Spartacus Attending the 2005 AAAI Conference

F. Michaud (contact author), C. Côté, D. Létourneau, Y. Brosseau, J.-M. Valin, É. Beaudry, C. Raïevsky, A. Ponchon, P. Moisan, P. Lepage, Y. Morin, F. Gagnon, P. Giguère, M.-A. Roux, S. Caron, P. Frenette, F. Kabanza

LABORIUS - Research Laboratory on Mobile Robotics and Intelligent Systems

Department of Electrical Engineering and Computer Engineering

Université de Sherbrooke, Sherbrooke (Québec, Canada) J1K 2R1

http://www.gel.usherbrooke.ca/laborius

Email: laborius-challenge@listes.USherbrooke.ca

Abstract

Spartacus is our robot entry in the 2005 AAAI Mobile Robot Challenge, making a robot attend the National Conference on Artificial Intelligence. Designing robots that are capable of interacting with humans in real-life settings can be considered the ultimate challenge when it comes to intelligent autonomous systems. One key issue is the integration of multiple modalities (e.g., mobility, physical structure, navigation, vision, audition, dialogue, reasoning). Such integration increases the diversity and also the complexity of interactions the robot can generate. It also makes it difficult to monitor how such increased capabilities are used in unconstrained conditions, whether it is done while the robot is in operation of afterwards. This paper reports solutions and findings resulting from our hardware, software and decisional integration work on Spartacus. It also outlines perspectives in making intelligent and interaction capabilities evolve for autonomous robots.

Introduction

Introduced in 1999, the AAAI Mobile Robot Challenge (or simply the AAAI Challenge) consists of having a robot start at the entrance of the conference site, find the registration desk, register, perform volunteer duties (e.g., guard an area) and give a presentation (Maxwell *et al.* 2004). The long-term objective is to have robots participate just like humans attending the conference.

We became interested in taking on this challenge because it deals with fundamental integration challenges in making an autonomous mobile robot operate in natural settings. Table I presents the characteristics of all the AAAI Challenge entries since the event started in 1999 (Meeden *et al.* 2000; Schultz 2001; Yanco & Balch 2003; Kuipers & Stroupe 2003; Maxwell *et al.* 2004; Smart *et al.* 2005). We can observe a progression in the level of integration demonstrated by the robots. For instance, our 2000 Pioneer 2 robot named Lolitta used sonars as proximity sensors, navigated in the environment by reading written letters and symbols using a pan-tilt-zoom (PTZ) camera, interacted with people through a touch-screen interface, displayed a graphical face to express the robot's emotional state, determined what to do next using a finite-state machine (FSM), recharged itself when needed, and generated a HTML report of its experience (Michaud *et al.* 2001). EMIB (Emotion and Motivation for Intentional selection and configuration of Behavior-producing modules) was the computational architecture used to integrate all the capabilities into a fully autonomous system (Michaud 2002). Grace came in 2003 (Simmons *et al.* 2003), mounted on a bigger robotic platform, adding vocal commands, map navigation, registration-line detection and natural language (NL) understanding. Software integration became a predominant issue with such a diverse set of capabilities on Grace. The different computational modules had to be manually activated at the appropriate time during the different steps of the Challenge. Another B21 robot named Lewis (Smart *et al.* 2003) presented in 2004 an integrated implementation for the Challenge's tasks, using CMU's robot navigation toolkit (CARMEN (Montemerlo, Roy, & Thrun 2003)) for path planning and localization. This robot had previously demonstrated the use of a framing algorithm, allowing it to take pictures of people in open settings (Byers *et al.* 2003).

Reported difficulties regarding these entries are summarized by the following (Gockley

TABLE I

AAAI Robot Challenge Participants

NAME	HARDWARE	CHARACTERISTICS	DEMONSTRATION
FOOF 1999 USC	Pioneer platform; PTZ color camera; Arrow support (contact switches)	Behavior-based navigation; Direction pointing using a fluorescent orange arrow	Navigation without a map, seeking help from humans
OB2K 1999 CMU	Scout platform; Speech synthesis; Screen-based GUI with NL input	Three-tiered architecture (parser, sequencer, skills); Goal state defined from NL	Navigation with or without a map, seeking help from humans
CEREBUS 2000 Northwestern U.	Magellan platform with camera; Speech synthesis; Screen-based interface	Behavior-based navigation; NL understanding and response	Presentation
Lolitta Hall 2000 U. Sherbrooke	Pioneer platform; PTZ color camera; Touchscreen interface; Charging station	Hybrid architecture (EMIB); Navigation by reading symbols; FSM scheduling; Interaction through menus; Animated face display	Integrated version of the entire Challenge; Sign recognition; Presentation using HTML reports; Guard a room
CoWorker 2000, iRobot	CoWorker platform; PT color camera; Bidirectional audio	Internet teleoperation; Waypoint navigation	Navigation portion of the entire Challenge
Leo 2000, MIT	B21 platform; Laser range finder	Large-scale concurrent mapping and localization	"Hands-off" autonomous locomotion
Grace & George 2002, 2003 CMU, NRL, Metrica, Northwestern, Swarthmore	B21 platforms; Laser range finder; Flat panel display; PTZ color camera; Stereo vision on PT unit; 1 wireless microphone headset; 2 onboard computers (LINUX); 2 laptop computers (Windows)	Direction using audio input; Map-based navigation; Line detection; Gesture recognition; People/Color object tracking; Message reading; Animated face display; ViaVoice; OpenGL Festival; NAUTILUS NL	Controlled execution of the entire Challenge; Software integration using IPC; Navigation using vocal commands and CARMEN; Error recovery; Natural language interaction
Lewis 2003, 2004 Washington U.	B21 platform; Laser range finder; Directed perception PT; unit; 2 color cameras; Touchscreen interface; Keyboard	FSM scheduling; Map-based navigation; Color object tracking; Message reading; Festival	Integrated version of the entire Challenge; Navigation using CARMEN; Sign recognition; Preprogrammed speech scripts
Spartacus 2005 U. Sherbrooke	Custom-designed platform; Laser range finder; PTZ color camera; Touchscreen interface; Electronic display; Microphone array; Business card dispenser; 1 onboard computer; 2 laptop computers; 1 external laptop	Hybrid architecture (MBA); Message reading; Sound localization, tracking and separation; Planning and scheduling; Nuance; Festival	Integrated version of the entire Challenge; Navigation using CARMEN; Mapping using pmap; Localization of visual and audio messages; Temporal task sequencing

et al. 2004; Smart *et al.* 2003; Maxwell *et al.* 2004; Simmons *et al.* 2003):

- Robust integration of different software packages and intelligent decision-making capabilities;
- Natural interaction modalities in real-life settings (speech, gesture recognition);
- Adaptation to environmental changes for localization;
- Monitoring and reporting decisions made by the robot.

The need to address these problems motivates our work on Spartacus, our 2005 AAAI Challenge robot entry described in this paper. Section 2 presents Spartacus' robotic platform and computational architecture. Section 3 describes software integration tools and capabilities implemented on the robot. Section 4 presents results and observations in terms of integrated capabilities and performances, followed by Section 5 with a discussion on important issues that we want to address in the near future. Section 6 concludes the paper.

Spartacus the Autonomous Robot

Shown in Figure 1, Spartacus is a custom-built robotic platform designed for high-level interaction with people in real-life settings. It is a differential steering wheeled platform with a humanoid-like upper torso. Spartacus is equipped with a SICK LMS200 laser range finder, a Crossbow IMU400 MEMS inertial measurement unit, a Sony SNC-RZ30N 25X PTZ color camera, an array of eight microphones placed on the robot's body, a touchscreen interface, an audio amplifier and speakers, a business card dispenser and an LED electronic display. The robot is powered by eight 12 Volt 7 Amp-Hour Sealed Lead Acid batteries. Spartacus also carries its own chargers so that it can be plugged directly into a regular electric outlet. Low-level interfaces of most of these components are implemented following a distributed approach, using different microcontroller subsystems that communicate with each other through a shared CAN 2.0B 1 Mbps bus (Michaud *et al.* 2005). This approach facilitates debugging and extensions to the platform. High-level processing is carried out using an embedded Mini-ITX computer (Pentium M 1.7 GHz). The Mini-ITX computer is connected to the low-level microcontrollers through a CAN bus device, to the laser range finder through a serial port and to the serial controller of the PTZ camera. Two laptop computers (Pentium M 1.6 GHz) are also installed on the

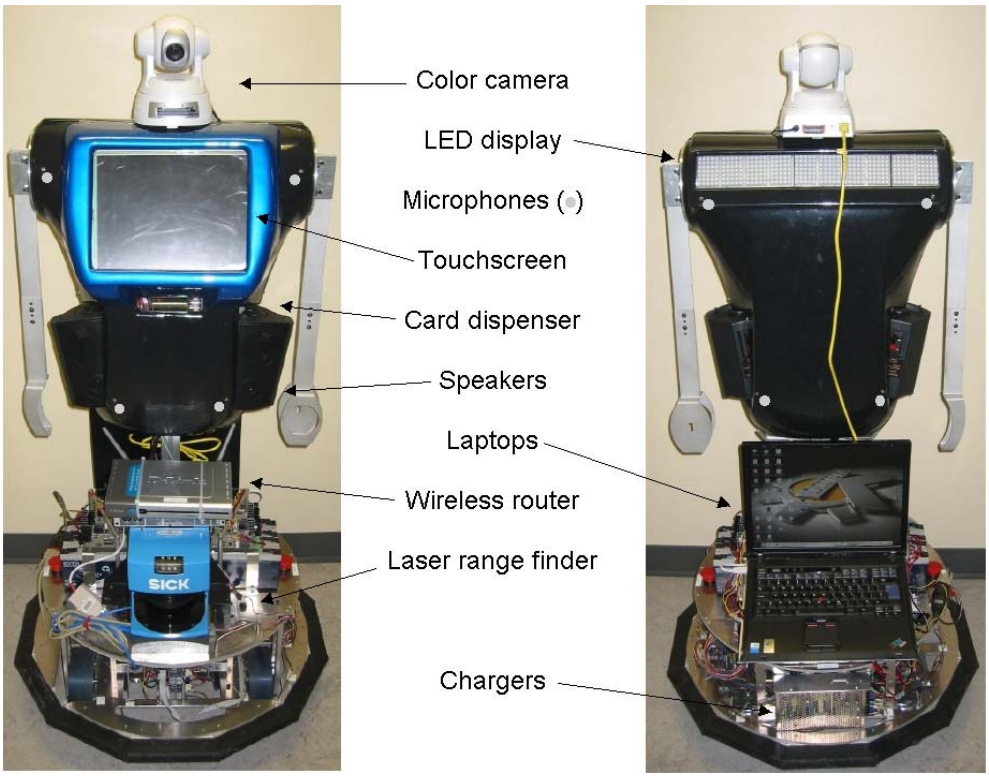

Fig. 1. Spartacus (front view, back view).

platform. One is equipped with a RME Hammerfal DSP Multiface sound card using eight analog inputs that simultaneously sample signals coming from the microphone array. It is also connected to the audio amplifier and speakers using the audio output port. The other laptop does video processing and is connected to the camera through a 100 Mbps Ethernet link. Communication between the three on-board computers is accomplished with a 100 Mbps Ethernet link. Communication with external computers can be achieved using the 802.11g wireless technology, giving the ability to easily add remote processing power or capabilities if required. All computers are running Debian GNU/Linux.

The computational architecture we are developing to manage Spartacus' decision-making capabilities is based on the notion of motivated selection of behavior-producing modules. We refer to it as MBA, for Motivated Behavioral Architecture. The architecture contains different motivational sources derived from perceptual influences, pre-determined scenarios, navigation algorithms, planning or other types of reasoning algorithms. Our intention in distinguishing these influences through different motivational sources is to simplify pro-

gramming of the robot's intentions in accomplishing various tasks. Motivations are used to efficiently balance the robot's adaptation to the contingencies of the world and the accomplishment of its goals. Our computational architecture tries to combine multiple influences coming from behavior-based systems (Matarić 1997; Arkin 1998), path planning and deliberation (Haigh & Veloso 1998; Simmons & Apfelbaum 1998), shared memory (Maxwell & et al. 2001) and motivational drives (Parker 1998; Breazeal *et al.* 2000; Michaud & Vu 1999; Stoytchev & Arkin 2004), demonstrated on various robotic platforms over the years. All of these elements are important for autonomous robots operating in unpredictable and partially observable environments, trying to establish a balance between what can be foreseen and what emerges from the interaction dynamics with the world.

Figure 2 represents MBA. It is composed of three principal elements:

1. Behavior-producing modules (BPMs) define how particular percepts and conditions influence the control of the robot's actuators. They can be typical behavior-based reactive controllers with or without internal states, goal-oriented behaviors or other types of behaviors. The actual use of a BPM is determined by the arbitration scheme (realized through BPM Arbitration, which can be priority-based, data fusion, action selection or defuzzification, depending on the implementation) and the BPM's activation conditions, as derived by the BPM Selection module.
2. Motivational sources (or Motivations) recommend the use or the inhibition of tasks to be accomplished by the robot. Motivational sources are categorized as either instinctual, rational or emotional. Instinctual motivations provide basic operation of the robot using simple rules. Rational motivations are more related to cognitive processes, such as navigation and planning. Emotional motivations monitor conflictual or transitional situations between tasks. They also monitor changes in commitments the robot establishes with other agents, humans or robots, in its environment. Motivations can be derived from percepts, results and states of current goals, and by monitoring tasks or BPMs (using the Exploitations link). Behavior exploitation indicates when a BPM is effectively used to control the robot's actuators, and serves as an abstraction of the robot's interactions within the environment. An active behavior may or may not be used to control the robot, depending on the sensory conditions it monitors and the

arbitration mechanism used to coordinate the robot's behaviors. An active behavior is exploited only when it provides commands that actually control the robot.

3. Dynamic Task Workspace (DTW) organizes tasks in a tree-like structure according to their interdependencies, from high-level/abstract tasks (e.g., deliver message), to primitive/BPM-related tasks (e.g., avoid obstacles). Through the DTW, motivations exchange information asynchronously on how to activate, configure and monitor BPMs. Motivations can add and modify tasks by submitting modification requests (m), queries (q) or subscribe to events (e) regarding the task's status.

Motivations are kept generic and independent from each other and from the BPMs through tasks posted in the DTW. For instance, one instinctual motivational source may monitor the robot's energy level to issue a recharging task in the DTW, which activates a behavior that would make the robot detect and dock into a charging station. Meanwhile, if the robot knows where it is and can determine a path to a nearby charging station, a path planning rational motivation can add a subtask of navigating to this position. With multiple tasks being issued by the motivational sources, the BPM Selection module determines which BPMs are to be activated according to recommendations (rec) made by motivational sources. A recommendation can either be negative, neutral or positive, or take on real values within this range regarding the desirability of the robot to accomplish specific tasks. The activation values (Activations) reflect the resulting robot's intentions derived from interactions between the motivational sources through the DTW.

MBA is a generalization of EMIB (Michaud 2002). The System Know-How (SNOW) module, not present in EMIB, acts as an adapter between BPMs and DTW, making it is possible to decouple task representation contained in the DTW from BPMs. Consequently, both of them can use their own domain representations and concepts without having to share information directly. For instance, the SNOW contains the necessary knowledge to define and communicate tasks parameters (p) and behaviors results (res) between BPMs and DTW. This facilitates the addition of new motivational sources and tasks to the decision processes, with minimal changes to the software implementation of the computational architecture. Motivations and tasks are kept independent of the hardware implementation, for code reuse from one robotic platform to another.

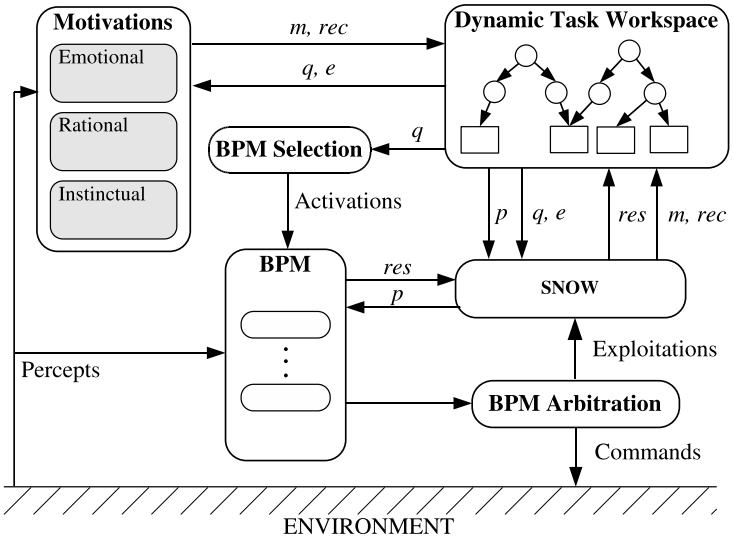

Fig. 2. MBA computational architecture.

Software Integration on Spartacus

Our 2005 implementation attempted to have Spartacus participate in all the AAAI Mobile Robot events, to ensure the validity of the integration work in multiple contexts. The following capabilities were integrated:

- **Autonomous Navigation**. When placed at the entrance of the convention center, the robot autonomously find its way to the registration desk by wandering and avoiding obstacles, searching for information regarding the location of the registration desk, following people talking or going towards communicated directions. Once registered, the robot can use a map of the convention center. The two navigation tools used are CARMEN and pmap. CARMEN, the Carnegie Mellon navigation toolkit (Montemerlo, Roy, & Thrun 2003), is a software package for laser-based autonomous navigation using previously generated maps. The pmap package[1] provides libraries and utilities for laser-based mapping in 2D environments to produce high-quality occupancy grid maps. These maps were then converted into CARMEN's format.
- **Vision Processing**. Extracting useful information in real-time from images taken by the onboard camera enhances interaction with people and the environment. For

[1]http://robotics.usc.edu/~ahoward/pmap

instance, the robot could benefit from reading various written messages in real-life settings, messages that can provide localization information (e.g., room numbers, places) or identity information (e.g., reading name badges). Object recognition and tracking algorithms also make it possible for the robot to interact with people in the environment. We use two algorithms to implement such capabilities : one that can extract symbols and text from a single color image in real world conditions (Letourneau, Michaud, & Valin 2004); and another one for object recognition (Lienhart & Maydt 2002) and tracking to identify and follow regions of interest in the image such as human faces and silhouettes (using the Open CV library). Once identified, these regions can be tracked using color information, as achieved in (Perez *et al.* 2002).

- **Sound Processing**. Hearing is an important sense for interaction in open settings. Simply using one or two omnidirectional microphones on a robot may not be enough: it seems very difficult to filter out all of the noise generated in public places. Using a microphone array reveals to be a robust solution for the localization, tracking and separation of sound sources. Our approach is capable of simultaneously localizing and tracking up to four sound sources that are in motion over a 3 meters range, in the presence of noise and reverberation (Valin *et al.* 2004; Valin, Michaud, & Rouat 2006). We also developed a method for real-time simultaneous separation of sound sources (Valin, Rouat, & Michaud 2004). The separated sources are then processed to recognize vocal messages, using software packages such as Nuance[2]. We tested Nuance with data generated by our system, with speech utterances (i.e., four consecutive digits spoken in English) simultaneously made by three separate speakers in an open environment (not crowded). Recognition accuracy was 84% on average for the three speakers, including 87% accuracy for the front speaker. In the same conditions, human listeners achieved 81% recognition rate, but only by focusing on the front speaker alone. Only one of the five listeners was able to achieve the same accuracy as the robot, but again for only one (the front speaker) out of the three speakers.
- **Touchscreen Display**. Various information can be communicated through this device, such as: receiving information from people using a menu interface; displaying

[2]http://www.nuance.com

graphical information such as a PowerPoint presentation or a map of the environment; requesting the execution of a volunteer task (i.e., deliver a message, guard); and expressing emotional states using a virtual face. We chose to use QT3[3] for the graphical interface development because of ease of use and portability.

Software integration of all these elements is not a simple plug-and-play process. Most of them were developed independently, adopting different design and implementation requirements. This led us to develop MARIE (Mobile and Autonomous Robot Integrated Environment)[4] (Cote *et al.* 2006a; 2006b). MARIE is a system integration framework used to link multiple software packages. MARIE is oriented towards a rapid prototyping approach to develop and integrate new and existing software for robotic systems. MARIE's design efforts have been focused on distributed robotics component-based middleware framework development, enhancing reusability of applications and providing tools and programming environments to build integrated and coherent robotics systems. The robotics community still has to explore a great variety of ideas, application areas (each one having its own set of constraints, e.g., space, military, human-robot interaction), while coping with continuously evolving computing technologies. Consequently, being able to reuse previous implementations and adapt to upcoming robotics standards easily, without major code refactoring, provide longer life cycles for actual and future implementations.

MARIE's philosophy is based on the creation of reusable software blocks, called components, which implement functionalities by encapsulating existing applications, programming environments or dedicated algorithms. The idea is to configure and interconnect these components through common communication mechanisms (either on the same processing node, or distributed across multiple nodes running similar or different operating systems), to implement the desired functionalities, using software applications and tools available through MARIE.

MARIE provides component development frameworks and software tools, such as communications abstraction, processes management, data filters and conversions, configurations and different execution models (event handling, iteration loop, finite state machine, etc.). Four types of components are used: Application Adapter (AA), Communication

[3]http://www.trolltech.com/products/qt

[4]http://marie.sourceforge.net

Adapter (CA), Application Manager (AM) and Communication Manager (CM). AA's role is to interface applications into MARIE's application design framework and make them interact with each other. CA ensures communication compatibility between AAs. For example, it makes it possible to connect an AA providing data at a fixed rate with an AA requiring it asynchronously. Currently available CAs in MARIE are Splitters, Switches, Mailboxes and Shared Maps. A Splitter sends data from one source to multiple destinations without the sender needing to be aware of the receivers. A Switch acts like a multiplexer sending data to the selected output. A Mailbox creates a buffered interface between asynchronous components. A Shared Map is used to share data between multiple components. AMs and CMs are system level components that instantiate and manage components either locally or across distributed processing nodes. They can, for instance, restart a component not responding for a while, or can move components from one processing node to another to avoid CPU overloads.

Although the use of MARIE's frameworks and software tools is highly encouraged to save time and efforts, MARIE is not limited to them. Developers can use the best solution to integrate software applications and interconnect components by having the possibility to extend or adapt existing components and available frameworks. MARIE's underlying philosophy is to complement existing applications, programming environments or software tools, and therefore it is to be used only when required and appropriate.

Results

Figure 3 illustrates Spartacus' overall implementation of MBA using the capabilities made available through MARIE (additional technical descriptions can be found in (Cote *et al.* 2006b; Beaudry *et al.* 2005; Michaud *et al.* 2005)). BPM Arbitration scheme used is priority-based. Recommendations and BPM Activations are binary values. Four actuators are controlled by BPMs, each of them having one or more associated behaviors:

1. The Motor actuator has eight associated behaviors: Emergency Stop, stopping the robot when abnormal conditions are detected; Rest, stopping the robot from moving; Avoid, making the robot move safely in the environment; Obey, following vocal requests; Dock, stopping the robot while waiting to be connected to a charging station; Goto, directing the robot to a specific location; Follow Audio Localization, making the

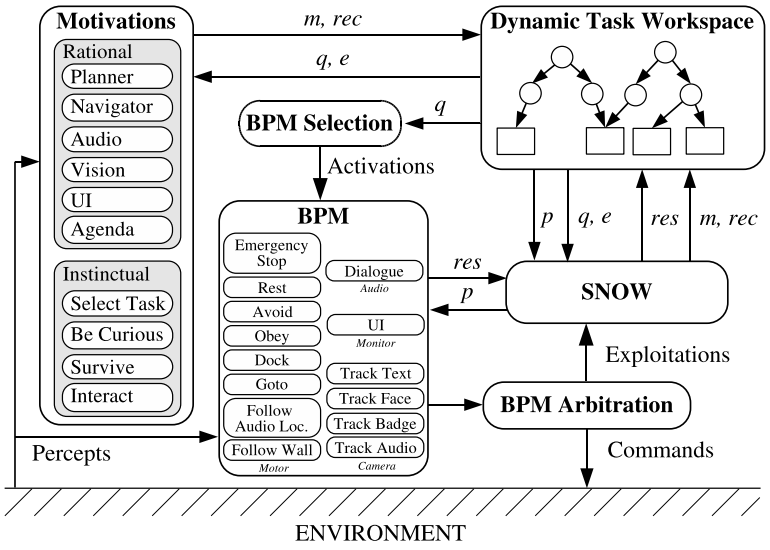

Fig. 3. MBA computational architecture implemented for Spartacus 2005 entry.

robot follow an audio source; Follow Wall, making the robot follow a wall (or corridor) when detected, otherwise generating a constant forward velocity.

2. The Camera actuator (PTZ color camera) is controlled by four behaviors: Track Text, centering and zooming on detected text areas in the image; Track Face, centering a perceived face in the image; Track Badge, centering and zooming on a detected name badge in the image; Track Audio, pointing the camera in the direction of an audio source.
3. The Audio actuator (i.e., the sound card output) is associated with the Dialogue behavior.
4. The Monitor actuator is linked to the UserInterface (UI) behavior, using the touch-screen interface to interact with users.

Only instinctual and rational motivations are implemented in this version, with rational motivations having greater priority over instinctual ones in case of conflicts (e.g., conflicting recommendations for the same task). For instinctual motivations, Select Task selects one high-level task when none has yet been prioritized. For instance, from a set of tasks that require the robot to go to specific locations, this motivation selects the task with the location physically closest to the robot. Be Curious motivates the robot to discover

and collect information about its environment. Survive urges the robot to maintain its physical integrity by recommending to avoid obstacles. Interact is a process allowing the robot to socialize with people. The other modules are rational motivations. Planner is a motivational source sequencing primitive tasks necessary to accomplish high-level tasks according to temporal constraints and the robot's limited capabilities (as defined by BPMs). Our first implementation is a simple reactive planning module that interleaves planning and execution (Beaudry *et al.* 2005), like (Haigh & Veloso 1998) and (Lemai & Ingrand 2004). Navigator determines the path to a specific location, as required for tasks in the DTW. Audio and Vision motivate the robot to do tasks according to what is perceived (e.g., track badge, localize sound sources). UI is a process executing user's requests for tasks. Agenda generates predetermined tasks according to the AAAI Mobile Robot Competition context (e.g., find registration, register, go to the presentation room, etc., for the Challenge).

MBA's implementation using MARIE requires 42 components (~ 50 000 lines of code) composed of 26 AAs, 14 CAs and two external applications (the Audio Server and NUANCE). Except for the two external applications, component interconnections are all sockets-based using Push, Pull and Events dataflow communication mechanisms (Zhao 2003) with XML encoding for data representation. The Audio Server and NUANCE use their own communication protocols. MARIE's current version is in C++ (~ 10 000 lines of code) and uses ACE (Adaptive Communication Environment) library (Schmidt 1994) as low-level operating system functions. MARIE's Application Manager is partially implemented, requiring AAs and CAs to be initialized manually using scripted commands. Also, the Communication Manager is not yet implemented, meaning that component configuration must be set manually.

Representation of Spartacus' software architecture is illustrated in Figure 4. Distributing applications across multiple processing nodes is not difficult with MARIE, having chosen network sockets as the communication mechanism. MBA also reveals to be well adapted for a distributed implementation over multiple processing nodes. The Vision component uses one on-board laptop computer, Audio components uses the second one, and all other components are executed on Spartacus' on-board Mini-ITX computer. Blocks with similar

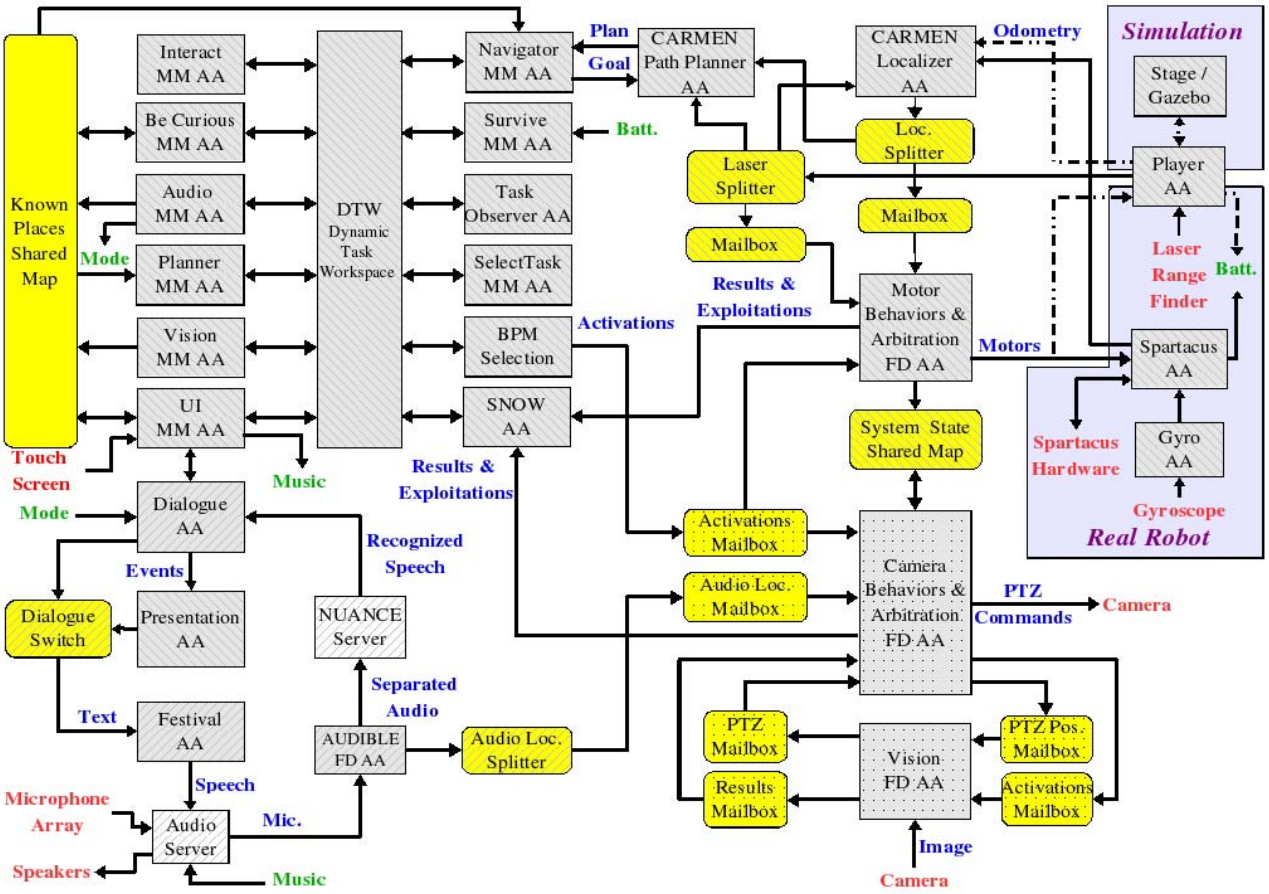

Fig. 4. Representation of Spartacus' software architecture implemented using MARIE.

backgrounds in Figure 4 run on the same computer.

Spartacus's software architecture provides two setups. For the Real Robot setup, SpartacusAA combines wheels odometry with gyroscopic data, and pushes the result at a fixed rate (200 Hz) to its interconnected components. Laser data is collected by PlayerAA, interfacing the Player library specialized for sensor and actuator abstraction (Vaughan, Gerkey, & Howard 2003), supporting the SICK LMS200 laser range finder. PlayerAA pushes data at a fixed rate (200 Hz) to connected components. For the Simulation setup, odometry and laser data are both collected with PlayerAA, generated either by Stage (2D) or Gazebo (3D) simulators (Vaughan, Gerkey, & Howard 2003). CARMEN Localizer AA and CARMEN Path Planner AA provide path planning and localization capabilities. CARMEN is composed of small processes communicating through a central server. CARMEN's inte-

gration was realized by creating an AA that starts several of these processes depending on the required functionality and on data conversion from CARMEN's to MARIE's format.

FlowDesigner (FD) program (Cote *et al.* 2004) is a modular data-flow programming environment that facilitate visualization and understanding of the robot's processing loops, sensor conditioning and actuator control. RobotFlow is its robotic toolbox. FlowDesigner's graphical user interface facilitates the interconnection of reusable software blocks without having to compile the program every time minor changes are made, making it appropriate for rapid prototyping. FlowDesigner and RobotFlow are used to implement Behavior & Arbitration FD AA, handling BPMs and their arbitration. FlowDesigner use a synchronous pull mechanism to get data coming from different elements such as localization, path plan, laser, audio localization, dialogue command and system states, requiring the use of Mailbox CA components. By buffering input data, mailboxes allows AAs running at different rates to be interconnected. Behavior & Arbitration FD AA generates motor commands at a fixed rate (5 Hz).

The Audio Server is interfacing the RME Hammerfal DSP Multiface sound card, and NUANCE Server is interfacing NUANCE. DialogueAA is a stand-alone AA that manages simultaneous conversations with people. This is made possible with the use of AUDIBLE FD AA, interfacing our sound source localization, tracking and separation algorithms implemented with RF/FD and using Spartacus' microphone array. It generates a number of separated audio channels that are sent to NUANCE Server and Behavior & Arbitration FD AA. Integrating NUANCE in an AA was challenging since it is a proprietary application with a fixed programming interface, and because its execution flow is tightly controlled by NUANCE's core application, which is not accessible from the available interface. To solve this problem, we created an independent application that uses a communication protocol already supported by MARIE. Recognized speech data is sent to Dialogue AA, responsible of the human-robot vocal interface. Speech generated by the robot is handled by Festival (Taylor 1999). The Dialogue AA conversation context mode is selected by the Audio MM AA, monitoring the tasks present in the DTW that require speech interaction.

The global execution of the system is asynchronous, having most of the applications and AAs pushing their results at variable rates (determined by the computation length of

their algorithms when triggered by new input data). Synchronous execution is realized by having fixed rate sensor readings and actuator command writings.

Using this implementation, Spartacus did enter all events (Challenge, Scavenger Hunt, Open Interaction) at the 2005 AAAI Mobile Robot Competition. Each event was programmed as a special configuration mode, which could be requested by the experimenter (using speech or from the tactile interface shown in Figure 5).

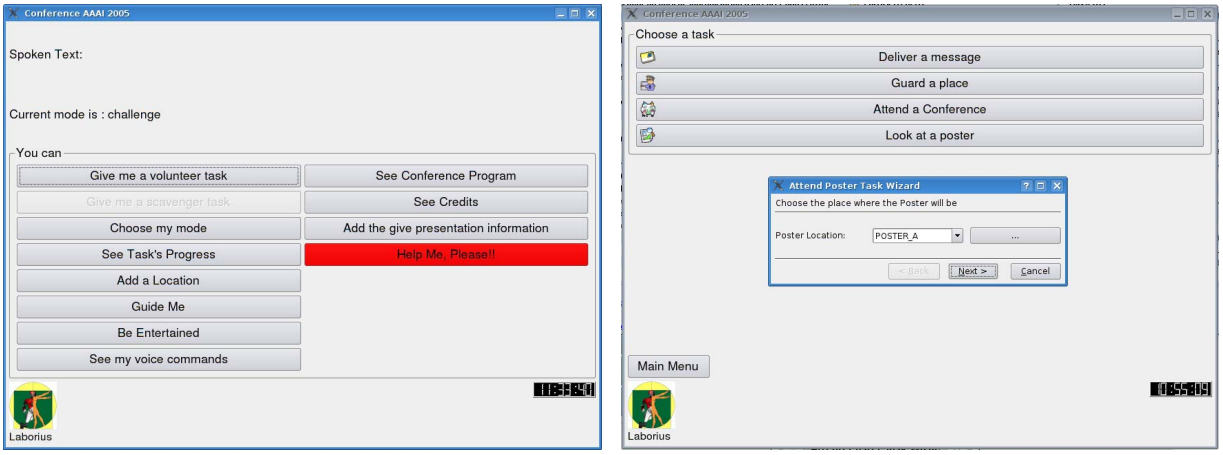

Fig. 5. Touchscreen interfaces: Challenge mode (left) and volunteer tasks (right).

Figure 6 shows pictures taken during the event. Complete integration was demonstrated, combining the following: map building with localization and path planning, audio tracking of speakers around the robot (positioning the camera in the direction of where the loudest sound came from), speech recognition in "cocktail party effect" conditions, scripted vocal response to specific requests (around 30 different requests), reading the room names (printed on 8.5" $\times$ 11" sheets of paper using Arial font, similar to AAAI badges), and dynamic scheduling of tasks set according to temporal constraints. Using pmap to obtain an accurate map of the environment was quite simple: it only required to cover the environment to map by controlling the robot from a joystick, gathering positioning and laser data so that pmap could generate the map. This was done before the event took place and took approximately 45 minutes to complete. Using the map and CARMEN to localize Spartacus in the world, events such as recognizing a specific sentence or reading a particular message were memorized and labeled on the map for further reference.

Fig. 6. Pictures of Spartacus at the 2005 AAAI Challenge.

Unfortunately, to demonstrate Spartacus' speech recognition capabilities in open settings, we had to sacrifice visual tracking of people. More CPU processing power is required to fully integrate all of our available vision software. Also, speaker volume could not be set to its maximum since a humming sound coming out of the speakers was interfering with two of the nearby microphones.

Demonstrating a complete integrated system with Spartacus does not mean however that all of the underlying problems are solved. The experimental conditions revealed to be extremely challenging. Here is a list of the main issues we experienced:

- Spartacus' motor drives were set to safely operation on hard surface conditions, but not on carpet (making the robot go slower than expected). We did not want to make any changes to this hardware threshold, in case it would have damaged the motor drives. Eventually, to move on soft carpets and in crowded environments, Spartacus could be designed to use an AZIMUT base for its locomotion. AZIMUT, also demonstrated at the 2005 AAAI Mobile Robot Exhibition, is a symmetrical platform with four independent articulations that can be wheels, legs or tracks, or a combination of these (Michaud *et al.* 2005). By changing the direction of its articulations, AZIMUT is capable of moving sideways without changing its orientation, making it omnidirectional. Such locomotion capability would surely help navigate in crowded conditions.
- Trying to get in one of the five possible elevators in the hotel, as the doors opened up, Spartacus did not have enough time to get onboard before the doors would close.

Not knowing which door was going to open, and not being able to keep one open long enough to let the robot move in front of it and enter, we finally had to manually operate the robot inside one elevator.

- Trials were conducted in very difficult conditions: registering to a conference does not usually happen right in the midst of a reception, when tables are being installed, in conditions in which it is even difficult for humans to sustain a conversation, fighting the crowd. We should have made Spartacus to be more directive in what it wanted to do (e.g., asking people to move away if it has to do something), and not always kept on responding to people's requests.
- Being able to simultaneously extract sound sources and process them with NUANCE is CPU intensive. Explaining Spartacus capabilities next to it during demos made Spartacus process long audio streams that sometimes were discarded, and generated inappropriate delays during vocal interactions. Spartacus stops listening only when it speaks so that it does not try to understand itself. This limits the amount of unnecessary processing, but does not allow the robot to understand a request made by someone while Spartacus speaks.
- Displaying what the robot said on the touchscreen revealed quite useful, as it sometimes was difficult to hear what the robot was saying because of the high-level of noise in the environment. This could be improved by displaying bigger graphical objects and messages.
- Spartacus sometimes stopped moving, not listening to vocal requests because it was trying to read an unanticipated but potential message in its view. Visual influences were prioritized over audio, and this should be set according to the robot's task. One additional improvement would be to clearly display what the robot is currently doing on its touchscreen interfaces or its LED electronic display.
- Handling complex situations that may arise when completing the Challenge from start to end requires running the robot for 1 to 1.5 hour (which is approximately Spartacus' entire energetic capability). Trying to demonstrate the capabilities of a planner is therefore difficult in the limited periods of time allowed for evaluation during the event. For instance, during one trial of 19 minutes, the Planner was invoked only 12

times and took on average 1.259 ms (between 0.15 ms to 12.93 ms) for generating simple plans. These plans were generated because of the addition of new tasks in the DTW (3 times), the addition of new locations on the map (3 times), the occurrence of a Message Delivery task (3 times out of 5 because of failure to remain in desired location), and replanning caused by earlier achievement of tasks (1 time). No failures were observed because of a planning error, and no tasks were discarded because of anticipated failures. Longer trials are required to observe and study the performance of the Planner motivation.

- Localization in such conditions was also very challenging. Figure 7 shows the sequence of estimated poses of the robot being given the following mission at the registration desk: take a picture at location A and deliver a message taken at location A to location B. Having estimated poses on an irregular path can be explained by two main reasons: 1) with only a few people around it, the robot had difficulty perceiving walls and obstacles modeled in its map, and 2) during the trial, the reception area was altered (new tables were installed and closed doors were opened, making the map invalid). Under such conditions, with the reception area being pretty busy all the time, Spartacus did not moved much. Therefore, we spent very limited time on evaluating Spartacus' localization components and instead used the robot's available energy to focus on its interaction and planning capabilities.

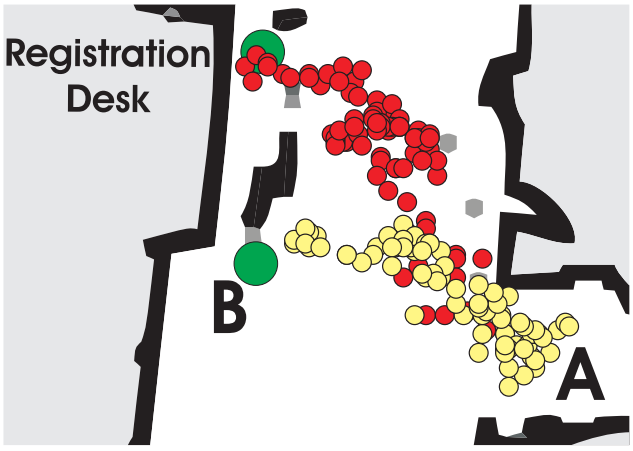

Fig. 7. Trace of CARMEN's estimated belief states from the Registration desk to location A (dark grey dots), and from location A to location B (light grey dots).

- It is also very difficult to come up with quantified results regarding the performance of the robot in the open settings and uncontrolled conditions of the Challenge. To our knowledge, no other entry (listed in Table I) has yet presented such analysis. A way to memorize what happened during a trial, both internally and externally to the robot, is required. This would enable off-line analysis of the performance of the robot, and provide useful information for comparison over trials. For instance, with everything happening so fast with the robot being placed in the reception area, keeping a complete log of CARMEN's states, the audio streams extracted by AUDIBLE and snapshots of images taken by the camera in the direction of the closest sound sources would have helped extract quantitative and qualitative data about Spartacus' behavior.

Discussion

Spartacus' first implementation provided interesting insights on software integration challenges in designing autonomous mobile robots. Meeting the integration needs using MARIE's rapid prototyping approach revealed to be very valuable. MARIE provides the capability of reusing existing programming packages and interconnect them through a system integration framework to benefit from their respective approaches, instead of having to choose only one of them or to reimplement them. This ensures efficient progress in discovering the underlying issues with autonomous reasoning of mobile robots. It also provides a flexible team development framework. At the peak of Spartacus' software development process, eight software developers were working concurrently on the system. Most of them only used Application Adapters to create their components, conducting unit and blackbox testing with pre-configured system setups given by one integrator. The ability to change between simulation and robotic setups with only few system modifications gave us the possibility to do quick simulations and integration tests. Nearly 75% of the system functionalities were validated in simulation and used as is in the real world setup. In both simulated and real setups, configurations of components receiving laser and odometry data were exactly the same, abstracting data sources and benefiting from components modularity and MARIE'S rapid prototyping approach. Communication protocols and operating system tools for component and application development were added when required. Components were incrementally integrated to the system as they became

available. It took around eight days, spread over a four weeks period, to have a fully integrated system using MARIE.

Being able through MARIE to quickly interconnect components to create a complete implementation, without focusing on optimization right away, proved beneficial. Such an exploration strategy gave us the ability to quickly reject software designs or component implementations without investing too much time and effort. For Spartacus, we originally thought that tighter synchronization between components would be necessary to obtain a stable system and support real-time decision-making. For instance, having connected all of Spartacus' components together, we observed that performances were appropriate with the processing power available as long as we did not overload the computers with too many components. Noticing that, we decided to wait before investing time and energy working on component timing constraints, to focus on Spartacus' integration challenges. In addition, the implementation of AM and CM were at an early stage of development with MARIE at the time of the competition. Not being able to use such functionalities, we had to manually configure and manage many components executed on multiple processors (load balancing, boot sequence, failure detection, etc.), which proved to be an error-prone and non-trivial task. Our implementation therefore confirms the importance of having GUI and system management tools in MARIE (such as AM and CM functionalities (Cote *et al.* 2006a)) to build, configure and manage components automatically.

As our integration work progressed and about three months before the AAAI competition, we started to face serious difficulties in tracking decisions made by the system simply by observing its behaviors in the environment or by looking at logs from different software components, something that was always possible with simpler implementations. We decided to develop a graphical application for on-line monitoring or off-line analysis of the decisions taken by the robot. This application, named the LogViewer and partially shown in Figure 8, revealed to be very valuable. The underlying objectives with the LogViewer is to facilitate integration work by offering visualization tools of the decisional components added to the robot. It requires coherent and user-friendly representation of the integrated components so that developers can concentrate on their combined usage instead of spending time extracting and interpreting data in log files. The upper section of the LogViewer

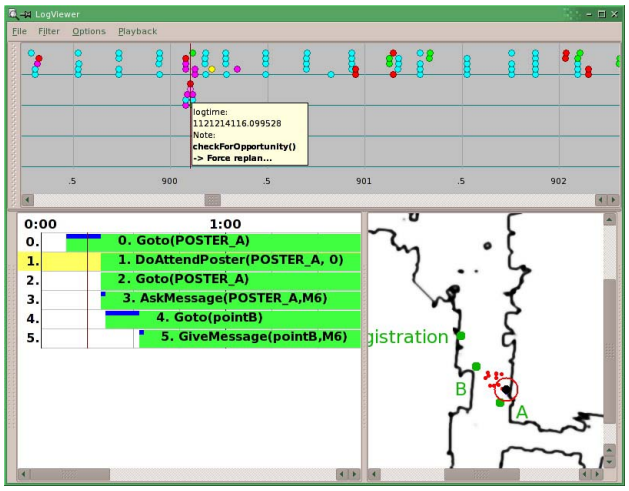

Fig. 8. LogViewer: reduced view of the current plan and the map with labeled places.

contains a timeline view of DTW (first line) and planner events (second lines), and behaviors' activations and exploitations (not shown). The bottom section shows detailed information according to the position of the horizontal marker on the timeline: a list of DTW's tasks and properties (not shown), active motivations (not shown), the current plan (shown), the map of the environment (shown) and the trajectory of the robot (not shown), the behaviors and their activations and exploitations (not shown).

Conclusion

Designing a mobile robot operating in public settings probably addresses the most complete set of issues related to autonomous and interactive mobile robot design, with system integration playing a fundamental role at all levels. With this paper, we explain how these issues are addressed using Spartacus and MBA's computational architecture. Optimization is still required to make the integration work more smoothly and to be able to evaluate the global performance of the robot in accomplishing the Challenge. However, Spartacus made substantial progress in integrating and demonstrating important capabilities for having an autonomous mobile robot participate to a conference. Software like MARIE and the LogViewer illustrate the importance of software engineering tools to address the underlying integration challenges. Such tools play an important part of the scientific process of studying and designing highly-integrated and complex robotic systems.

One of our objectives in the near future is to continue developing these tools and to improve Spartacus' integrated capabilities (e.g., recognizing electric outlets, gestures, and interacting through a natural language interface). We plan to make the LogViewer evolve into a common, context-based interface for both human-robot interaction, performance monitoring, quantitative assessments and comparative evaluations of the robot's behavior in real-life settings. We also want to explore new avenues like developing a comparison framework for different localization tools and planning algorithms or porting the same implementation on robotic platforms from different manufacturers and with heterogeneous capabilities. As entries will move from proof-of-concept demonstrations to robust systems, the AAAI Challenge is becoming a scientific venue directly taking on the artificial intelligence's fundamental objective of working in the wild, messy world[5].

Acknowledgments

The authors gratefully acknowledge the contribution of the Canada Research Chair (granted to F. Michaud), the Natural Sciences and Engineering Research Council of Canada and the Canadian Foundation for Innovation, in the support of this work.

References

Arkin, R. C. 1998. *Behavior-Based Robotics*. The MIT Press.

Beaudry, É.; Brosseau, Y.; Côté, C.; Raïevsky, C.; Létourneau, D.; Kabanza, F.; and Michaud, F. 2005. Reactive planning in a motivated behavioral architecture. In *Proceedings American Association for Artificial Intelligence Conference*, 1242–1247.

Breazeal, C.; Scassellati, B. 1998. Infant-like social interactions between a robot and a human caregiver. *Adaptive Behavior* 8(1):49–74.

Byers, Z.; Dixon, M.; Smart, W. D.; and Grimm, C. M. 2003. Say cheese! Experiences with a robot photographer. In *Proceedings Fifteenth Innovative Applications of Artificial Intelligence Conference*, 65–70.

Côté, C.; Létourneau, D.; Michaud, F.; Valin, J.-M.; Brosseau, Y.; Raïevsky, C.; Lemay,

[5]As stated by Ronald Brachman in the 2005 AAAI Conference Presidential Address.

M.; and Tran, V. 2004. Code reusability tools for programming mobile robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1820–1825.

Côté, C.; Brosseau, Y.; Létourneau, D.; Raïevsky, C.; and Michaud, F. 2006a. Using MARIE in software development and integration for autonomous mobile robotics. *International Journal of Advanced Robotic Systems, Special Issue on Software Development and Integration in Robotics* 3(1):55–60.

Côté, C.; Létourneau, D.; Raïevsky, C.; Brosseau, Y.; and Michaud, F. 2006b. Using MARIE for mobile robot software development and integration. In Brugali, D., ed., *Software Engineering for Experimental Robotics*. Springer Tracts on Advanced Robotics.

Gockley, R.; Simmons, R.; Wang, J.; Busquets, D.; DiSalvo, C.; Caffrey, K.; Rosenthal, S.; Mink, J.; Thomas, S.; Adams, W.; Lauducci, T.; Bugajska, M.; Perzanowski, D.; and Schultz, A. 2004. Grace and George: Social robots at AAAI. Technical Report WS-04-11, AAAI Mobile Robot Competition Workshop. pp. 15-20.

Haigh, K., and Veloso, M. 1998. Planning, execution and learning in a robotic agent. In *Proceedings Fourth International Conference on Artificial Intelligence Planning Systems*, 120–127.

Kuipers, B., and Stroupe, A. 2003. The AAAI-2002 Robot Challenge. *AI Magazine* 24(1):65–76.

Lemai, S., and Ingrand, F. 2004. Interleaving temporal planning and execution in robotics domains. In *Proceeedings National Conference on Artificial Intelligence*, 617–622.

Létourneau, D.; Michaud, F.; and Valin, J.-M. 2004. Autonomous robot that can read. *EURASIP Journal on Applied Signal Processing, Special Issue on Advances in Intelligent Vision Systems: Methods and Applications* 17:1–14.

Lienhart, R., and Maydt, J. 2002. An extended set of Haar-like features for rapid object detection. In *Proceedings of the International Conference on Image Processing*, 900–903.

Matarić, M. J. 1997. Reinforcement learning in the multi-robot domain. *Autonomous*

Robots 4(1).

Maxwell, B. A., and Meeden, L. 2001. Reaper: A reflexive architecture for perceptive agents. *AI Magazine* 22(1):53–66.

Maxwell, B.; Smart, W.; Jacoff, A.; Casper, J.; Weiss, B.; Scholtz, J.; Yanco, H.; Micire, M.; Stroupe, A.; Stormont, D.; and Lauwers, T. 2004. 2003 AAAI robot competition and exhibition. *AI Magazine* 25(2):68–80.

Meeden, L.; Schultz, A.; Balch, T.; Bhargava, R.; Haigh, K. Z.; Bohlen, M.; Stein, C.; and Miller, D. 2000. The AAAI 1999 mobile robot competitions and exhibition. *AI Magazine* 21(3):69–78.

Michaud, F., and Vu, M. T. 1999. Managing robot autonomy and interactivity using motives and visual communication. In *Proceedings International Conference on Autonomous Agents*, 160–167.

Michaud, F.; Audet, J.; Létourneau, D.; Lussier, L.; Théberge-Turmel, C.; and Caron, S. 2001. Experiences with an autonomous robot attending the AAAI conference. *IEEE Intelligent Systems* 16(5):23–29.

Michaud, F.; Létourneau, D.; Arsenault, M.; Bergeron, Y.; Cadrin, R.; Gagnon, F.; Legault, M.-A.; Millette, M.; Paré, J.-F.; Tremblay, M.-C.; Lepage, P.; Morin, Y.; and Caron, S. 2005. Multi-modal locomotion robotic platform using leg-track-wheel articulations. *Autonomous Robots, Special Issue on Unconventional Robotic Mobility* 18(2):137–156.

Michaud, F.; Brosseau, Y.; Côté, C.; Létourneau, D.; Moisan, P.; Ponchon, A.; Raievsky, C.; Valin, J.-M.; Beaudry, É.; and Kabanza, F. 2005. Modularity and integration in the design of a socially interactive robot. In *Proceedings IEEE International Workshop on Robot and Human Interactive Communication*, 172–177.

Michaud, F. 2002. EMIB – Computational architecture based on emotion and motivation for intentional selection and configuration of behaviour-producing modules. *Cognitive Science Quaterly, Special Issue on Desires, Goals, Intentions, and Values: Computational Architectures* 3-4:340–361.

Montemerlo, M.; Roy, N.; and Thrun, S. 2003. Perspectives on standardization in mobile robot programming: The Carnegie Mellon navigation (CARMEN) toolkit. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2436–2441.

Parker, L. E. 1998. ALLIANCE: An architecture for fault tolerant multirobot cooperation. *IEEE Transactions on Robotics and Automation* 14(2):220–240.

Perez, P.; Hue, C.; Vermaak, J.; and Gangnet, M. 2002. Color-based probabilistic tracking. In *Proceedings of the European Conference on Computer Vision*, 661–675.

Schmidt, D. C. 1994. ACE: An object-oriented framework for developing distributed applications. In *Proceedings USENIX C++ Technical Conference*.

Schultz, A. C. 2001. The 2000 AAAI mobile robot competitions and exhibition. *AI Magazine* 22(1):67–72.

Simmons, R., and Apfelbaum, D. 1998. A task description language for robot control. In *Proceedings Conference on Intelligent Robotics and Systems*.

Simmons, R.; Goldberg, D.; Goode, A.; Montemerlo, M.; Roy, N.; Sellner, B.; Urmson, C.; Schultz, A.; Abramson, M.; Adams, W.; Atrash, A.; Bugajska, M.; Coblenz, M.; MacMahon, M.; Perzanowski, D.; Horswill, I.; Zubek, R.; Kortenkamp, D.; Wolfe, B.; Milam, T.; and Maxwell, B. 2003. Grace : An autonomous robot for the AAAI robot challenge. *AI Magazine* 24(2):51–72.

Smart, W. D.; Dixon, M.; Melchior, N.; Tucek, J.; and Srinivas, A. 2003. Lewis the graduate student: An entry in the AAAI robot challenge. Technical report, AAAI Workshop on Mobile Robot Competition. p. 46-51.

Smart, W. D.; Tejada, S.; Maxwell, B.; Stroupe, A.; Casper, J.; Jacoff, A.; Yanco, H.; and Bugajska, M. 2005. The 2004 mobile robot competition and exhibition. *AI Magazine* 26(2):25–35.

Stoytchev, A., and Arkin, R. 2004. Incorporating motivation in a hybrid robot architecture. *Journal of Advanced Computational Intelligence and Intelligent Informatics* 8(3):269–274.

Taylor, P. 1999. The Festival speech architecture. URL: http://www.cstr.ed.ac.uk/projects/festival/.

Valin, J.-M.; Michaud, F.; Hadjou, B.; and Rouat, J. 2004. Localization of simultaneous moving sound sources for mobile robot using a frequency-domaine steered beamformer approach. In *Proceedings IEEE International Conference on Robotics and Automation*, 1033–1038.

Valin, J.-M.; Michaud, F.; and Rouat, J. 2006. Robust 3D localization and tracking of sound sources using beamforming and particle filtering. In *Proceedings International Conference on Audio, Speech and Signal Processing*, 221–224.

Valin, J.-M.; Rouat, J.; and Michaud, F. 2004. Enhanced robot audition based on microphone array source separation with post-filter. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2123–2128.

Vaughan, R. T.; Gerkey, B. P.; and Howard, A. 2003. On device abstractions for portable, reusable robot code. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2421–2427.

Yanco, H., and Balch, T. 2003. The AAAI-2002 mobile robot competition and exhibition. *AI Magazine* 24(1):45–50.

Zhao, Y. 2003. A model of computation with push and pull processing. Master's thesis, University of California at Berkeley, Department of Electrical Engineering and Computer Science.