2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Development of a multimode navigation system for an assistive robotics project

A. Cherubini, G. Oriolo, F. Macrì, F. Aloise, F. Babiloni, F. Cincotti, D. Mattia

Abstract— **Assistive technology is an emerging area where robotic devices can be used to strengthen the residual abilities of individuals with motor disabilities or to help them achieve independence in the activities of daily living. This paper deals with a project aimed at designing a system that provides remote control of home-installed appliances, including the Sony AIBO, a commercial mobile robot. The development of the project is described by focusing on the design of the robot navigation system. Single step, semi-autonomous and autonomous operating modes have been realized to provide different levels of interaction with AIBO. Automatic collision avoidance is integrated in all cases. The performance of the navigation system is shown by experiments. Moreover, the system underwent clinical validation, in order to obtain a definitive assessment through patient feedback.**

I. Introduction

The development of service robots for healthcare and assistance to elderly or disabled persons is an active area of research and development. Several projects have been undertaken in this field, ranging from navigation aids for the visually impaired [1] to robots for assisting individuals with motor disabilities [2]. Many assistive robotics projects aim at increasing the quality of the user life in his/her house.

In this field, it is important to design versatile systems, which can assist the patient in certain tasks, e.g., managing home appliances, carrying objects, or monitoring the environment [3]. These systems should fulfill basic requirements with respect to safety, cost and user friendliness. In particular, it should be possible to collect signals for controlling devices or robots in an 'intelligent home' from different sources depending on the patient residual abilities. Recently, electroencephalographic brain signals [4], and implanted Brain Computer Interfaces [5] have been used. The systems should be validated by experiments on potential users [6]. Another typical use of robotic technologies in this context is directed to partial recovery of the patient mobility. Semi-autonomous navigation systems for wheelchairs [7] are an example. These systems adapt to various levels of disability by offering different autonomy levels [8], [9].

This work has been partially supported by the Italian Telethon Foundation Grant GUP03562.

A. Cherubini, G. Oriolo and F. Macrì are with the Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza", Via Eudossiana 18, 00184 Roma, Italy. {cherubini,oriolo}@dis.uniroma1.it

F. Aloise, F. Cincotti and D. Mattia are with the Fondazione Santa Lucia IRCCS, Via Ardeatina 306, 00179 Roma, Italy. {f.aloise,f.cincotti,d.mattia}@hsantalucia.it

F. Babiloni is with the Dipartimento di Fisiologia Umana e Farmacologia, Università di Roma "La Sapienza", Piazzale Aldo Moro 5, 00185 Roma, Italy. fabio.babiloni@uniroma1.it

In this paper, we present the basic algorithms developed for a robot navigation system and their application in the ASPICE (Assistive System for Patient's Increase of Communication, ambient control and mobility in absence of muscular Effort) project [10]. One central feature of the ASPICE system is the possibility for the user to remotely control the motion of a mobile robot (a Sony AIBO) with a reduced set of commands. Depending on the residual abilities of the user, as well as on the desired task, it is possible to choose between different operating modes. Automatic obstacle avoidance is integrated in the system to guarantee safe, collision-free motion in cluttered environments.

The paper is organized as follows. In Sect. II the architecture of the ASPICE system is briefly illustrated. In Sect. III, the main features of the AIBO robot are described. Section IV presents the primitives developed for the robot framework, at perception and motion levels. The robot navigation modes that we implemented, on the basis of the ASPICE requirements, are outlined in Sect. V. Experiments are reported in Sect. VI. Other issues not covered in the previous sections are mentioned in the conclusion.

II. The ASPICE project

The ASPICE project received in 2004 a renewable two-year funding grant from TELETHON, an italian medical research charity foundation. The project involves three partners, among which the Clinical Neurophysiopathology Laboratory of the Fondazione Santa Lucia IRCCS and the Robotics Lab of the University of Rome "La Sapienza".

ASPICE is aimed at developing a technological aid allowing neuromotor-disabled users to improve or recover their mobility and communication in the surrounding environment. The project is addressed towards those patients in which the residual muscular strength is low and practical obstacles or security concerns do not allow a displacement from bed [10]. Hence, the major requirements are: adaptability to different levels of disability, low cost, and robustness to the setting. Depending on the user requirements, the assistive device will be a program running on a common low-power PC, on a palmtop, or on a powerful workstation.

The ASPICE architecture, with input and output devices, is shown in Fig. 1. Some key elements of the system are:

- various input devices for easy access to the Control Unit: standard input devices (mouse, eye tracker, voice recognition) and a Brain-Computer Interface (BCI);
- the Control Unit, which receives signals from the input devices via a Graphic User Interface (GUI) and converts

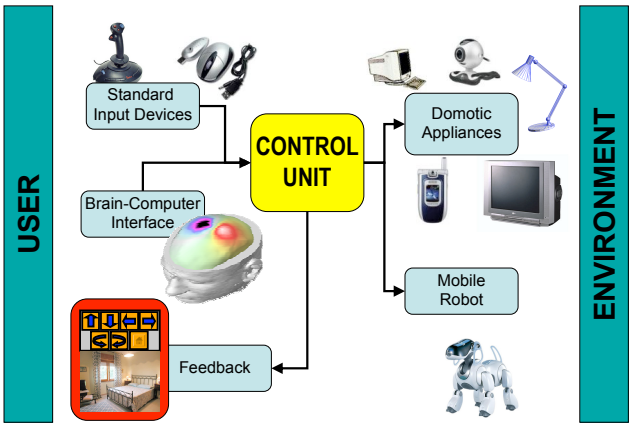

Fig. 1. The ASPICE architecture.

them in commands that drive the output devices (the domotic appliances or a mobile robot);
- the mobile robot;
- a number of domotic appliances, which must comply with the patient's need for ambient control: TV, fan, lights, video camera, telephone, personal computer;
- visual feedback (either through the fixed video camera or through the robot vision system) to provide the user with an increased sense of presence in the environment.

The Control Unit contains drivers for all output devices; in some cases, existing drivers are utilized, whereas in other cases (e.g., the mobile robot) the driver has been designed "ad hoc" for the system. The BCI detects the activation patterns of the brain, and whenever the user induces a voluntary modification of these patterns, it is able to translate it into an action associated to the user will. The BCI used in ASPICE is based on variations of the EEG rhythmic activity; the signals are captured by means of an electrode cap, and processed by a dedicated software package. All signals between the input devices and the Control Unit, and between the latter and the output devices (including visual feedback) are transmitted over a wireless connection.

In assistive robotics projects, a major requirement is the user friendliness of the robotic platform. Although in recent years users are becoming more acquainted with technology, characteristics such as low cost, safety, and low request for maintenance are still fundamental needs of any biomedical robotic application. In particular, clinicians have often emphasized the importance of working with a friendly-looking robot, in order to limit the psychological impact on patients. In our case, these considerations led to the choice of the dog-like robot Sony AIBO ERS-7 for inclusion in the system. Besides, studies on improvement of quality of life, among elderly, using AIBO, have given good results [11].

AIBO should be driven around the user home with a small set of commands. It should also assist the patient in visually monitoring the environment and communicating with the caregiver. Partial autonomy should be implemented in order to avoid collisions with obstacles and AIBO should be able to charge its battery when needed without user intervention.

As aforementioned, an objective of the ASPICE project

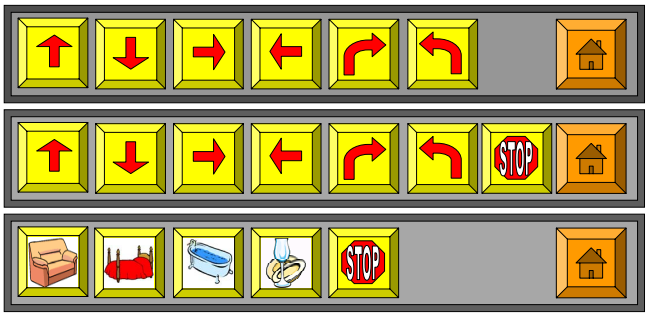

Fig. 2. ASPICE navigation GUIs: single step (top), semi-autonomous (center) and autonomous (bottom) modes. In each GUI, the home button brings back to the ASPICE main GUI.

is compatibility with a variety of users and their level of disability. In this spirit, three navigation modes have been developed: *Single step*, *Semi-autonomous* and *Autonomous* mode. The user is expected to choose single step navigation to retain complete control of the robot; e.g., for fine motion in cluttered areas. In semi-autonomous navigation, the user specifies the main direction of motion, leaving to the robot the task of avoiding obstacles. Finally, in the autonomous mode, a target point in the environment is assigned by the user, and the robot travels to the target; this is useful for quickly reaching important locations. This mode is expected to be particularly useful for severely impaired patients, which are unable to send frequent commands. All three navigation modes must contain some level of obstacle avoidance.

Each navigation mode is associated to a GUI in the ASPICE Control Unit. The GUIs are shown in Fig. 2. By selecting the corresponding button from the single step GUI, the user can control the step direction. From the semi-autonomous mode GUI, the user can select one of six directions – the same of single step mode – or stop the robot. Instead, in the autonomous navigation GUI, each button corresponds to a destination in the user apartment (here, the living room, the bedroom, the bathroom, and the kitchen).

III. The Robot Platform: AIBO

The platform used in this work is a quadruped robot, Sony AIBO ERS-7 (see Fig. 3). AIBO is a low-cost robot, widely used for research purposes. The robot is equipped with 20 actuated joints, a CMOS camera, two distance sensors (on the head and on the chest), an accelerometer, a stereo microphone, a MIDI speaker, a set of leds and pression sensors. A wireless card enables remote control.

AIBO's real-time operating system APERIOS runs a specialized layer called OPEN-R, a cross-development environment based on C++. The robot behavior is programmed by loading all executable and configuration files on a memory stick which is read by the on-board processor. In spite of the above features, the AIBO robot presents many limitations. The most severe are the following:

- the closed hardware prevents the addition of sensors and/or actuators;
- since Sony does not release the driver code, we had to realize from scratch an *ad hoc* driver for this work;

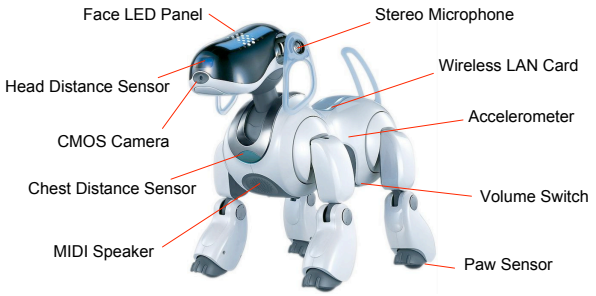

Fig. 3. The Sony AIBO ERS-7 used in the ASPICE Project.

- the head distance sensor and the CMOS camera move in accordance, making it impossible for the distance sensor to detect obstacles in directions other than the one pointed by the camera; a tradeoff between moving the head for video feedback and moving it for obstacle detection/avoidance had to be reached;
- the chest sensor is constrained to the robot body and peculiarly oriented, thus limiting its effective utility;
- vibrational and slipping effects during the quadruped gait cycle make odometric reconstruction very inaccurate in the long run;
- the variable attitude of AIBO during its gait precludes the use of an external sensory system (e.g., based on infrared triangulation with a detector placed on the robot) for solving the localization problem.

IV. Primitives

In order to utilize AIBO, specific primitives have been developed and integrated in the driver framework. These have been designed to fulfill the robot driver requirements: obstacle detection/avoidance, motion control, and path planning.

Let us define the reference frames which will be used:

- the *robot frame* (Fig. 4) with origin fixed at the robot center projection on the ground, x axis in the forward direction, y axis pointing the left side of the robot, and z axis in the vertical direction;
- the *image frame* (Fig. 5) with origin fixed at the top left corner of the image, horizontal ${}^{i}x$ axis pointing right, and ${}^{i}y$ axis pointing downward;

A. Perception primitives

The main features that the robot should perceive are the obstacles that it should avoid and the landmarks that it needs for localization and path planning purposes. We chose to use the robot range sensors to detect obstacles, and the camera to recognize visual landmarks. We use a local two-dimensional occupancy grid (OG) to represent the detected obstacles, built by the *occupancy grid generator*. The visual landmarks that we use are straight white lines (SWL) and coded squares (CS) placed on the floor. Thus, a *straight white line extractor* (SWLE) and a *coded square extractor* (CSE) have been developed. Moreover, the visual landmarks (VL) should be located in sequential scenes. This task is accomplished by a *visual landmark tracker* (VLT).

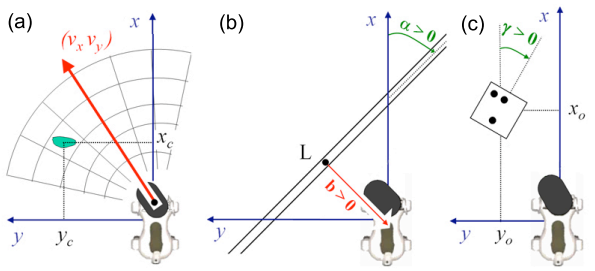

Fig. 4. Relevant variables utilized in: (a) occupancy grid generation, (b) straight white line tracking, and (c) coded square tracking.

1) Occupancy grid generator: The robot should be able to recover robust and useful spatial descriptions of its surrounding obstacles, using sensory information. These descriptions should be used for short-term planning in the environment. To do this, we used a tesselated two-dimensional representation of spatial information called the occupancy grid, which in the past years proved to be extremely efficient for performing path planning and obstacle avoidance in unknown and unstructured environments [12].

In our approach, the two range finders (head and chest) are used to detect obstacles, although the chest sensor, due to its limited range and particular orientation (see Fig. 3), can only detect close obstacles and therefore is not used to compute the OG. Thus, only the head sensor is utilized to build the local OG by moving the head pan joint along a sinusoidal profile spanning an angular width of 90°. While the origin of the occupancy grid is always on the head pan axis, and its longitudinal extent is limited by the range of the head distance sensor (1 m), its orientation (i.e., the direction of its bisectrix) is the same as the direction of motion $(v_x\ v_y)$ (see Fig. 4a). Obviously, due to the joint limit, it is impossible to build occupancy grids for backward motions. The grid may be built with the robot either stationary or in motion. In the second case, the head pan movement is synchronized with the gait cycle, and odometric data (reconstructed through the leg joint encoders) are used to build a consistent map. When the pan cycle is complete, a cell in the grid is considered to be occupied if there is a sensor reading indicating an obstacle inside that cell.

2) Straight white line extractor: A requirement of the robot driver is straight white line extraction. In order to be independent from color classification, the straight white lines are detected by search only on the luminance signal $I\left({}^{i}x,\ {}^{i}y\right)$. Line edges are searched among pixels with a strong gradient of luminance ∇I, as explained in [13].

The SWLE returns the coordinates of the n_j pixels belonging to each straight white line SWL_j among the N_{SWL} lines extracted on the image frame (Fig. 5):

$$[{}^{i}x_r\ {}^{i}y_r]^T_{SWL,j} \qquad r = 1, \ldots, n_j \qquad j = 1, \ldots, N_{SWL}$$

3) Coded square extractor: Along with the SWL, we have used as visual landmarks a set of white coded squares laid on the ground. The identity and orientation of each square is uniquely identified through a black dots code. We arranged from 1 to 7 black dots on the border of the squares, in order

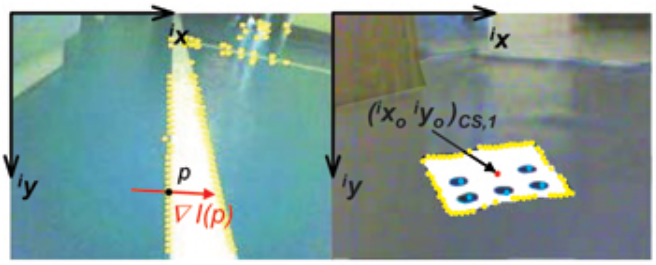

Fig. 5. Extracting edges (yellow), for SWL (left) and CS (right) detection. The detected CS center is marked in red, and dot centers in cyan.

to generate 15 configurations (see Fig. 6) which uniquely define the landmark identity (defined by its label: ID) and orientation. Edges of squares are searched as in the SWLE. More details on the CSE algorithm are given in [13]. The CSE returns, for each of the N_{CS} detected coded squares: the image coordinates of the square center o and of the centers of the n_{dots} black dots (Fig. 5):

$$[{}^i x_o \ {}^i y_o]^T_{CS,l} \qquad [{}^i x_m \ {}^i y_m]^T_{CS,l}$$
$$l = 1, \ldots, N_{CS} \qquad m = 1, \ldots, n_{dots} \qquad n_{dots} = 1, \ldots, 7$$

4) Visual landmark tracker: The SWLE and CSE only take into account information from the current image, and give no long-term knowledge. Thus, consistent landmarks must be obtained by comparing the extracted landmarks over consecutive images. This is done by projecting the characteristic points of each extracted VL from the image frame $[{}^i x \ {}^i y]^T_{VL}$ to the robot frame $[x \ y \ z]^T_{VL}$. Such mapping is not one-to-one, and can only determine, given the camera intrinsic parameters, the projecting ray corresponding to each point $[{}^i x \ {}^i y]^T_{VL}$ [14]. However, in our application, since all the VLs are on the ground plane, the problem can be solved in closed form, as in [15]. The VLT algorithm returns the characteristics of the VLs (shown in Fig. 4b and Fig. 4c), validated in a sufficient number of consecutive frames:

$$[b \ \alpha]^T_{SWL,j} \qquad\qquad\qquad j = 1, \ldots, N_{SWL}$$
$$ID_l \qquad \gamma_l \qquad [x_o \ y_o \ 0]^T_{CS,l} \qquad l = 1, \ldots, N_{CS}$$

B. Motion primitives

From a kinematic viewpoint, AIBO can be considered an omnidirectional robot: three velocities (forward v_x, lateral v_y, and angular v_θ around the robot center, positive for CCW rotation) can be independently specified (Fig. 4a). Whenever the robot velocities are specified in workspace coordinates as $V = [V_x \ V_y]^T$ (e.g., when they are imposed by a user command), they must be mapped to the configuration space. To perform this conversion, we have used two strategies. The first (*omnidirectional translational motion*), consists of simply setting $[v_x \ v_y \ v_\theta]^T = [V_x \ V_y \ 0]^T$. Instead, the second (*nonholonomic-like motion*), consists in setting:

$$\begin{aligned} v_x &= V_x \\ v_y &= 0 \\ v_\theta &= \text{ATAN2}(V_y, V_x) \end{aligned} \tag{1}$$

The advantages of each strategy will be illustrated later.

Basic motion primitives for controlling the robot legs in order to obtain the desired motion $[v_x \ v_y \ v_\theta]^T$ are based on the quadruped parameterized walk [16], which is widely used in the four-legged robot community, and which we will not discuss in detail. Velocity commands computed by the motion primitives are scaled if any of them exceeds the physical limits of the actuators. We also developed two vision-based motion primitives: the *landmark approacher* (LA), and the *straight line follower* (SLF), which use visual information returned by the perception primitives to guide the robot. In practice, the robot actuators are driven by a *visual servoing* scheme. Since the VLT returns the position of the visual landmarks relative to the robot, *position-based* visual servo control turns out to offer a better solution than *image-based* servoing. The two *vision-based* motion primitives are explained below.

1) Landmark approacher: When the robot finds a landmark, it should approach it, in order to get a better perception, which can be useful for localization purposes. It is a *posture stabilization* task with reference configuration defined by the position and orientation of the landmark. Since no smooth time-invariant feedback can solve this problem for a nonholonomic mobile robot [17], and path minimization on the other hand is desired, omnidirectional motion is used. The walk that drives the robot implements a proportional closed-loop control strategy for reducing the robot relative distance and orientation with respect to the landmark.

This is done by setting:

$$\begin{aligned} v_x &= \kappa_\text{T} \ x_{VL} \\ v_y &= \kappa_\text{T} \ y_{VL} \\ v_\theta &= \kappa_\text{R} \ \theta_{VL} \end{aligned}$$

For SWL approaching, $[x \ y \ \theta]^T_{VL} = [b \sin\alpha \ \ b\cos\alpha \ \ -\alpha]^T$. Similarly, for CS approaching, $[x \ y \ \theta]^T_{VL} = [x_o \ y_o \ -\gamma]^T$. In both cases, κ_T and κ_R are positive given gains.

2) Straight line follower: This primitive should solve the *path following* problem for the SWLs. We adopted a nonholonomic model for the robot, in order to obtain a more "natural-looking" walk. Both linear and non-linear smooth state-feedback control solutions are possible. AIBO is modeled here as a unicycle robot, with velocities $[v_x \ \ v_y{=}0 \ \ v_\theta]^T$, and the task can be achieved by using only one control variable, namely v_θ. Linear feedback control can be realized by tangent linearization of $\dot{b}$ and $\dot{\alpha}$, in the neighborhood of $(b = 0, \alpha = 0)$. This gives a second order linear system which is controllable, and thus asymptotically stabilizable by linear feedback on v_ϑ, if $v_x = v_f > 0$ is fixed. A stabilizing linear feedback is of the form:

$$v_\vartheta = (k_2 b - k_3 \alpha)\, v_f$$

with an appropriate choice of positive gains k_2 and k_3.

V. Robot Navigation Modes

All three navigation modes are based on the algorithms presented in Sect. IV. The Single step mode and the Semi-autonomous mode utilize only the occupancy grid generator. The first sequentially uses the OG and implements motion

control, whereas, in the latter, OG generation and motion control are executed simultaneously. The Autonomous mode utilizes *all* the primitives presented in Sect. IV.

A. Single step mode

With single step motion, the robot can be driven, with a fixed step size, in any six directions (forward, backward, lateral left/right, CW and CCW rotations). Before performing the motion command, the robot generates the appropriate OG (oriented along the intention of motion) from its stationary position and verifies whether the step can be performed without colliding with obstacles. Depending on the result of the collision check, the robot decides whether or not to step in the desired direction. Note that, since no OG can be built for backward or rotational motions, the corresponding step commands should be used with care.

B. Semi-autonomous mode

With semi-autonomous motion, the user specifies general directions of motion, which the robot should track as closely as possible. Instead of executing a single step, the robot walks continuously in the specified direction until it receives a new command (either a new direction or a stop). If the specified direction is forward or lateral[1] (i.e., the user desired direction of motion in the workspace is $V_{des} = [V_{des,x}\ 0]^T$ or $V_{des} = [0\ V_{des,y}]^T$), autonomous obstacle avoidance is obtained by the use of potential fields. The algorithm used in this case is explained below.

The OG is generated as the robot moves, and then used to compute the robot velocities. Our algorithm uses vortex and repulsive fields to build the velocity field. For each occupied cell on the OG, $c = [x_c\ y_c]^T$, with x_c and y_c cell coordinates in the robot frame (see Fig. 4a), define the repulsive potential [18] as:

$$U_r\left(\|c\|,\eta\right) = \begin{cases} K_r\left(\dfrac{1}{\|c\|} - \dfrac{1}{\eta}\right)^2 & \text{if } \|c\| \le \eta \\ 0 & \text{else} \end{cases}$$

where $\|c\|$ is the distance of the cell from the robot center, η the radius of influence of U_r, and K_r a given gain. Repulsive and vortex [19] fields induced by each cell are obtained respectively as gradient and rotor of the potential:

$$f_c^r = \begin{pmatrix} \dfrac{\partial U_r(\|c\|,\eta_r)}{\partial x_c} \\ \dfrac{\partial U_r(\|c\|,\eta_r)}{\partial y_c} \end{pmatrix} \qquad f_c^v = \begin{pmatrix} \pm\dfrac{\partial U_r(\|c\|,\eta_v)}{\partial y_c} \\ \mp\dfrac{\partial U_r(\|c\|,\eta_v)}{\partial x_c} \end{pmatrix}$$

Note the different radii of influence η_r and η_v of repulsive and vortex fields. By choosing $\eta_v > \eta_r$, we obtain velocity fields that are essentially vortices at large distances, and become increasingly repulsive at close range. The signs of $f^v_{c,x}$ and $f^v_{c,y}$ depend on the position of the occupied cell with respect to the robot sagittal plane: a cell in the right (left) half of the grid will induce a CW (CCW) vortex.

[1]As for the single step mode, no obstacle avoidance can be performed when executing backward or rotational motions.

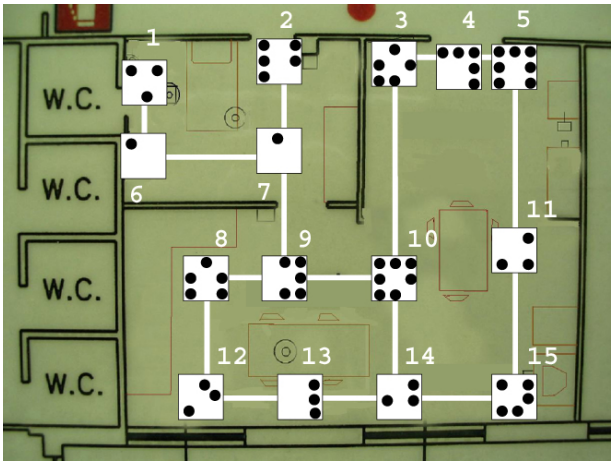

Fig. 6. The roadmap used in autonomous navigation mode. The ID labels of each CS are indicated. Note that crossings appear larger than they are.

The fields generated by all the occupied grid cells are then superimposed with the desired workspace velocity in order to obtain the total velocity field:

$$V = \sum_c f_c^r + \sum_c f_c^v + V_{des}$$

This velocity must be mapped to the configuration space velocities either with the omnidirectional translational motion conversion or by enforcing nonholonomic-like motion (1). The first is consistent with the objective of maintaining as much as possible the robot orientation specified by the user. Instead, with the second kind of conversion, the orientation of the robot is always tangent to the path; the grid provides more effective collision avoidance since the direction of its angle bisector coincides with the x axis (because v_y is null).

C. Autonomous mode

For the autonomous navigation mode (ANM), we designed a physical roadmap (Fig. 6) to reach and connect all relevant destinations in the experimental arena, and utilized a more sophisticated visual servoing scheme. The roadmap is formed by streets and crossings, all realized in white adesive tape laid on the ground. The perception primitives are used to identify streets (straight white lines) and crossings (coded squares) while the motion primitives are used to drive the robot. When approaching a landmark or following a street, the robot concurrently implements *landmark fixation*, in order to keep the landmark centered in the image plane. This is done by solving the inverse kinematics problem for the head joints.

The autonomous behavior is represented by a Petri Nets framework. The ANM Plan uses the following actions (note that at all times during the actions, the perception primitives are executed for searching and updating perceived data):

- *Seek streets*: The robot seeks streets in the environment, while avoiding collisions. Motion directions are predefined: AIBO alternates forward and rotation steps.
- *Approach the nearest street*: When it perceives some streets with the SWLE, and tracks them with the VLT, the robot uses the LA to walk towards the nearest.

- *Follow the street*: When the robot is sufficiently close to the street, it implements the linear SLF for walking on the street, until at least one crossing is detected.
- *Plan the path to destination*: When a crossing is detected with the CSE, and tracked with the VLT, the robot has univocally identified its ID and orientation. This information, along with the CS positions in the robot frame, and with the map, identifies the robot pose (position and orientation). The robot then uses a Dijkstra-based graph search [20] to find the shortest path to the destination. Depending on the result of the graph search, the robot will approach and follow another street (repeat the corresponding actions in the plan), or stop if the crossing corresponds to the desired destination.

The ANM Plan repeats the above actions until the destination is reached. Transitions that start or terminate the actions represent events (e.g., *Street seen*, or *Crossing near*) which are triggered by conditions on sensed information (e.g., distance from a line). The plan must also deal with action failures. For instance, whenever the robot loses visual contact with a street it was approaching, the system aborts the current action and moves to the state where the street is not seen, and so on, until the robot reaches the street again.

VI. Experiments

Here, we show the results of two experiments performed with the robot driver: the first is a comparison between the navigation modes, while the second is autonomous battery charging. We also discuss the ASPICE clinical validation and its results on the quality of life of users.

In the first experiment (Fig. 7), the user is expected to drive the robot from a start to a goal point (noted respectively "S" and "G" on the image). The task is repeated 5 times for each of the three navigation modes (single step, semi-autonomous, and autonomous) and results are averaged. The robot is driven by selecting commands on the ASPICE GUIs; a mouse is used as input device. In semi-autonomous navigation, omnidirectional translational motion is used for mapping desired user velocities to the configuration space. Comparison between the modes is based on execution time and user intervention (i.e., number of times the user had to intervene by clicking on the GUI for updating the commands). As expected, results (see Table I) confirm the qualitative properties expected for each mode. Note how the best choice depends not only on user's preference and ability but also on the specific task (e.g., position of the start and goal points in the environment, presence and position of obstacles).

In a second experiment, autonomous battery charging is tested. This behavior is also present in the Sony Driver, but since Sony does not release the code of its driver, we had to realize it *ad hoc* for this project. This experiment not only fulfills an ASPICE requirement, but also provides an additional testbed for the ANM. In fact, the AIBO Charging Station is placed near a marked crossing on the roadmap, and as soon as the battery level is low, the robot autonomously moves to the station. The experiment is illustrated in Fig. 8. The robot position at consecutive time frames is shown while

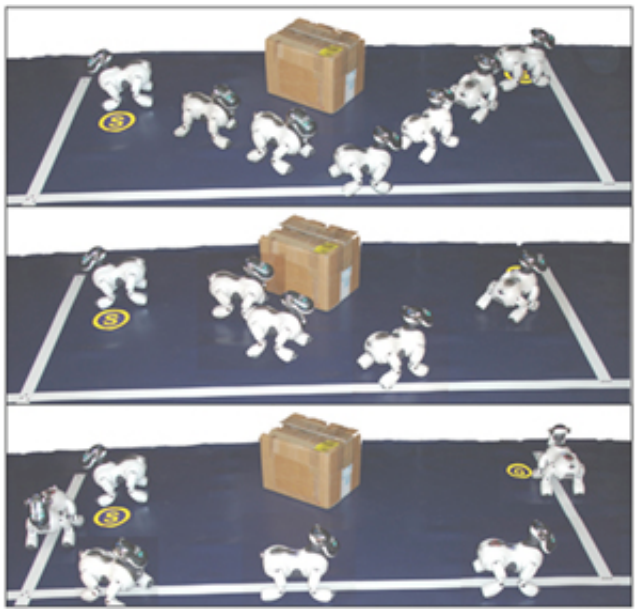

Fig. 7. Experiment with the ASPICE navigation modes: the user drives the robot from point S to point G using the single step (above), semi-autonomous (center) and autonomous (below) modes.

	execution time (sec)	user intervention (clicks)
single step	107	11
semi-autonomous	83	5
autonomous	90	1

TABLE I

Comparison between the three ASPICE navigation modes.

it approaches the roadmap, follows the street up to the battery charger crossing, detects it, and makes a turn in order to reach the charging station on the basis of the plan.

Other experiments, as well as simulations with Webots (a mobile robot simulation environment developed by Cyberbotics) were performed in the Robotics Lab of the University of Rome to test the navigation system, before starting the clinical validation on actual patients. The system was also tested by using the BCI (Brain Computer Interface) to drive the robot. The two experiments in Fig. 7 and 8 as well as a BCI-driven experiment are shown in the video clip attachment to this paper. Other video clips are available at: http://www.dis.uniroma1.it/∼labrob/research/ASPICE.html.

At this stage of the project, the system has been implemented and is available at the Fondazione Santa Lucia in Rome for validation with patients [21]. All domotic appliances used in ASPICE have been installed in the experimental apartment of the hospital. A portable computer runs the Control Unit program, and several input devices are available to cope with as many categories of users as possible. The subjects have been admitted to a neurorehabilitation program, and the whole procedure underwent the approval of the local ethical committee. Then, for two weekly sessions over 4 weeks, the patient and her/his caregivers have been practising with the ASPICE system. The experiments have been carried out with eight subjects suffering from Spinal Muscular Atrophy type II and six subjects suffering from Duchenne Muscular Dystrophy. Data on user satisfaction,

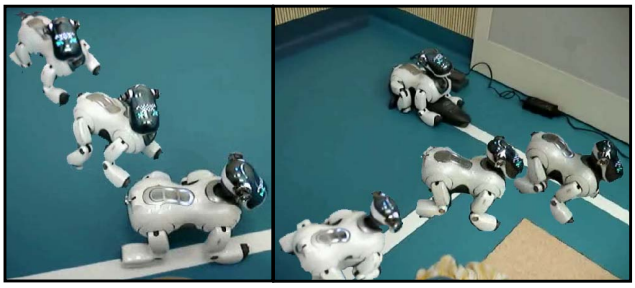

Fig. 8. Two phases of the battery charging experiment.

increase of independence, and reduction of caregiver workload, have been collected, structured in questionnaires. All of the patients were able to master the system and control AIBO within 5 sessions. Four of them interacted with the system via BCI. Most patients experienced (as they reported in the questionnaire) 'the possibility to interact with the environment by myself'. The average grade given by the patients to their 'personal satisfaction in utilizing the robot' was 3.04 on a 5-point scale. This is a very promising result, considering that the users were originally more accustomed to using and controlling the 'traditional' ASPICE appliances rather than the mobile robot.

VII. Conclusions and future work

The ASPICE project is a work in progress. After 24 months, the objective of developing a multimode navigation system for AIBO based on few simple primitives has been achieved. In particular, three navigation modes have been devised to provide the user with different levels of interaction and control of the mobile robot. These operating modes have been integrated in the ASPICE system, and have been tested by patients in a neurorehabilitation program. We expect that the results of the clinical validation will be fundamental for further advancement and adaptation of the ASPICE system.

Other aspects which were addressed during the development of the system have not been treated here. For example, the video feedback from the robot camera to the Control Unit is necessary for assisting the user in driving the robot, and for increasing his sense of presence within the environment. Each image frame acquired by the robot camera undergoes an on-board JPEG compression before being streamed over a wireless connection to the ASPICE Control Unit. A GUI for head control has also been developed to allow the user to move the robot head and point the camera in any desired direction. Another feature of the system is a GUI for vocal requests, which improves the communication of the patient with the caregiver. When the robot receives a vocal request (e.g., 'I am thirsty' or 'Please come'), it travels to a designated destination (where the caregiver is) and plays the corresponding prerecorded audio file with its speakers. Another issue that deserved attention is AIBO's walking gait, which was designed on the basis of patient feedback; for example, a typical request was to reduce the noise caused by leg contact with the ground.

References

[1] M. Bousbia-Salah, M. Fezari, and R. Hamdi, "A navigation system for blind pedestrians," in *16th IFAC World Congress*, 2005.

[2] S. Caselli, E. Fantini, F. Monica, P. Occhi, and M. Reggiani, "Toward a mobile manipulator service robot for human assistance," in *1st Robocare Workshop*, 2003.

[3] P. Harmo, T. Taipalus, J. Knuuttila, J. Vallet, and A. Halme, "Needs and solutions: home automation and service robots for the elderly and disabled," in *2005 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2005, pp. 2721–2726.

[4] A. Belic, B. Koritnic, V. Logar, S. Brezan, V. Rutar, G. Kurillo, R. Karba, and J. Zidar, "Identification of human gripping-force control from electro-encephalographic signals by artificial neural networks," in *16th IFAC World Congress*, 2005.

[5] L. R. Hochberg, M. D. Serruya, G. M. Friehs, J. A. Mukand, M. Saleh, A. H. Caplan, A. Branner, D. Chen, R. D. Penn, and J. P. Donoghue, "Neuronal ensemble control of prosthetic devices by a human with tetraplegia," *Nature*, vol. 442, pp. 164–171, 2006.

[6] K. Wada, T. Shibata, T. Saito, K. Sakamoto, and K. Tanie, "Psychological and social effects of one year robot assisted activity on elderly people at a health service facility for the aged," in *2005 IEEE Int. Conf. on Robotics and Automation*, 2005, pp. 2796–2801.

[7] T. Gomi and A. Griffith, "Developing intelligent wheelchairs for the handicapped," in *Assistive Technology and Artificial Intelligence*, 1998.

[8] S. Fioretti, T. Leo, and S. Longhi, "A navigation system for increasing the authonomy and the security of powered wheelchairs," *2000 IEEE Trans. on Rehabilitation Engineering*, vol. 8, no. 4, pp. 490–498, 2000.

[9] L. Kitagawa, T. Kobayashi, T. Beppu, and K. Terashima, "Semi-autonomous obstacle avoidance of omnidirectional wheelchair by joystick impedance control." in *2001 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 4, 2001, pp. 2148–2153.

[10] F. Cincotti, F. Aloise, F. Babiloni, M. G. Marciani, D. Morelli, S. Paolucci, G. Oriolo, A. Cherubini, F. Sciarra, F. Mangiola, A. Melpignano, F. Davide, and D. Mattia, "Brain-operated assistive devices: the ASPICE project," in *1st IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics*, 2006.

[11] M. Kanamori *et al.*, "Pilot study on improvement of quality of life among elderly using a pet-type robot," in *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 2003.

[12] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[13] A. Cherubini, G. Oriolo, F. Macrì, F. Aloise, F. Cincotti, and D. Mattia, "A vision-based path planner/follower for an assistive robotics project," in *Proceedings of 1st International Workshop on Robot Vision, VISAPP*, 2007 (to appear).

[14] P. I. Corke, *Visual control of robots: high performance visual servoing*. Research Studies Press LTD, 1996.

[15] T. Rofer *et al.*, "Robocup 2005 german team technical report," Center for Computing Technology - Universität Bremen, Tech. Rep., 2005.

[16] B. Hengts, D. Ibbotson, S. B. Pham, and C. Sammut, "Omnidirectional motion for quadruped robots," in *Robocup International Symposium, LNAI 2377*, A. Birk, S. Coradeschi, and S. Tadokoro, Eds. Springer, 2001.

[17] C. Canudas de Wit, B. Siciliano, and G. Bastin, *Theory of Robot Control*. Springer, 1996.

[18] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[19] A. De Luca and G. Oriolo, "Local incremental planning for non-holonomic mobile robots," in *1994 IEEE Int. Conf. on Robotics and Automation*, 1994, pp. 104–110.

[20] E. W. Dijkstra, "A note on two problems in connection with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[21] F. Cincotti, D. Mattia, F. Aloise, S. Bufalari, G. Schalk, G. Oriolo, A. Cherubini, M. G. Marciani, and F. Babiloni, "Non invasive brain-computer interface systems: towards their application as assistive technology," *submitted to Brain Research Bulletin*, 2006.