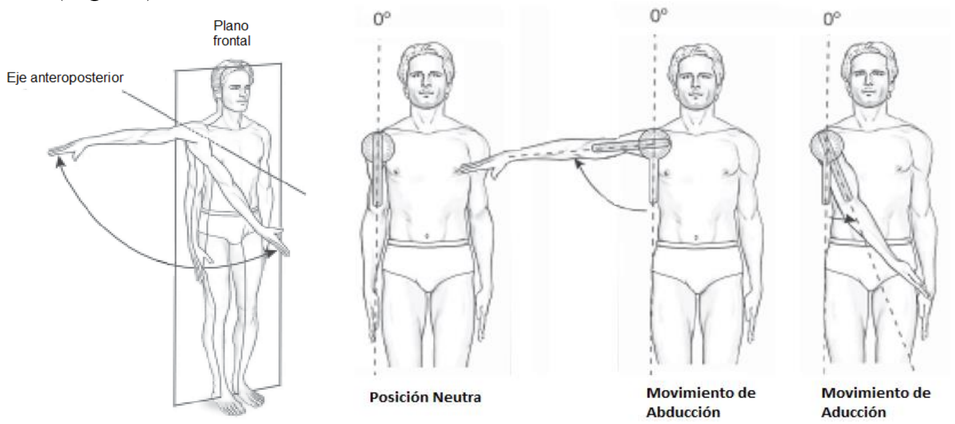


Fig. 11. Movimientos de Abducción-Aducción

c) Rotación interna-externa:

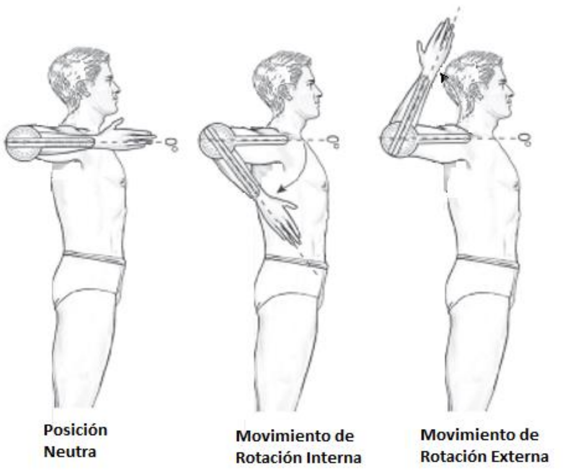
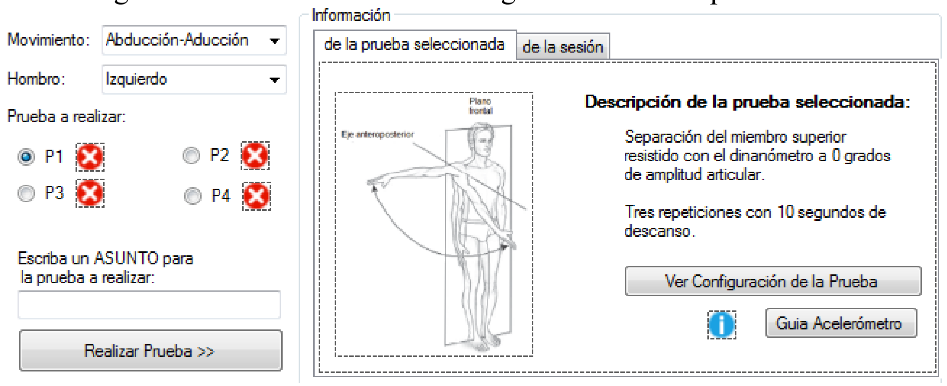

Posición Neutra
Movimiento de Rotación Interna
Movimiento de Rotación Externa

Fig. 12. Movimientos de Rotación Interna-Externa.

4.3 Pruebas/Tests

Con el objetivo de estudiar si existe algún tipo de alteración en el patrón muscular en los movimientos de flexión-extensión, de abducción-aducción y de rotación externa-interna del hombro se van a realizar una serie de 4 tests sobre el hombro afectado de manera que por cada sesión, se hayan extraído todos los datos indicados. De esta forma se requerirán a las mujeres tanto contracciones concéntricas como isométricas en diferentes grados de amplitud para obtener las cualidades electromiográficas del patrón muscular.

La Fig. 13 muestra el interfaz de configuración de estas pruebas.



Movimiento: Abducción-Aducción

Hombro: Izquierdo

Prueba a realizar:

[✓ P1] P2 P3 P4

Escriba un ASUNTO para la prueba a realizar:

Información

Descripción de la prueba seleccionada:

Separación del miembro superior resistido con el dinanómetro a 0 grados de amplitud articular.

Tres repeticiones con 10 segundos de descanso.

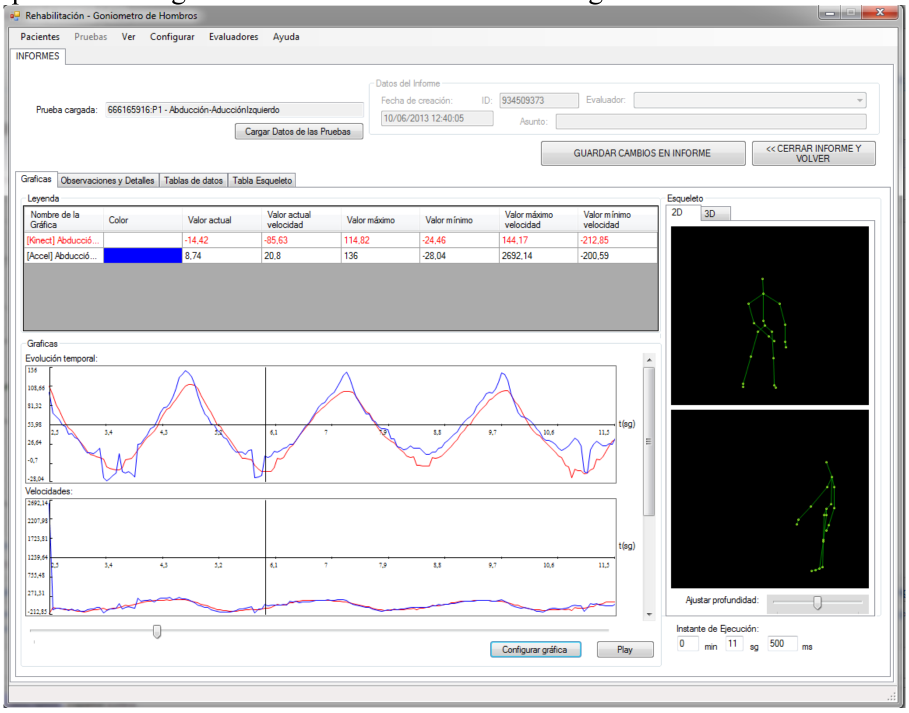
Fig. 13. Información de configuración de pruebas.

4.4 Generación de Informes

Una vez realizadas las pruebas, se generarán informes con la siguiente información:

- Gráficas de la evolución temporal de la amplitud articular, de la velocidad de movimiento y de la fuerza de la articulación bajo estudio.
- Representación de gráficas fasoriales amplitud-velocidad-fuerza articular.
- Representación 3D del esqueleto realizando los movimientos.
- Tablas con los datos guardados y el instante de tiempo.

La Fig. 14 muestra un ejemplo de las curvas generadas en un caso de estudio de TEST_1 para el moviento de abducción-aducción del hombro izquierdo. La Fig. 15 muestra la curva velocidad/ángulo.



Prueba cargada: 666165916:P1 - Abducción-AducciónIzquierdo

Datos del Informe

Fecha de creación: 10/06/2013 12:40:05

ID: 934509373

Evaluador:

Asunto:

Leyenda

Nombre de la Gráfica	Color	Valor actual	Valor actual velocidad	Valor máximo	Valor mínimo	Valor máximo velocidad	Valor mínimo velocidad
[Kinect] Abducció...		-14,42	-85,63	114,82	-24,46	144,17	-212,85
[Accel] Abducció...	🟦	8,74	20,8	136	-28,04	2692,14	-200,59

Instante de Ejecución: 0 min 11 sg 500 ms

Fig. 14. Gráfica de TEST_1 para abducción-aducción del hombro izquierdo.

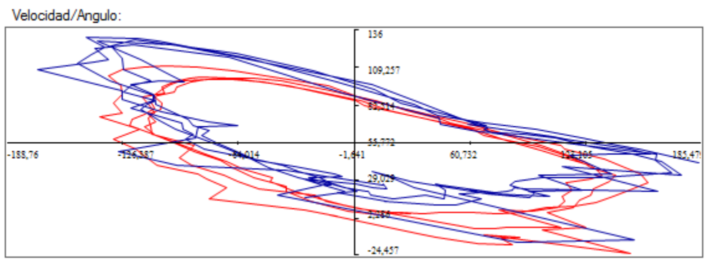


Fig. 15. Curva velocidad-ángulo.

5 Conclusiones y Trabajos Futuros

Este trabajo presenta un sistema de valoración analítica y funcional de extremidades superiores de muy bajo coste basado en el sensor Kinect y acelerómetros. Actualmente dicho sistema se está utilizando para estudiar la valoración funcional de extremidades superiores (principalmente hombro) de mujeres tratadas de cáncer de mama por parte del Grupo de Investigación en Fisioterapia en los Procesos de Salud de la Mujer del Departamento de Fisioterapia de la Universidad de Alcalá.

En cuanto a trabajos futuros se pretende desarrollar un sistema robotizado para abordar ejercicios de recuperación de la articulación del hombro. Éstos estarían encaminados a favorecer un correcto movimiento del mismo, integrando informaciones sensitivas y motoras a través del sistema nervioso central.

Agradecimientos

Este trabajo ha sido posible gracias al proyecto RoboCity2030 II financiado por la Comunidad de Madrid (S2009/DPI-1559).

Referencias

Lokomat. http://www.hocoma.com/products/lokomat/. Revisado el 11/06/2013.

Roboterapist 3D. http://www.instead-technologies.com/products/ robothe-rapist-3d/. Revisado el 11/06/2013.
ASIBOT Robot. http://roboticslab.uc3m.es/roboticslab/robot.php?id_robot=3. Revisado el 11/06/2013.

ARMIN. http://www.sms.mavt.ethz.ch/research/projects/armin. Revisado el 11/06/2013.

Krebs, H. Volpe, B. Williams, D. Celestino, J. Charles, S. Lynch, D. & Hogan, N. Robot-aided neurorehabilitation: a robot for wrist rehabilitation. IEEE Trans Neural Syst Rehabil Eng. 2007 Sep;15(3):327-35.

HAL. http://www.cyberdyne.jp/english/robotsuithal/. Revisado el 11/06/2013.

Zhou, H. & Hu, H. Human motion tracking for rehabilitation—A survey. Biomedical Signal Processing and Control 3, 18. 2008.

Ariel Performance Analysis System - APAS. http://www1.arielnet.com/Main/adw-04.html. Revisado el 11/06/2013.
VICON. http://www.vicon.com/. Revisado el 11/06/2013.

BTS kinematic. http://www.btsbioengineering.com/products/kinematics/. Revisado el 11/06/2013.

NedHombro/IBV. http://autonomia.ibv.org/es/nedhombroibv. Revisado el 11/06/2013.

VirtualRehab. http://virtualrehab.info/. Revisado el 11/06/2013.
SeeMe. http://www.virtual-reality-rehabilitation.com/products/seeme/what-is-seeme. Revisado el 11/06/2013.

Reflexion Measurement Tool. http://www.westhealth.org/news/press-release-west-health-institute-unveils-new-kinect-based-physical-therapy-technology. Revisado el 11/06/2013.

International Agency for Research on Cancer. About cancer mondial [Internet]; 2002. [Actualizado 23-06-2008; citado 03-06-2012]. Disponible en: http://www.dep.iarc.fr/

Lacey, J.J. Devesa, S. Brinton, L. Recent trends in breast cancer incidence and mortality. Environ Mol Mutagen 2002; 39: 82–8.

Cheville, A. Tchou, J. Barriers to rehabilitation folowing surgery for primary breast cancer. J Surg Oncol 2007; 95 (5):409-18.

Kuehn, T. Klauss, W. Darsow, M. Regele, R. Flock, F. Maiterth, F. et al. Long-term morbidity following axillary dissection in breast cancer patients: clinical assessment, significance for life quality and the impact of demographic, oncologic and therapeutic factors. Breast Cancer Res Treat 2000; 64: 275-86.

Bani, H. Fasching, P. Lux, M. Rauh, C. Willner, M. Eder, I. et al. Lymphedema in breast cancer survivors: assessment and information provision in a specialized breast unit. Patient Educ Couns 2007; 66:311-8.

CAPÍTULO 12

DESIGN AND KINEMATIC ANALYSIS OF AN HYBRID ROBOT FOR NEEDLE INSERTION PROCEDURE

L.J. PUGLISI[1], R.J. SALTAREN[1], G.REY PORTÓLES[2], C. GARCIA CENA[1]

[1]Centro de Automática y Robótica, UPM-CSIC; [2] Hospital RUBER Internacional; lisandro.puglisi@alumnos.upm.es

In this work the concept of a needle insertion surgery robot is presented. The mechanism is intended to constraint the needle to the correct insertion path. The mechanism is composed by a spherical parallel manipulator (SPM) 3$\underline{P}$SS-1S coupled to a 3$\underline{RRR}$ serial kinematic chain. The serial kinematic chain provides the positioning of the wrist, while the desired orientation is reached by the state of the spherical parallel manipulator. This kinematic decoupling property is exploited in order to find the inverse and direct kinematics of the robot.

The different configurations solutions for the inverse kinematics are introduced. The workspace and details of the implementations of the final prototype are described.

1 Introduction

Contemporary medicine is toward less invasive and more localized therapy (Vierra, 1995), mainly known as Minimally Invasive Surgery (MIS). One of these procedures is the percutaneous insertion of needles and catheters for biopsy, drug delivery and the placement of internal fiducials as surrogate for tumor location within the body.

Needle insertion intervention offers several advantages over traditional surgery, including less scarring, lighter anesthesia, reduced postoperative pain, reduced complications, and faster discharge from the hospital.

Currently, the conventional needle insertion procedure is a free-hand task with no exact reference for the entry point nor the direction of the needle. This is because the planning insertion process is based on the recognition of markers that the surgeon must place over the patient during the imagining session for trajectory planning. This methodology lacks accuracy and it is very common to make several erroneous insertions until the needle reaches the desired target, which may lead to internal hemorrhage and severe complications.

Robotic technologies can enhance the effectiveness of clinical procedures by coupling many information sources, such as volumetric medical images and perform complex actions in the operating room as reported in (Kazanzides, 2008) (Fichtinger, 2008) (Hager, 2088) (Camarillo, 2004) (Navarro, 2010). Particularly, a robotic device can insert needles with consistent accuracy and can overcome some of the shortcomings of the manual approach as mentioned in (Podder, 2005) (Podder, 2007) (Boctor, 2008) (Salcudean, 2008) (Hata, 2005) (Kronreif, 2006).

In this work we introduce the concept of a novel hybrid (serial–parallel) robotic arm for needle insertion guidance. The spherical parallel wrist is an improved version of the 3$\underline{P}$SU-1S manipulator, whose dimensional synthesis was first reported in (Puglisi, 2012). A discussion on the design and main attributes of the robot is presented. Then, the inverse and forward kinematics is detailed. Finally, the implementation of the prototype is presented remarking the details of its construction.

1.1 Needle Insertion Task Requirements

During an insertion procedure, the surgical instrument must maintain a pre-planned path, defined by two points: the target point and the entry point. The first one is where the tip of the instrument must reach, and the last one is where the needle is inserted into the patient (see Fig. 1(a)). The latter one is selected in order that the desired path of insertion does not interfere with organs nor bones.

The insertion procedure can be summarized in the following steps. First, the tip of the needle must be placed at the entry point. This task requires 3 DoF (see Fig. 1(b)), then the needle must be aligned with the desired path, requiring at least 2 more DoF (see Fig. 1(c)). Once that the needle is correctly placed and aligned with the desired path at the entry point, the insertion procedure begins, generally with a twisting movement of the needle (see Fig. 1(d)).

Therefore, if the insertion is performed manually, a robotic device implemented as a guidance tool (i.e. it keeps the position and orientation) only requires 5 DoF. However, if the insertion procedure is performed autonomously by the robot, it is preferable an additional DoF for the mechanism, since with 5 DoF only one reachable configuration is possible for each point of the desired path, resulting in a screw-like movement during insertion.

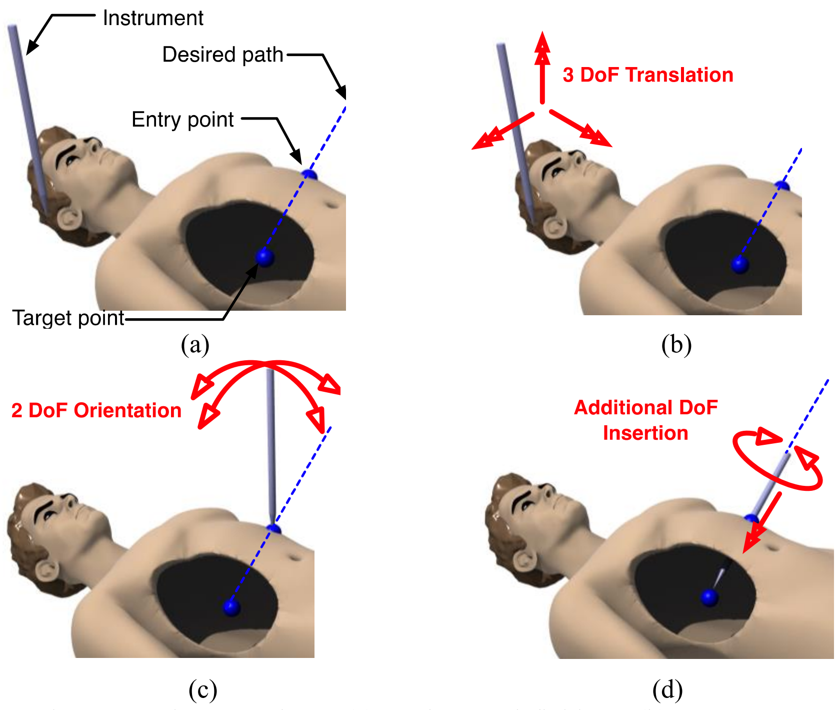


Fig. 1. Insertion procedure. 1(a): Trajectory definition. 1(b): Instrument's positioning. 1(c): Instrument's orientation. 1(d): Instrument's insertion.

2 Hybrid Kinematic Chain Robot

A robotic assistant must be carefully designed, in order to be accurate as well as rigid. Parallel mechanisms offer these physical properties, and some authors have proposed them for the robotic needle insertion guidance (Shoham, 2003) (Nakano, 2009) (Raoufi, 2008). However, the main drawback of a parallel mechanism is their reduced workspace. Although serial mechanisms overcome this problem (Li, 2010), they present a real challenge in order to actuate the last DoFs of the mechanism without introducing backlash (due to the transmission system) or excessive load (direct transmission) at the end effector.

Therefore a better approach would be a design with a decoupled kinematic composed of a serial chain designed for the positioning and a low weight parallel wrist designed for orientation. The concept of this hybrid mechanism is presented in Fig. 2. As it was mentioned above, the mechanism consists of two main parts: the 3R serial arm and the spherical parallel manipulator (SPM).

The arm is mounted over a non-actuated sliding base that permits a gross positioning of the mechanism. A holding device supports the instrument at the end effector.

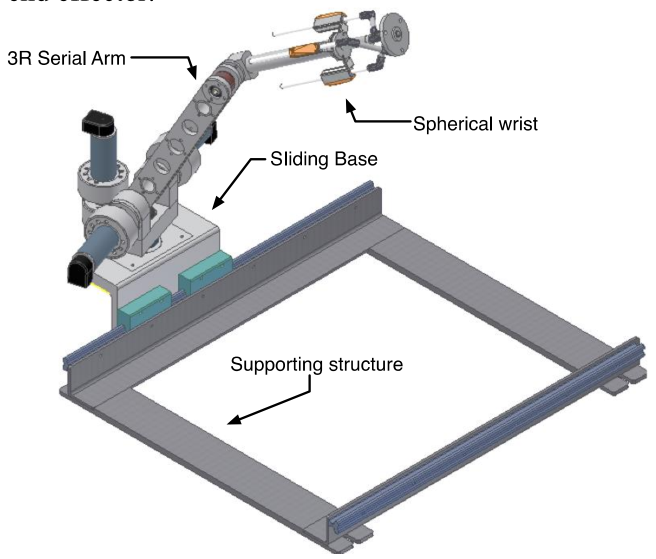


Fig. 2. Hybrid mechanism approach for robotic guided needle insertion.

3 Kinematic Modeling

3.1 Kinematic Decoupling

The decoupling kinematic method can be applied to those mechanisms whose last joints axes intersect at a common point, usually named the center of the wrist P_w, and allows decoupling the kinematic problem into a positioning problem of the P_w and an orientation problem of the EF. The first joints are associated to the positioning problem and the last joints to the orientation problem.

Let consider the core structure of the prototype, and attached a fixed frame O_{xyz} at the base of the mechanism, and a moving frame P_{uvw} to the end effector (EE) as shown in Fig. 3.

The relative orientation of the moving frame to the fixed frame is given by the rotation matrix ${}^{O}\boldsymbol{R}_{P} = [\boldsymbol{u};\ \boldsymbol{v};\ \boldsymbol{w}]$, where $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{w}$ are the direction cosines of the orientation of EE expressed in the fixed frame. The center of the wrist P_w is located at the intersection center of rotation of the SPM and can be expressed as equation (1), where L_{ef} is the distance from the wrist of the mechanism to the tip of the EE.

$$P_w = P - L_{ef}\,\hat{\boldsymbol{w}} \tag{1}$$

Given the position of the wrist and the orientation of the EE, there are two problems to be solved:
(1) **Position Problem**. Given P_w, the state of the first joints must be found.
(2) **Orientation Problem**. The state of the SPM must be found, in order to achieve the desired orientation of the EE. Therefore, the SPM must provide the orientation given by equation (2).

$$\boldsymbol{R}_{d_{456}} = \boldsymbol{R}_{\$}^{-1}\,{}^{O}\boldsymbol{R}_{uvw} \tag{2}$$

Where $\boldsymbol{R}_{\$}$ is the contribution of the serial kinematic chain to the orientation of the EE, employed for the wrist's positioning.

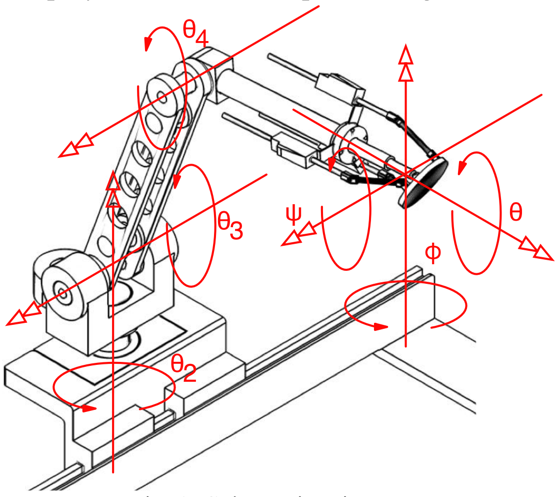


Fig. 3. Schematic Diagram.

3.2 Inverse Kinematics

3.2.1 Inverse Kinematics of the 3R serial chain

Let consider the mechanism at any given configuration as shown in Fig. 4, and attach a fixed reference frame O_{xyz} at the base of the kinematic chain, and assume that the position of the wrist $P_w = [P_{wx}, P_{wy}, P_{wz}]$ is given by equation (1).

As it can be seen, the axes of action of the second and third joint are parallel, then the movement of the wrist will be always constrained to a plane, where the direction of the normal to this plane will depend of the state of the first joint.

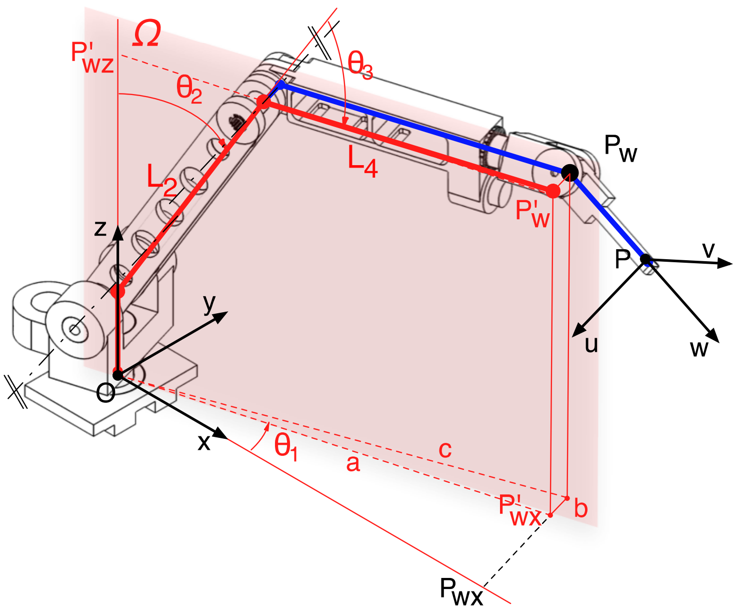


Fig. 4. Considerations for the kinematic analysis.

In order to simplify the analysis let's consider the plane *Ω*, that contains the movement of link L$_2$ and the origin of the fixed frame O_{xyz}. The projection of P_w on the plane is given by $P_{0w} = [P_{0wx}, P_{0wz}]$, and the projection of points P_w and P_{0w} on the plane z = 0, defines the triangle *abc*. Now, let's consider the top view of the mechanism shown in Fig. 5.

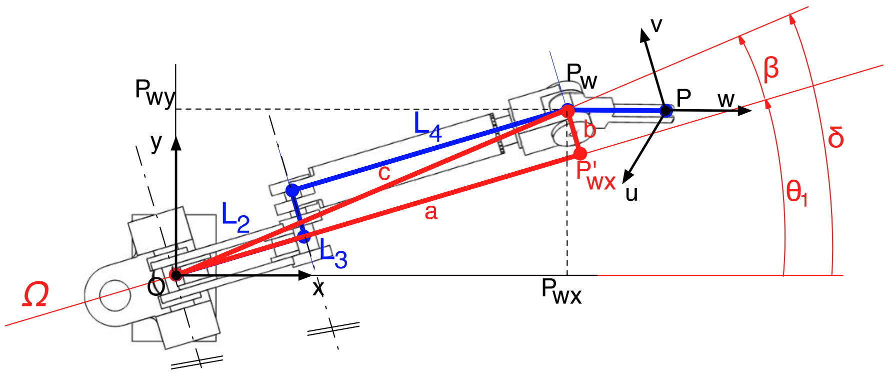


Fig. 5. Top view of the 3R kinematic chain.

It is easily seen that the state of the first join can be found as:

$$\theta_1 = \delta - \beta \tag{3}$$

and it can be expressed in terms of the elements of the triangle abc as equation (4).

$$\theta_1 = Atan2(P_{w_y}, P_{w_x}) - \sin^{-1}\left(\frac{b}{c}\right) \tag{4}$$

where b=L_3 and:

$$c = \sqrt{{P_{w_x}}^2 + {P_{w_z}}^2} \tag{5}$$

The state of the second and third joints can be found by considering the 2R serial kinematic chain conformed by link L_2 and the projection of link L_4 on the plane □Ω. Hence, the state of the third joint is given by equation (6).

$$\theta_3 = \cos^{-1}\frac{{P'_{w_x}}^2 + {P'_{w_z}}^2 - {L_2}^2 - {L_4}^2}{2\,L_2\,L_4} \tag{6}$$

The state of the second joint is given by equation (7).

$$\theta_2 = Atan2(\sin\theta_2, \cos\theta_2) \tag{7}$$

where:

$$\sin\theta_2 = \frac{(L_2 + L_4\,\cos\theta_3)\,P'_{w_x} - (L_4\,\sin\theta_3)\,P'_{w_z}}{\Delta}$$
$$\cos\theta_2 = \frac{(d_2.\sin\theta_3)\,P'_{w_x} + (L_2 + L_4\,\cos\theta_3)\,P'_{w_z}}{\Delta} \tag{8}$$

and

$$\Delta = {L_2}^2 + {L_4}^2 + 2\,L_2\,L_4\,\cos\theta_3 \tag{9}$$

The 3R kinematic chain will admit four possible configurations (see Fig. 6) defined by the angular relations summed up in Table 1.

Table 1. Possible configurations of the 3R Serial Chain.

θ_1	θ_2	θ_3	Configuration
$\delta - \beta$	θ_2^+	θ_3^+	Elbow Left-up
	θ_2^-	θ_3^-	Elbow left-down
$\delta + \beta + \pi$	$-\theta_2^+$	$-\theta_3^+$	Elbow right- up
	$-\theta_2^-$	$-\theta_3^-$	Elbow right - down

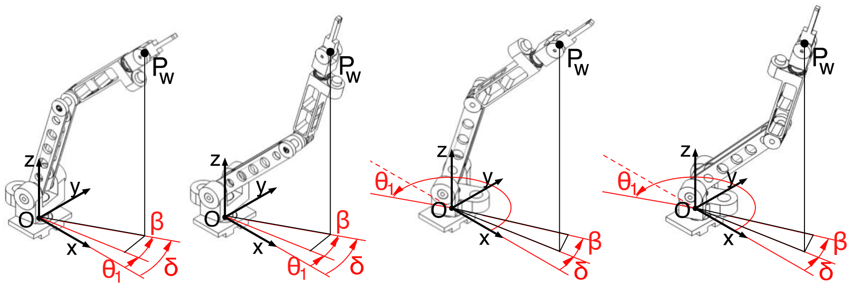


Fig. 6. 3R Configurations.

3.2.2 Inverse Kinematics of the Spherical PM

The inverse kinematic problem of the SPM, can be easily solved by considering:

1. The extreme of link L$_7$ (i.e. $B_i = [B_{ix}, B_{iy}, B_{iz}]$) can only move over the surface of the sphere ζ_i with center in $C_i = [C_{ix}, C_{iy}, C_{iz}]$ and radius L7 (see Fig. 7), given by (10):

$$(B_{i_x} - C_{i_x})^2 + (B_{i_y} - C_{i_y})^2 + (B_{i_z} - C_{i_z})^2 = L_7{}^2 \tag{10}$$

2. B_i corresponds with the distal extreme of the linear actuator, thus, B_i can only move over the line l_i defined by (10), whose direction vector $\boldsymbol{l}_i$ = *[0, 0, 1]* is associated to the axis of the prismatic actuator.

$$l_i: \quad B_i = A_i + \lambda_i \hat{\boldsymbol{l}}_{\boldsymbol{i}} \tag{11}$$

Equating (10) and (11), it can be demonstrated that $B_{ix} = A_{ix}$ and $B_{iy} = A_{iy}$. Taking this result into (10), B_{iz} can be found according to (12)

$$B_{i_z} = C_{i_z} \pm \sqrt{L_7{}^2 - (A_{i_x} - C_{i_x})^2 - (A_{i_y} - C_{i_y})^2} \tag{12}$$

Even though (12) has two possible solutions B^{-}_{iz} and B^{+}_{iz} whether choosing − or + sign, only B^{-}_{iz} is considered, since the B^{+}_{iz} solution leads to collisions between the link L$_7$ and the moving platform. This can be seen in Fig. 7 where the distal extreme of L$_7$ (considering an ideal spherical joint with a full range of movement), points upwards for the B^{+}_{iz} solution, and as a consequence L$_7$ (dashed green line) will collide with the end effector.

Then, the state of the prismatic actuator is given by (13).

$$\lambda_i = B_{i_z} - A_{i_z} \tag{13}$$

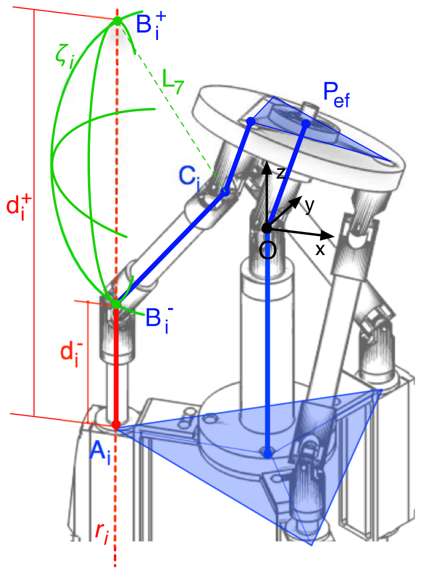


Fig. 7. Schematic wire representation of the 3PSS-1S. The triangular shade defines the base and moving platforms. The Intersection of the spherical characteristic of link L_7 and linear motion of point B_i is presented in green.

3.3 Direct Kinematics

Let's attach a reference frame O_{xyz} at one end of the horizontal guide, and let's consider as the initial configuration the one presented in Fig. 8. The position of the EE ${}^{O}P_{ef}$, expressed at the reference frame can be found by equation (14).

$$^{O}P_{ef} = \boldsymbol{M}_{\$}\,{}^{O}P_{w_0}^{T} + \boldsymbol{M}_{d_{456}}\,{}^{w}L_{ef_0}^{T} \tag{14}$$

Where,

$$\boldsymbol{M}_{\$} = \Pi_{i=1}^{3}\boldsymbol{M}_{\$_i} \tag{15}$$

is the overall screw displacement matrix associated to the movement of the first three joints whose parameter are summed up in Table 2. ${}^{O}P_{w0} = [L_{x0} - L_3, -L_{y0}, L_1 + L_2 + L_4 + L_5, 1]$ is the position of the wrist at the initial configuration expressed in homogenous coordinate referenced at O_{xyz}. $\boldsymbol{M}_{d456}$ is the contribution of the parallel mechanism to the orientation of the EE according to the state of the prismatic actuators d_4, d_5 and d_6. (See (Puglisi, 2013) for details). ${}^{w}L_{ef0} = [0, 0, L_6, 1]$ is the position of the EE at the initial configuration expressed in homogenous coordinate at the reference frame attached at the wrist P_{uvw}. The orientation of the EE expressed at the reference frame O_{xyz} is given by equation (16).

$$^{O}\boldsymbol{R} = \boldsymbol{R}_{\$}\,\boldsymbol{R}_{d_{456}} \tag{16}$$

Where, $\boldsymbol{R}_{\$}$ is the contribution to the orientation of the EE of the first three joints, given by the elements of the first three row and first three columns of the $\boldsymbol{M}_{\$}$. $\boldsymbol{R}_{d456}$ is the contribution of the parallel mechanism to the orientation of the EE.

Table 2. Screw's parameters

Joint i	$\$_i$	$\boldsymbol{s}_{oi}$	$\boldsymbol{s}_i$	θ_i	d_i
1	$\$_1$	$[L_{xo},-L_{yo}, 0]$	[0,0,1]	θ_1	0
2	$\$_2$	$[0,-L_{yo}, L_1]$	[1,0,0]	θ_2	0
3	$\$_3$	$[0,-L_{yo},L_1+L_p]$	[1,0,0]	θ_3	0

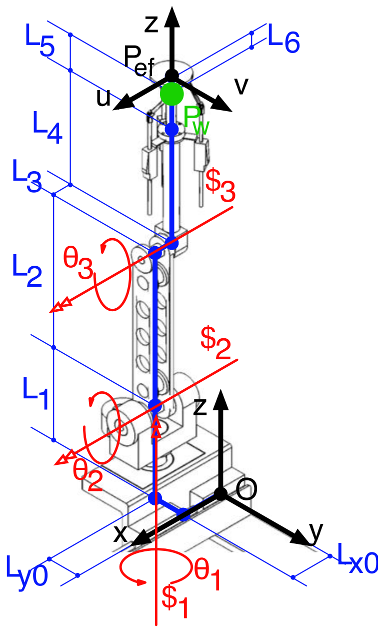


Fig. 8. Initial configuration. Definitions of screws associated to each joint.

4 Implementation

4.1 Construction Details

The prototype consists of a hybrid mechanism mounted over a passive sliding base, which facilitates the gross position of the mechanism. As it was mentioned previously, the hybrid mechanism is composed of a 3R serial kinematic chain for positioning the center of the wrist and a 3DoF parallel spheric mechanism that gives the final orientation of the end effector. (See Fig. 8).

Most of the pieces are made of aluminum, except for those that must support torques and efforts as the shafts and the supporting structure that are stainless steel machined pieces. In order to eliminate friction and guarantee a correct alignment of the shafts, ball bearings and thrust needle roller bearings are employed. The Sliding Base is composed of two sliding elements with adjustable clearance mounted on a rail (DryLin RT Linear Guide System) that permits a smooth motion in the 600mm range.

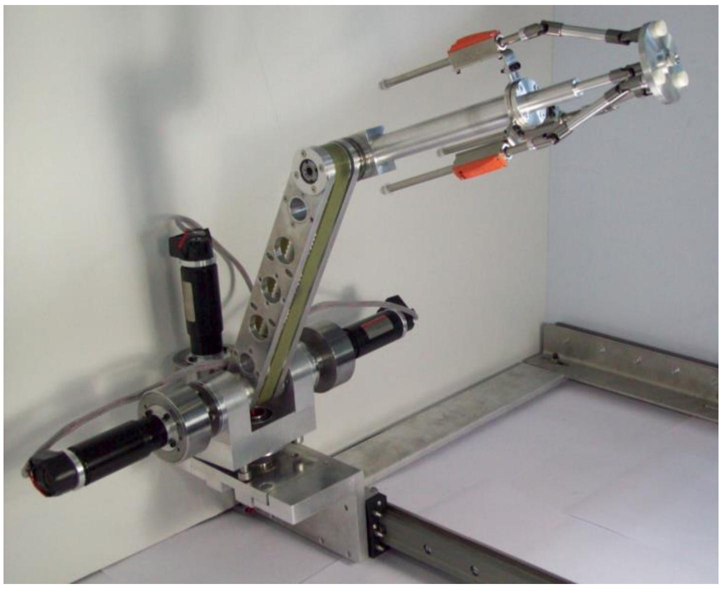


Fig. 9. Hybrid kinematic chain robot.

In order to minimize the weight of the mechanism at the end effector, the actuations of the 3R serial chain are located near the base, forcing to use a transmission system for the first and third joint of the mechanism as detailed in Fig. 10.a. The transmission system consists of two parallel shafts, each of them with a pulley attached to it, and a synchronous belt that transfer motion from one shaft to the other. This system allows having the actuation system (motor, gear, and encoder) at the base of the mechanism and not at the location of the joint.

The actuation system for the joints of the 3R serial arm, is composed by 24V brushed DC Maxon R motor, Harmonic drive, and HEDM-550 optical encoder from Agilent Technologies.

The optical encoders used for the 3R serial mechanism are relative sensors, thus an initial calibration must be done every time the mechanism is turn on. Therefore, three micro switches placed at specific location on the mechanism are employed as end of carrier sensor, providing a correlation of the counts of the encoders and the initial configuration of the mechanism.

The dimensions of the spheric wrist were taken and adapted from a previous analysis for a 3$\underline{\text{P}}$SU-1S spheric wrist (Puglisi, 2012).

The challenge for the 3$\underline{\text{P}}$SS-1S spheric wrist was to find a spheric joint that didn't limit the behavior of the mechanism. A very simple way to achieve it is to adapt a commercial available universal joint, by allowing to rotate freely at its location along its normal axis (see Fig. 10.b).

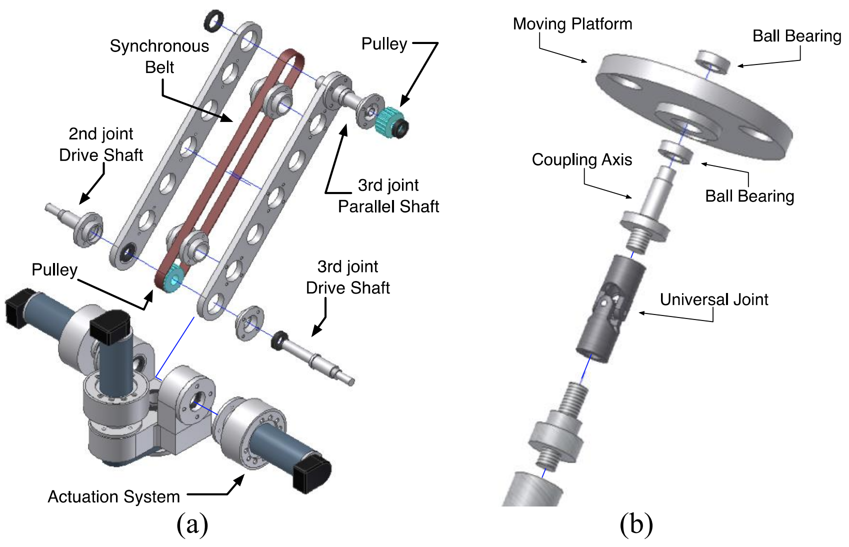


Fig. 10. Construction details.

For the spheric mechanism, 3 Linear DC-Servomotors LM 1247 080-01 of Faulhaber were used. These linear motors are provided with hall sensors that allow ultra low speeds and high positioning resolution (1/3000 pole pitch), thus no additional encoder is needed. The linear motor is composed of a magnetized rod that slides inside a controlled magnetic field, providing a controlled linear motion that can freely rotate over its axis of action. This characteristic is advantageously used for the adaptation of the universal joints placed at the end of the rod of the linear motors.

4.2 System Architecture

In order to control the Maxon motor, a six axes Galil motion controller DMC-2163 with 3 four-quadrant DC Servoamplifier ADS 50/550/5 of Maxon, connected via Ethernet is used. Three MCLM 3003/06 C external motion controller with CAN open interface are use in order to control the linear motors. These motion controllers are connected to the dedicated PC via an USB-CAN IXXAT converter. The schematic diagram of the system architecture is presented in Fig. 11.

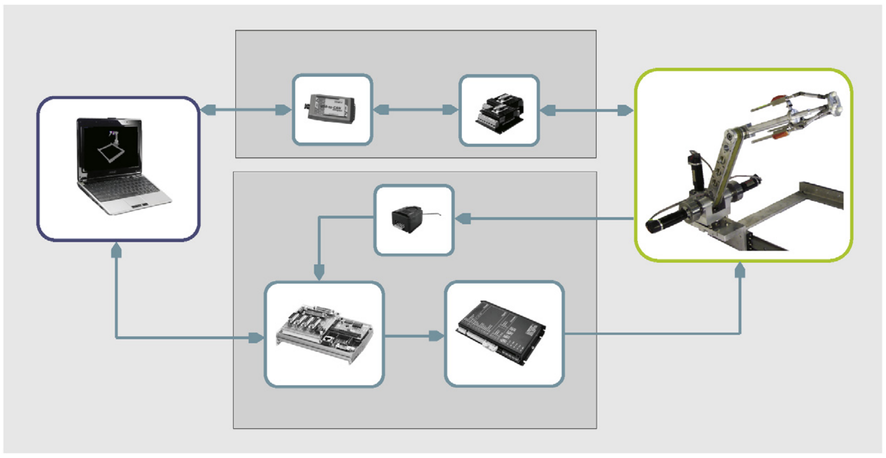


Fig. 11. Hardware architecture of the robotic arm.

5 Conclusion and discussion

In this work, the concept for a needle insertion surgery robot intended to assist and guide the needle during a needle insertion surgery is summed up. The prototype consists of a hybrid mechanism: a 3R serial kinematic chain for positioning its wrist, and a 3PSS-1S spheric parallel mechanism for the orientation of the EE. This mechanism provides of 6DoF movement plus and additional sliding based that allows a first gross setup. The mobility of the mechanism is redundant for the task, since only 5DoF are required. However, this redundancy allows to deal with the reduced workspace of the spherical wrist, avoid its singularities and provide multiples configurations of the mechanism for the same path of insertion.

Acknowledgements

The authors would like to thank the financial support of the Spanish Government through the CICYT Project Ref. DPI2009-08778 and also to Comunidad de Madrid which supports the project ROBOCITY2030-II Ref. P2009/DPI-1559. Mr. Puglisi belongs to the MAEC-AECID grant programs.

References

Boctor, E.M.; Choti, M.A.; Burdette, E.C.; Webster, R.; 2008; Three-dimensional ultrasound-guided robotic needle placement: an experimental evaluation, International Journal (2008). http://dx.doi.org/10.1002/rcs.184.

Camarillo, D.B.; Krummel, T.M.; Salisbury, J.K.; 2004; Robotic technology in surgery: past, present, and future, American Journal of Surgery (4A Suppl.) (2004) 2S–15S. http://dx.doi.org/10.1016/j.amjsurg.2004.08.025.

Fichtinger, G.; Kazanzides, P.; Okamura, A.M.; Hager, G.D.; Whitcomb, L.L.; Taylor, R.H.; 2008; Surgical and interventional robotics: part II: surgical CAD–CAM systems, IEEE Robotics & Automation Magazine/IEEE Robotics & Automation Society (3) (2008) 94–102. http://dx.doi.org/10.1109/MRA.2008.927971.

Hager, G.D.; Okamura, A.M.; Kazanzides, P.; Whitcomb, L.L.; Fichtinger, G.; Taylor, R.H. ; 2008; Surgical and interventional robotics: part III: surgical assistance systems, IEEE Robotics & Automation Magazine/IEEE Robotics & Automation Society (4) (2008) 84–93. http://dx.doi.org/10.1109/MRA.2008.930401.

Hata, N.; Hashimoto, R.; Tokuda, J.; Morikawa, S.; 2005; Needle guiding robot for MR-guided microwave thermotherapy of liver tumor using motorized remote-center-of-motion constraint, in: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pp. 1652–1656, April. http://dx.doi.org/10.1109/ROBOT.2005.1570350.

Kazanzides,P.; Fichtinger, G.; Hager,G.; Okamura, A.; Whitcomb, L.; Taylor, R.; 2008; Surgical and interventional robotics—core concepts, technology, and design, IEEE Robotics & Automation Magazine (2) (2008) 122–130. http://dx.doi.org/10.1109/MRA.2008.926390.

Kronreif, G. ; Füurst, M.; Ptacek, W.; Kornfeld, M.; Kettenbach, J.; 2006; Robotic system for image guided therapie B-RobII, in: I. J. Rudas (Ed.), 15th International Workshop on Robotics in Alp-Adria-Danube Region, RADD 2006, Balantonfüred, Lake Balaton, Hungary.

Li, J.; Wang, S.; Wang, X.; He, C.; 2010; Optimization of a novel mechanism for a minimally invasive surgery robot, The International Journal of Medical Robotics + Computer Assisted Surgery: MRCAS 6 (1) (2010) 83–90. http://dx.doi.org/10.1002/rcs.293.

Nakano, T.; Sugita, N.; Ueta, T.; Tamaki, Y.; Mitsuishi, M.; 2009; A parallel robot to assist vitreoretinal surgery, International Journal of Computer Assisted Radiology and Surgery 4 (6) (2009) 517–526. http://dx.doi.org/10.1007/s11548-009-0374-2.

Navarro, J.S.; Garcia, N.; Perez, C.; Fernandez, E.; Saltaren, R. ; Almonacid, M.; 2010; Kinematics of a robotic 3UPS1S spherical wrist designed for laparoscopic applications, The International Journal of Medical Robotics + Computer Assisted Surgery: MRCAS (3) (2010) 291–300. http://dx.doi.org/10.1002/rcs.331.

Podder, T.; Sherman, J.; Clark, D.; Messing, E.; Rubens, D.; Strang, J.; Liao, L.; Brasacchio, R.; Zhang, Y.; Ng, W.; Yu, Y.; 2005; Evaluation of Robotic Needle Insertion in Conjuction with in Vivo Manual Insertion in the Operating Room, IEEE, 2005, http://dx.doi.org/10.1109/ROMAN.2005.1513758.

Podder, T.K.; Ng, W.S.; Yu, Y.; 2007; Multi-channel robotic system for prostate brachytherapy, in: Conference Proceedings: . . .Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference, 2007, pp. 1233–1236. http://dx.doi.org/10.1109/IEMBS.2007.4352520.

Puglisi, L.J.; Saltaren, R.J.; Moreno, H.A.; Cárdenas, P.F.; Garcia, C. ; Aracil Santonja, R.; 2012; Dimensional synthesis of a spherical parallel manipulator based on the evaluation of global performance indexes, Robotics and Autonomous Systems 60 (8) (2012) 1037–1045. http://dx.doi.org/10.1016/j.robot.2012.05.013.

Puglisi, L.J.; Saltaren, R.J.; Portoles, G.R.; Moreno, H.; Cardenas, P.F.; and Garcia, C.; 2013. Design and kinematic analysis of 3pss-1s wrist for needle insertion guidance. Robotics and Autonomous Systems, 61(-), http://doi:10.1016/j.robot.2013.02.001.

Raoufi, C.; Goldenberg, A.A.; Kucharczyk, W.; 2008; A new hydraulically/pneumatically actuated MR-compatible robot for MRI-guided neurosurgery, in: 2008 2nd International Conference on Bioinformatics and Biomedical Engineering, pp. 2232–2235. http://dx.doi.org/10.1109/ICBBE.2008.889.

Salcudean, S.E.; Prananta, T.D.; Morris, W.J.; Spadinger, I.; 2008; A Robotic Needle Guide for Prostate Brachytherapy, Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on, May 2008, pp. 2975-2981. http://dx.doi.org/10.1109/ROBOT.2008.4543662.

Shoham, M.; Burman, M.; Zehavi, E.; Joskowicz, L.; Batkilin, E.; Kunicher, Y.; 2003; Bone-mounted miniature robot for surgical procedures: concept and clinical applications, IEEE Transactions on Robotics and Automation (5) (2003) 893–901. http://dx.doi.org/10.1109/TRA.2003.817075.

Vierra, M.; 1995, Minimally invasive surgery, Annual Review of Medicine, pp. 147-158. http://dx.doi.org/10.1146/annurev.med.46.1.147.

CAPÍTULO 13

SISTEMA DE REHABILITACIÓN COGNITIVA PARA LA ASISTENCIA EN ACTIVIDADES COTIDIANAS DE PACIENTES TRAS SUFRIR UN ACCIDENTE CEREBRO-VASCULAR

JAVIER ROJO[1], JOSÉ M. COGOLLOR[1], MATTEO PASTORINO[2], ALESSIO FIORAVANTI[2], JOSE BREÑOSA[1], MARÍA TERESA ARREDONDO[2], MANUEL FERRE[1], RAFAEL ARACIL[1], JOSE MARÍA SEBASTIÁN Y ZÚÑIGA[1]

[1] Centre for Automation and Robotics CAR (UPM-CSIC), José Gutiérrez Abascal 2, 28006, Madrid, Spain: javier.rojo.lacal@upm.es, jm.cogollor@upm.es, jose.brenosa@upm.es, m.ferre@upm.es, rafael.aracil@upm.es, jsebas@etsii.upm.es.

[2] Life Supporting Technologies, ETSIT-UPM, Avda. Complutense 30, Madrid, Spain: mpastorino@lst.tfo.upm.es, afioravanti@lst.tfo.upm.es, mta@lst.tfo.upm.es.

Según la Organización Mundial de la Salud cada año 15 millones de personas sufren un accidente cerebro-vascular. Aproximadamente un 33% sufren, tras este suceso, algún tipo de discapacidad permanente (MacKay and Mensah, 2004). Este artículo se centra en describir el diseño e integración de un sistema cuyo principal objetivo es proporcionar rehabilitación cognitiva a pacientes que sufren discapacidad tras sufrir un accidente cerebro-vascular.

El nivel de dependencia de dichos pacientes implica un presupuesto cuantioso siendo necesario que estén supervisados y ayudados por un terapeuta.

El sistema de rehabilitación cognitiva permite que estos pacientes no dependan de la estancia en el hospital o centro de rehabilitación, donde generalmente la rehabilitación y alta médica se limitan al aspecto físico, permitiendo que la rehabilitación se pueda trasladar a la vivienda del paciente en un ambiente más familiar y cercano. Dicho sistema cumple la función de terapeuta virtual interactuando con el paciente informándole sobre los errores cometidos y cómo corregirlos. Esta información se visualiza a través de una pantalla en la cual se le muestran ayudas, mensajes de texto y/o animaciones, que también pueden ser transmitidas a través de un reloj inteligente como medio de interacción con el paciente en caso de ser necesario.

Para que el sistema detecte los errores cometidos, el sistema tiene que procesar en tiempo real cierta información relevante que se adquiere del escenario por medio de sensores (de fuerza y aceleración) colocados en los objetos que el paciente utiliza para realizar la tarea en cuestión y por cámaras.

El desarrollo de este sistema para rehabilitación cognitiva se enmarca dentro del proyecto europeo CogWatch, coordinado por la Universidad de Birmingham. Actualmente, se encuentra en la fase de evaluación del primer prototipo con el que se están realizando las primeras pruebas tanto técnicas como con pacientes.

1 Introducción

Un reciente estudio realizado en España cifra la tasa de incidentes de eventos cerebrovasculares en sujetos de 18 años o más en 186,96 casos por cada 100.000 habitantes y año. Este estudio ha dado estimaciones muy robustas pues se carecía de estudios acerca de la incidencia de ictus en España con grandes denominadores sobre la población. Además dicho estudio ha concluido un aumento significativo de las cifras de incidencia en los próximos años debido al progresivo envejecimiento poblacional (Díaz-Guzmán et al., 2012).

Existen dos tipos principales de accidentes cerebrovasculares en función de su etiología: Los más comunes son los que tienen como causa la obstrucción, conocidos como isquemia y, en segundo lugar, se producen los accidentes causados por la fuga de sangre en el tejido cerebral, conocido como hemorragia. La isquemia está clasificada por la forma en que el bloqueo se ha producido. Un coágulo que se forma en una región del sistema circulatorio fuera del cerebro puede viajar a través de los vasos sanguíneos y obstruir una arteria del cerebro. Este coágulo libre se llama émbolo y comúnmente se forma en el corazón. Cuando una isquemia es causada por un émbolo es conocido como accidente cerebrovascular embólico o embolia. Por otro lado, en el caso de la aparición de coágulos de sangre en una arteria del cerebro, que permanecen fijos en la pared de la arteria, hasta crecer lo suficiente como para bloquear el flujo de sangre a través de esta, el episodio es denominado accidente cerebrovascular trombótico. La distinta naturaleza de las causas se traduce en una gran incidencia de casos.

Los datos publicados por la Organización Mundial de Accidentes Cerebrovasculares indican que una tercera parte de las personas que han sufrido un accidente cerebrovascular padecen, tras este suceso, algún tipo de discapacidad por la pérdida de facultades motoras o cognitivas que les impiden la práctica de actividades cotidianas (World Stroke Organization - Home, n.d.).

Por lo general, los profesionales de la salud reconocen que el plazo de atención al ictus es típicamente corto, el hospital a menudo se basa y centra en la rehabilitación física en lugar de la rehabilitación cognitiva. Independientemente de su estado funcional, los pacientes son dados de alta con frecuencia por motivos físicos con el supuesto de que la rehabilitación cognitiva, de ser necesaria, continuará en casa del paciente.

Sin embargo, los métodos actuales de tratamiento se ven obstaculizados por diversos motivos como pueden ser la falta de reconocimiento de la prevalencia y el impacto de la enfermedad, la falta de recursos de los terapeutas y las pruebas limitadas de una terapia eficaz en la que basar el tratamiento.

Entre las posibles secuelas que pueden derivar de un ictus, se pueden destacar dos fenómenos comúnmente asociados y que imposibilitan, en gran medida, la realización de actividades cotidianas (Liepmann, 1908) y (Hermsdörfer et al., 2003). Estos fenómenos son la Apraxia y el ADS (por sus siglas en inglés, Action Disorganization Syndrome) que se definen como:

- **Apraxia**: Un trastorno neurológico de la capacidad de movimiento intencional aprendido que no se explica por los déficits en el sistema motor primario o en los sistemas sensoriales (Rothi y Heilman, 1997).
- **ADS**: Errores cognitivos al realizar tareas que se conforman de múltiples pasos (Morady y Humphreys, 2009).

El sistema propuesto tiene como objetivo servir de gran ayuda a pacientes que han sufrido un accidente cerebro-vascular, algunos de los cuales padecen secuelas como pueden ser Apraxia o ADS durante la ejecución de tareas cotidianas para así garantizar su independencia y descargarles de sus sucesivas visitas al correspondiente hospital y/o centro de rehabilitación.

Los pacientes que sufren estos síndromes, a pesar de mantener sus habilidades motoras, cometen errores cognitivos durante tareas cotidianas que antes realizaban de una forma automática. Los pacientes suelen cometer errores de omisión al realizar una tarea.

En el siguiente capítulo se presenta una visión general del sistema propuesto y el modo en que se ha enfocado el problema que supone la rehabilitación cognitiva. En las secciones 3 y 4 se describirá la arquitectura general del software, su metodología de diseño y la experiencia que presenta al usuario final. Posteriormente, en la sección 5, se describe el estudio realizado para la evaluación técnica del primer prototipo del sistema. Finalmente la sección 6 presenta las conclusiones obtenidas de la evaluación técnica y las futuras mejoras en base a los resultados obtenidos.

2 Hacia la rehabilitación cognitiva

Dicha rehabilitación se pretende que sea personalizada y en la propia casa del paciente.

Está claro que las tareas que un ser humano realiza en su vida diaria son de diversos tipos y de distintos niveles de complejidad. Así, se puede decir que las tareas que un ser humano desarrolla de forma cotidiana se clasifican en cuatro clases generales: preparación de comida, preparación de bebida, aseo y vestimenta.

Durante el tiempo de vida y de desarrollo y de la evaluación del sistema, se elegirá una tarea de cada clase mencionada anteriormente como apropiada y representativa de proporcionar rehabilitación. Actualmente se está trabajando en la tarea de preparación de un té, la cual se enmarca dentro de la clase "preparación de bebida".

Para proporcionar la rehabilitación, el sistema se compone de una serie de dispositivos inteligentes que permiten proporcionar una sesión de rehabilitación de forma remota sin la necesidad de la presencia de un terapeuta.

Sin embargo, para que el sistema tenga el efecto que se persigue, no solo se deben escoger los dispositivos adecuados sino que es imprescindible entender la ejecución de la tarea a realizar en la rehabilitación ya que el sistema se puede asemejar a un terapeuta virtual, por lo que debe tener la capacidad de saber cuándo existe o no un error, debido a que distintos individuos pueden ejecutar la tarea de forma distinta pero logrando el objetivo final.

Por tanto, la tarea debe ser analizada en detalle y descompuesta en su más bajo nivel. Y es que, en términos generales, al seleccionar una tarea en concreto, ésta se debe subdividir y ramificar en diferentes acciones básicas las cuales deben ejecutarse secuencialmente aunque no siempre en un orden específico. Esta técnica se debe a Cooper y Shallice (Cooper & Shallice, 2000) y se basa en la idea de que segmentar las tareas garantiza que ésta puede realizarse de diferentes formas si se conocen las acciones básicas que la componen. (Ver Fig. 1)

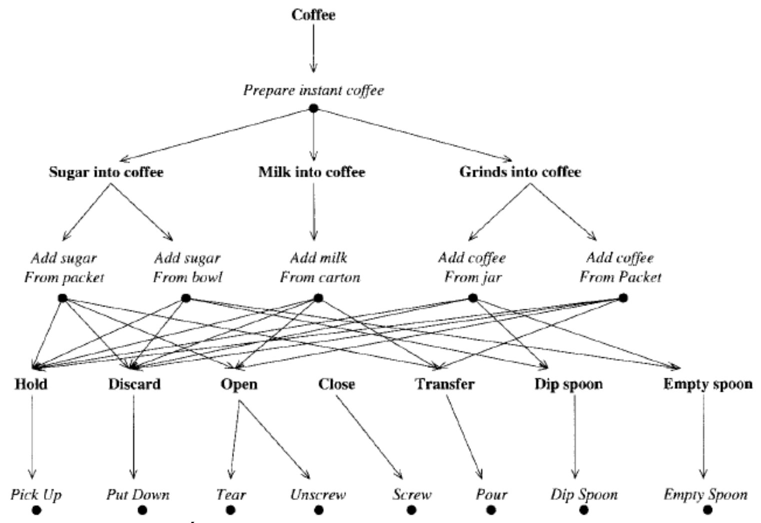


Fig. 1. Árbol de acciones básicas para preparar café

Esta técnica es muy relevante por dos motivos: en primer lugar, si una misma tarea puede realizarse de formas distintas, no es necesario imponer al paciente un orden específico por lo que se le permite mayor libertad y posibilidades en sus formas de proceder; y en segundo lugar, el sistema pueda adaptarse al mismo tiempo para utilizarse en diferentes tareas ya que existen muchas acciones compartidas por diferentes actividades incluidas en una misma clase, es decir, muchos de los movimientos que se realizan para preparar un té se encuentran también si se desea preparar cualquier otra bebida.

Desde el punto de vista de obtener algún tipo de diagnosis mediante el sistema, la segmentación de la tarea también puede ayudar a conocer en qué acciones y movimientos específicos el paciente presenta más dificultades.

Como ya se ha mencionado anteriormente, la rehabilitación se precisa que se lleve a cabo en casa del propio paciente para que éste se sienta más cómodo y en un lugar totalmente familiar y cercano. Existen algunos estudios en los cuales se demuestran las ventajas de proporcionar rehabilitación doméstica, (Gamberini, 2002) y (Scopelliti, 2005).

Para ello, el sistema se compone de tres módulos principales: objetos instrumentados que el paciente usa para realizar la tarea en cuestión, un reloj que vibra al cometer ciertos errores y finalmente, un monitor/monitores para advertir al paciente de sus errores e indicarle cómo corregirlos. (ver Fig. 2)

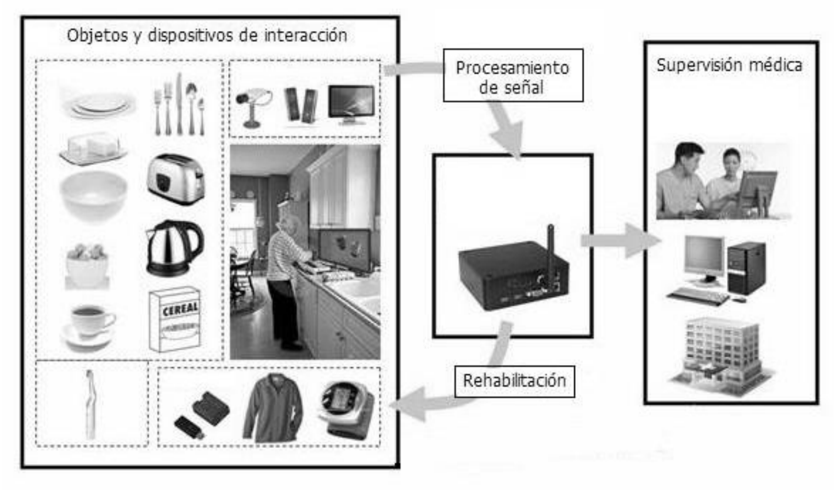


Fig. 2. Arquitectura general del sistema

Adicionalmente, se utilizan dispositivos de video, como, por ejemplo, Kinect^{TM}, los cuales sirven para monitorizar la ejecución de la sesión y a partir del procesamiento de imagen obtener la posición de las manos del paciente para su posterior estudio y evaluación.

Cabe destacar que los objetos instrumentados son los mismos que el propio paciente usa diariamente como pueden ser cucharas, teteras o jarras de agua. De esta manera se garantiza que el paciente se sienta en un entorno familiar y conocido mediante el uso de sus propios objetos y no se tenga que recurrir a implantar hardware adicional específico en su casa que aumente el coste de instalación.

Como el sistema deberá garantizar en el futuro que las necesidades del paciente sean satisfechas, teniendo en cuenta las diferentes clases de actividades mencionadas, es preciso disponer de componentes que aseguren la flexibilidad y portabilidad del prototipo. En este caso se ha utilizado un ordenador "All-In-One" y, alternativamente, un sistema soportado por "tablets", que el paciente puede transportar a lo largo de las diferentes habitaciones requeridas (ver Fig. 3). Éstas no suponen un coste muy elevado ya que es cada vez más común su presencia en nuestras casas. El ordenador "All-In-One", al llevar una CPU integrada en el propio monitor, garantiza ergonomía y uso práctico al poder servir al mismo tiempo de pantalla de televisión.

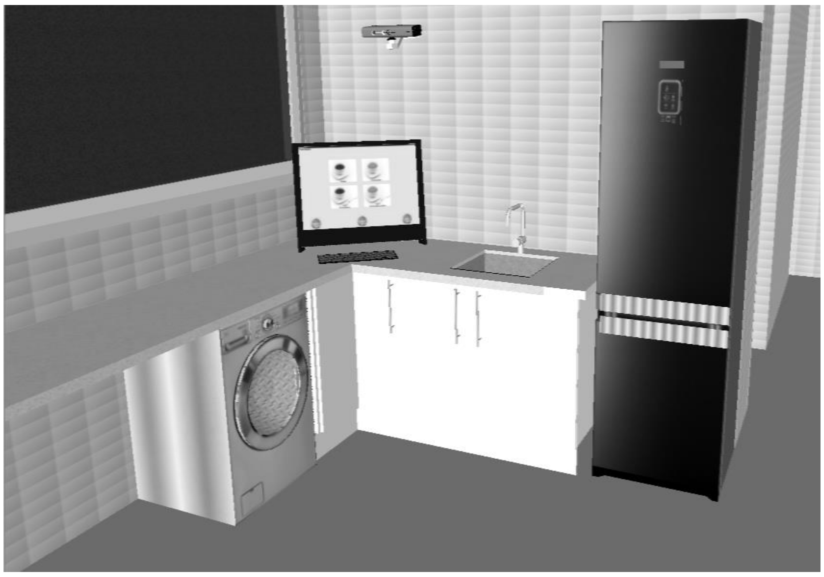


Fig. 3. Diseño conceptual del sistema instalado en la cocina

La principal función de estos monitores es proporcionar al paciente una serie de ayudas según el nivel de riesgo del error cometido. Estas ayudas se pueden clasificar en: semánticas, donde la información del error es enviada en forma visual, sensomotriz o auditiva, y dinámicas, las cuales proporcionan información más precisa de cómo se puede corregir el error.

Este tipo de ayudas las podemos encontrar en nuestro entorno día tras día, solo hay que pensar en el objetivo de los semáforos. El término de ayuda (en inglés "cue") se define como información externa relevante a la ejecución de un movimiento (Horstink et al., 1993). En general, las ayudas se dividen en fuentes de información de distinto tipo tanto espacial como temporal. Las primeras proporcionan información acerca de hacia dónde dirigir el movimiento mientras que las temporales informan acerca de cuándo realizarlo.

A partir de la clasificación anterior, si el objetivo es agarrar un objeto, como por ejemplo la tetera para echar agua en la taza, se hace necesario obtener información sobre el comportamiento de dicho objeto para proporcionar ayudas sobre todo espaciales, para indicar hacia dónde debe dirigirse dicho objeto en caso de que el paciente haga un mal uso del mismo. De ahí que los objetos que el paciente utiliza estén instrumentados con acelerómetro y sensores de fuerza para enviar su posición y la forma en la que están siendo manipulados al procesador central del sistema situado en el propio ordenador "All-In-One".

Además de la importancia de estos monitores sobre su interacción con el paciente para guiarle en la rehabilitación, cabe destacar su relevancia al proporcionar un canal de comunicación con el terapeuta o clínico, el cual dispone también del hardware y software especializado para supervisar remotamente la sesión de rehabilitación y almacenar toda la información que estime oportuna para la evaluación. Más aún, el clínico tiene la posibilidad de personalizar la interfaz correspondiente, programar sesiones, acceder a su historial y evaluar la eficacia del sistema. Toda esta información se recogerá en una base de datos local encriptada conforme a las leyes de privacidad y protección de datos.

Finalmente, es preciso destacar la participación de expertos en el área para asegurar la eficiencia del sistema. Como se puede observar, la implantación del mismo engloba aspectos tanto técnicos como médicos. Por este motivo, se han tenido que llevar a cabo diversos cuestionarios y entrevistas para conocer la opinión de los profesionales en el sector de la rehabilitación doméstica y sobre todo, para tener claros los requisitos que el sistema debe cumplir desde el punto de vista de interacción con el paciente.

Entre la información precisada se puede destacar tanto experiencias pasadas con pacientes que han sufrido este tipo de enfermedades como opiniones personales acerca del impacto que este sistema propuesto tendrá en la vida del paciente y su posible adaptación al sistema de rehabilitación.

Evidentemente, el sistema debe permitir al paciente que se sienta, en la medida de lo posible, igual de tranquilo y con la misma confianza que si estuviera con el terapeuta.

3. Arquitectura general del sistema

Una vez establecidas las bases para la utilización y la instalación del sistema, en esta sección se definen los detalles técnicos de la plataforma, con particular atención a la arquitectura general (ver Fig. 4). El sistema se compone de dos subsistemas principales, un sub-sistema cliente (CCS) y un sub-sistema Servidor (SS).

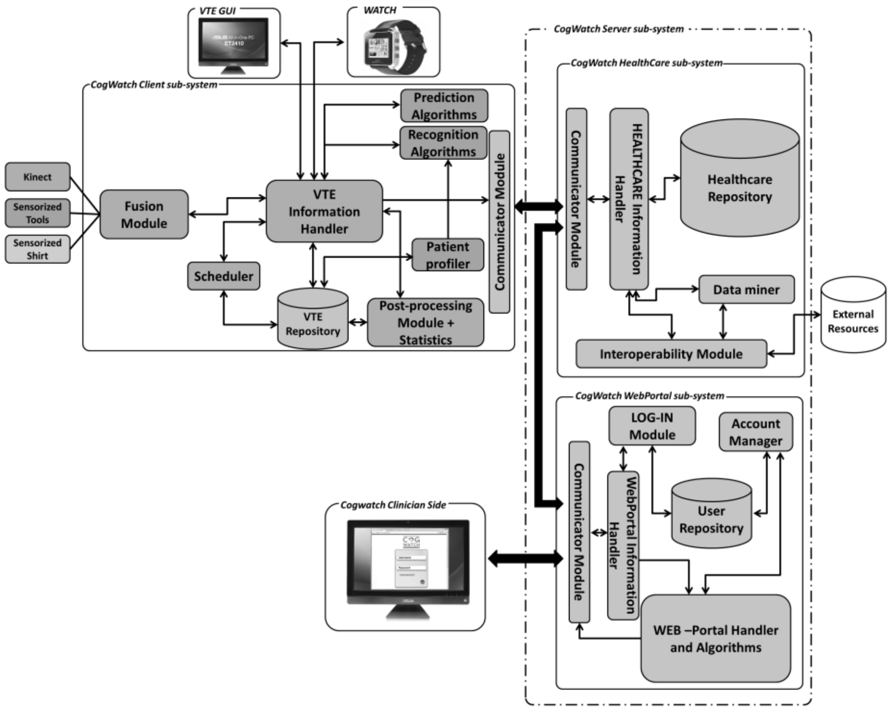


Fig. 4. Arquitectura principal del sistema

3.1. Sub-sistema cliente (CCS)

El CCS, ubicado en la casa del paciente, tiene como principal foco la adquisición de datos durante las sesiones de rehabilitación, bien a través del uso de los objetos sensorizados o bien a través de la interfaz gráfica de usuario. El CCS Gestiona los datos de los pacientes y envía los resultados de las sesiones de rehabilitación al personal sanitario responsable de la evaluación de las sesiones.

Por un lado, como se ha anticipado anteriormente, el CCS está compuesto por una serie de dispositivos que se usan para capturar y analizar el comportamiento y el rendimiento de la actividad de los pacientes. Estos dispositivos se dividen en dos categorías para: *monitorización* y *guiado* de los pacientes durante las sesiones.

Los dispositivos de monitorización incluyen los sistemas basados en la visión y los objetos sensorizados que se utilizarán durante la ejecución de las sesiones de rehabilitación. La Tabla 1 contiene la descripción de los sistemas de monitorización usados.

Tabla 1. Descripción de los sensores

Sensor	Descripción
Microsoft Kinect™	Dispositivo que se encarga de la adquisición de la información relativa a las manos del paciente, el movimiento de las muñecas y graba el video general de la ejecución de la tarea.
Objetos sensorizados	Objetos de uso cotidiano equipados con acelerómetros y sensores de presión usados para capturar la interacción del paciente y recoger datos para el reconocimiento de movimientos.

Los objetos sensorizados utilizados en esta primera versión de la plataforma están relacionados con la tarea de preparación de un té, y, en particular, se enumeran a continuación:

- Tetera.
- Taza.
- Jarra de leche.
- Jarra de agua.

Además, otros objetos no sensorizados por causa de su naturaleza fungible o su reducido tamaño necesarios para realizar dicha tarea son:

- Azúcar.
- Bolsitas de té.
- Cuchara de té.
- Dos cuencos para depositar las bolsas de té (antes y después de su uso)

Los objetos se colocan en una posición inicial específica (ver Fig. 5), con posiciones fijadas a priori con el fin de dar facilidad al paciente a la hora de configurar una nueva sesión. Además, se utilizan posiciones fijas para poder detectar el uso y el movimiento de los objetos a través del sensor Kinect™.

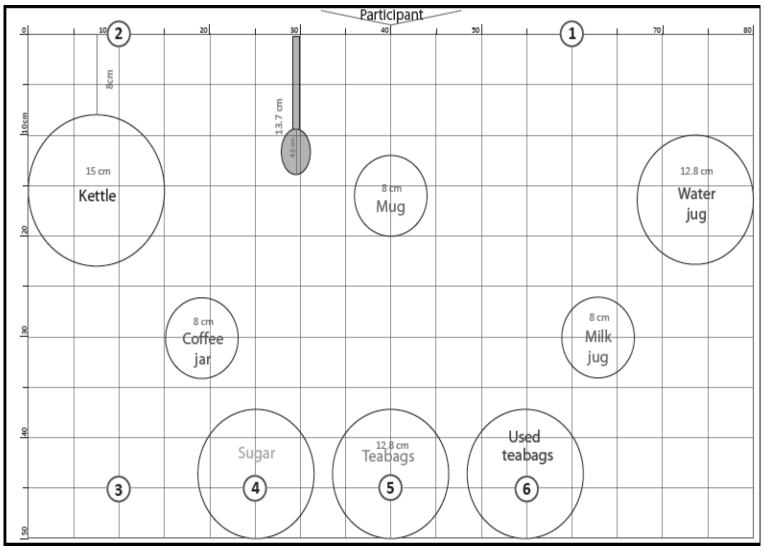


Fig. 5. Panel de configuración de los objetos

Los dispositivos que se usan para guiar a los usuarios son un reloj inteligente con capacidad de vibración y un ordenador "All-In-One" denominado VTE (Virtual Task Execution) usado para la ejecución de las tareas. En la Tabla 2 se puede encontrar una descripción más detallada de estos dispositivos.

Tabla 2. Descripción de los dispositivos de guiado

Dispositivo	Descripción
Reloj inteligente	Este dispositivo tiene la funcionalidad de vibrar en el caso en que se detecte un error por parte del paciente durante la sesión de rehabilitación.
VTE	Es un ordenador del tipo All-in-One con funcionalidades táctiles para proporcionar a los pacientes las ayudas correspondientes para la corrección de los errores cometidos y el guiado durante las sesiones de rehabilitación. Además aloja los algoritmos para recoger, procesar y guardar todos los datos que proceden de la sesión de rehabilitación.

Como se ha descrito en la tabla anterior, el VTE aloja los módulos de software del CCS, encargados de recoger y procesar la señal recibida de los sensores, además de guardar los resultados de la sesión y sus estadísticas en la Base de Datos. La Tabla 3 contiene los detalles de los módulos software del CSS.

Tabla 3. Módulos software del CCS

Módulo	Descripción
Gestor de información	Está a cargo del tratamiento de los datos cuando se detecta un evento, aportando lógicas basadas en prioridades al fin de evitar que el sistema se colapse.
Planificador	Módulo encargado de la gestión de los planes de rehabilitación.
Repositorio VTE	Es la base de datos encargada del archivo de los datos procedentes de las sesiones de rehabilitación ejecutadas.
Algoritmos de predicción e identificación	El sistema debe reconocer de forma automática en qué etapa de la ejecución de la tarea se encuentra el paciente, para estimar si la tarea se ha ejecutado de forma correcta y proponer las consiguientes ayudas en caso de error. Estos algoritmos tienen como entrada los datos del sensor Kinect ™ y de los objetos sensorizados.
Módulo de comunicación	Es el modulo encargado de la comunicación entre la CCS y la SS. Contiene algoritmos de criptografía para garantizar la privacidad de los datos transmitidos y un acceso seguro.

3.2. Sub-sistema Servidor (SS)

El SS se compone del módulo *HealthCare* (HS) y del módulo *Portal Web* (WS). El HS es el módulo remoto encargado de recibir y almacenar los datos de las sesiones de rehabilitación. Está concebido para ser instalado en el centro de rehabilitación y es capaz de gestionar los datos procedentes de los recursos externos. La Tabla 4 contiene los detalles de este módulo.

Tabla 4. Descripción de los módulos del HS

Módulo	Descripción
Módulo de comunicación	Es el módulo responsable de la comunicación con el lado CSS, además de garantizar la privacidad y la seguridad a través de algoritmos de criptografía.
Gestor de información HS	Es el módulo encargado de manejar la información e invocar el módulo apropiado cada vez que se detecta un evento.
Repositorio HS	Es la base de datos encargada de archivar de los datos del paciente, del personal médico además de los resultados de las sesiones de rehabilitación ejecutadas y sus estadísticas.
Módulo de interoperabilidad	Es el módulo responsable de garantizar la interoperabilidad del sistema con sistemas externos y normativas existentes.

El WS es el encargado de mostrar los datos y las estadísticas de las sesiones de rehabilitación al personal clínico autorizado. El módulo es instalado en un único servidor externo, basado en un portal web, y es accesible sólo por usuarios autorizados, en particular personal clínico y personal de administración. En la siguiente tabla se muestra una descripción de los diferentes módulos, (ver Tabla 5).

Tabla 5. Descripción de los módulos de CW

Módulo	Descripción
Gestor de perfiles	Gestiona los perfiles de usuarios
Gestor de identificación	Gestiona los permisos de conexión al sistema remoto
Gestor de Portal Web	Es el módulo encargado de la gestión del intercambio de información entre diferentes sub-módulos y encapsula la lógica del sistema centralizado.

4 Interacción con el usuario

El sistema proporciona al paciente y al personal clínico, unas interfaces simples y atractivas con el fin de facilitar la interacción con el sistema.

La interfaz dedicada al paciente se encuentra en el VTE y su objetivo es proporcionar las ayudas en forma de imágenes, videos y mensajes (de texto y/o verbales) que hacen que los pacientes se den cuenta del error cometido y traten de corregir la acción no ejecutada o ejecutada de forma incorrecta.

En la siguiente figura (ver Fig. 6) se enseña la pantalla principal de la interfaz, la cual se usa para seleccionar la tarea que se quiere ejecutar.

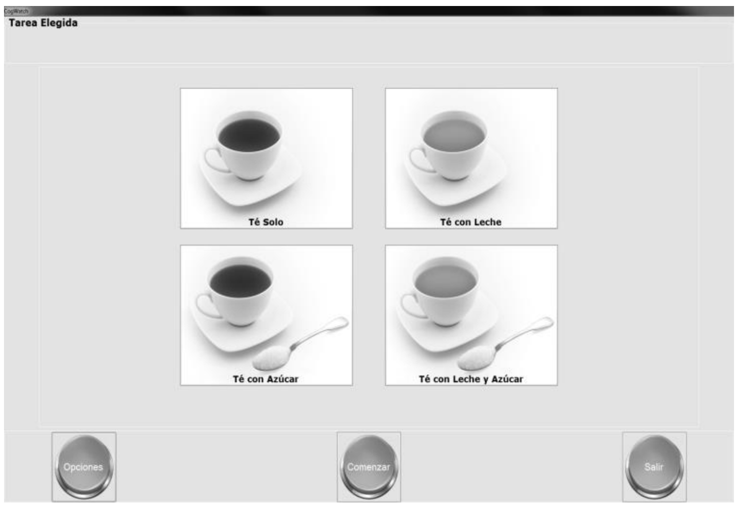
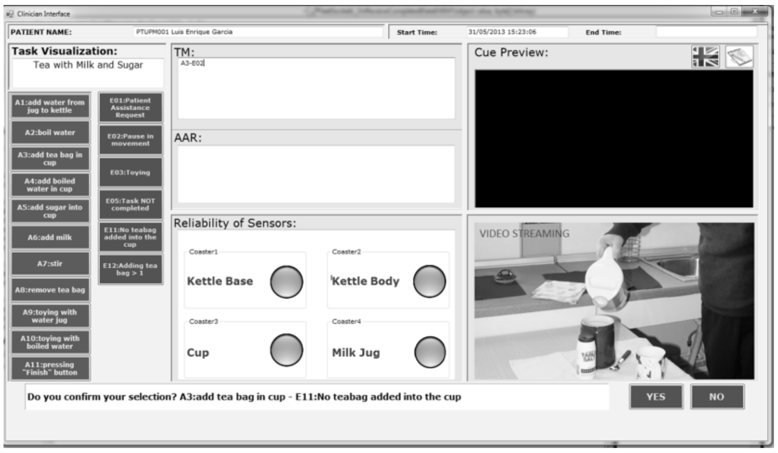


Fig. 6. Pantalla principal de la interfaz de usuario.

Finalmente, la interfaz dedicada al personal clínico permite a los profesionales comprobar y controlar el rendimiento del sistema y del paciente, durante las sesiones de rehabilitación, se instala en un ordenador común y se compone de varias características que hacen que el personal clínico pueda supervisar el sistema y actuar cuando sea necesario (ver Fig. 7).



PATIENT NAME: PTUPM001 Luis Enrique Garcia
Start Time: 31/05/2013 15:23:06
End Time:

Task Visualization: Tea with Milk and Sugar

TM: A3-E02

Do you confirm your selection? A3:add tea bag in cup - E11:No teabag added into the cup

Fig. 7. Pantalla principal de la interfaz del personal clínico.

La interfaz está compuesta por las siguientes sub-ventanas:

- **Visualización de tareas:** enseña la tarea elegida por el paciente en el VTE.
- **Sub-tareas y errores:** lista de las acciones y errores que podrían ocurrir durante la ejecución de la tarea. Aquí el médico puede seleccionar la acción correspondiente una vez que el paciente la haya ejecutado además de elegir el error cometido durante la ejecución; en el caso que la acción y el error sean identificados de forma automática por el VTE, el clínico puede validarla antes de que la ayuda sea visualizada en la interfaz del paciente, es decir, el VTE.
- **Vista previa:** cuadro de comando donde el clínico puede visualizar y validar la ayuda que se enviará al VTE.
- **Fiabilidad de los sensores:** ventana de la interfaz donde se enseña el estado de los sensores.
- **Reproducción de vídeo:** ventana de la interfaz donde se puede ver el video del paciente en directo, durante la ejecución de las sesiones de rehabilitación.

5 Experimentos y estudio de resultados

Tras la descripción del sistema y su arquitectura, pasamos a detallar la evaluación técnica realizada. Esta evaluación técnica del sistema se ha realizado para comprobar la capacidad de interacción con el paciente y la practicidad del sistema durante su uso en el hogar en la ejecución de las tareas cotidianas. Dichas pruebas han sido realizadas en un entorno apropiado, el Living Lab situada en la Escuela Técnica Superior de Telecomunicaciones se ha erigido como un entorno apropiado para la realización de las pruebas.

A continuación se muestra un plano de esta casa inteligente (ver Fig. 8).

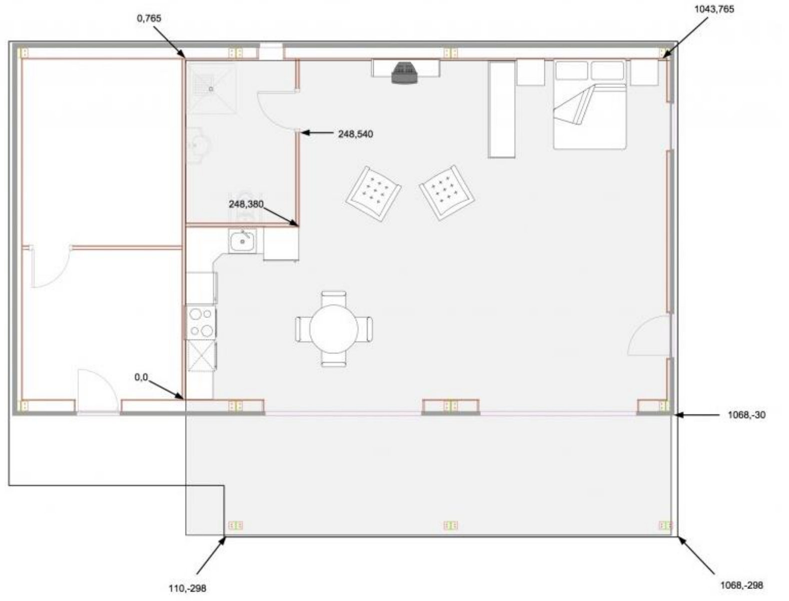


Fig. 8. Plano de la casa inteligente escenario de las pruebas.

Como se puede observar en el propio plano, este Living Lab permite reconstruir la integración del sistema de rehabilitación en diferentes estancias de una casa. Para la evaluación técnica se ha seleccionado la tarea de preparación de un té, explicada en los capítulos anteriores. El escenario "cocina" fue recreado en la propia cocina del Living Lab, permitiendo un entorno propicio para las pruebas. Los distintos voluntarios que se prestaron a realizar la tarea de preparación de un té dispusieron del instrumental y la ambientación requerida para simular el entorno doméstico.

En las pruebas realizadas se ha solicitado a los voluntarios realizar distintos tipos de té: té solo, té con leche, té con azúcar y té con azúcar y leche en diferentes secuencias de acciones y cometiendo errores de forma intencionada a fin de cubrir todas las acciones y tipos de errores mostrados en las Tablas 6 y 7.

Las sub tareas realizadas en base a la secuencialización explicada en el capítulo 2 se muestran a continuación, (ver Tabla 6). Cada voluntario tuvo que realizar una tarea específica, con o sin errores, y presionar el botón finalizar al terminar la tarea.

Tabla 6. Sub tareas de la tarea global de preparación del té.

Tarea	Descripción
A1	Verter Agua en el hervidor/tetera
A2	Calentar el agua
A3	Añadir bolsa de té a la taza
A4	Añadir agua caliente a la taza
A5	Añadir azúcar a la taza
A6	Añadir leche a la taza
A7	Remover
A8	Quitar la bolsa de té
A9	Jugar con jarra de agua
A10	Jugar con agua caliente
A11	Presionar el botón de finalizar

La clasificación de los tipos de errores posibles se muestra a continuación, (ver Tabla 7):

Tabla 7. Clasificación de errores.

Error	Descripción
Adición	Añadir algún ingrediente o realizar una acción no requerida
Omisión	Olvidar realizar una tarea
Perseverancia	La repetición involuntaria de un paso o subtarea
Anticipación	Realizar un paso o sub-tarea anterior en secuencia a lo habitual
Perplejidad	Un retraso o vacilación en la realización de una acción
Juego	Perseverar la intención de tocar o mover un objeto sin realizar una acción concreta

A partir de los resultados obtenidos se pudo comprobar que los errores que se detectaban con mayor frecuencia eran los de “perplejidad” y “juego” no correspondiéndose siempre con la realidad los errores de pausa en movimiento dados por “perplejidad”. Estos errores precisan de una definición de tiempos variable en función de las acciones realizadas, permitiendo así mayor versatilidad al sistema. Este problema, sin embargo, no supuso un gran déficit en la experiencia del usuario dado el sistema de prioridades y la agrupación de ayudas que se realizó de cara a suministrar una realimentación oportuna y coherente al paciente. En base a ello se agruparon los errores, anteriormente clasificados, en tres grupos.

El primer grupo se compone de errores en los que la ayuda proporcionada se compone de un texto en el monitor o mensaje oral para indicar el error y abortar el sistema. Este protocolo se adopta cuando el error podría ser peligroso para el usuario y la tarea no puede ser completada.

El segundo grupo está compuesto por los errores menos peligrosos en los que la información suministrada al usuario se divide en tres etapas:

- En primer lugar, se muestra al usuario una imagen de la acción correcta.
- En caso de no ejecutarla, a continuación se muestra un video real de la misma acción.
- Por último, si la pausa en el movimiento es continua, se proporciona un último mensaje informando acerca del error y el sistema aborta permitiendo al usuario relajarse y tratar de intentarlo de nuevo más tarde.

El tercer y último grupo está compuesto por los errores relacionados con distracciones leves. La información suministrada al usuario también se dividió en tres etapas, pero de una forma diferente de la mencionada anteriormente:

- En primer lugar, una vibración del reloj avisa al usuario.
- Si no es consciente del error, a continuación se muestra un texto o mensaje oral en el monitor.
- Por último, si la pausa en el movimiento es continua, también se proporciona un mensaje final como en el caso anterior informando acerca del error, y el sistema aborta permitiendo al usuario relajarse y tratar de intentarlo de nuevo más tarde.

6 Conclusiones

El sistema presentado está concebido para ser un Sistema de Salud Personal (SSP), capaz de ofrecer una solución sanitaria de bajo coste, intuitiva y práctica orientada a la rehabilitación de pacientes que sufren Apraxia y ADS.

En este artículo se han presentado los progresos en la realización de un sistema de rehabilitación cognitiva que supone un avance en el área de la rehabilitación. La situación actual de los pacientes que experimentan algún tipo de déficit cognitivo, tras un episodio de accidente cerebrovascular, suponen un alto coste respecto al sistema propuesto. Además, el sistema propuesto aporta una alternativa al tratamiento tradicional, que requiere la asistencia continuada por parte de un terapeuta ocupacional.

Tras describir brevemente la problemática actual de los pacientes que sufren esta enfermedad se ha abordado la arquitectura y funcionalidad de la propuesta realizada. La evaluación técnica ha resultado positiva y considera el proyecto apto para la realización de pruebas con pacientes.

Esta propuesta, como se ha tratado en secciones anteriores, permite la rehabilitación cognitiva y la asistencia del paciente de forma remota. Los resultados de la evaluación clínica podrían suponer un avance en la interacción persona máquina puesto que se evaluará la eficacia de los diferentes dispositivos de realimentación que se suministrarán al paciente.

Las pruebas de evaluación técnica encontraron problemas de definición de tiempos para la realización de las actividades. Este problema se manifestó únicamente en casos aislados, y supuso aportar ayudas innecesarias al paciente. Aunque el problema pudiese resultar arbitrario e irrelevante, no permite el proceso de aprendizaje pretendido en la rehabilitación. Sin embargo, al prevalecer el sistema de prioridades, se garantiza que el usuario finalice siempre la sesión bajo condiciones de seguridad. A partir de los resultados obtenidos se realizó una redefinición de los temporizadores internos que permite su configuración. Esto resultó ser una solución para el problema descrito.

En futuros prototipos se plantea mejorar las capacidades de interacción de los dispositivos con el paciente para mejorar su experiencia de usuario. Por ejemplo, enviando mensajes a través de relojes inteligentes con interfaces táctiles.

Agradecimientos

Este trabajo ha sido financiado por la Comisión Europea en el marco de la concesión FP7-ICT-2011-288912.

Los autores quieren reconocer especialmente el trabajo realizado por los colaboradores del proyecto CogWatch, que trabajan en diferentes áreas pero no contribuyen directamente en la redacción del presente documento.

Referencias

Cooper, R., & Shallice, T. , 2000.Contention scheduling and the control of routine activities, Cognitive Neuropsychology, 17(4), 297-338.

Díaz-Guzmán J, Egido JA, Gabriel-Sánchez R, Barberá-Comes G, Fuentes-Gimeno B, Fernández-Pérez C. 2012. Stroke and Transient Ischemic Attack Incidence Rate in Spain: The Iberictus Study. Cerebrovascular Diseases 34(4):272-281.

Gamberini, L., Riva, G., Spagnolli. A., Raya, M. A., Zapata, C. B., Baños, R., Psychnology Journal, 2002.

Hermsdörfer, J., Blankenfeld, H., Goldenberg, G., 2003. The dependence of ipsilesional aiming deficits on task demands, lesioned hemisphere, and apraxia. Neuropsychologia 41, 1628–1643.

Horstink, M., De Swart, B., Wolters, E., & Berger, H., 1993. Paradoxical behavior in Parkinson's disease, E. C Wolters, & P. Scheltens (Eds). Mental dysfunction in Parkinson's disease, proceedings of the European congress on mental dysfunction in Parkinson's disease. Amsterdam: Vrije Universiteit.

Liepmann, H., Drei Aufsätze aus dem Apraxiegebiet, Berlin: Karger, 1908.

MacKay, J., Mensah, G.A., 2004. The atlas of heart disease and stroke. World Health Organization.

Morady, K., Humphreys, G.W., 2009. Comparing action disorganization syndrome and dual-task load on normal performance in everyday action tasks. Neurocase 15, 1–12.

Rothi, L.J.G., Heilman, K.M., 1997. Apraxia: the neuropsychology of action. Psychology Press.

Scopelliti, M., Giuliani, M.V., Fornara, F., 2005."Robots in a domestic setting: a psychological approach", Universal Access in the Information Society 4(2):146-155.

World Stroke Organization - Home [Documento WWW], n.d. URL http://www.world-stroke.org/ (ultimo acceso 6.5.13).

ÍNDICE ALFABÉTICO DE AUTORES