REALITY & VIRTUAL REALITY IN MOBILE ROBOTICS

B.R. Duffy, G.M.P. O'Hare, R.P.S. O'Donoghue, C.F.B. Rooney, R.W Collier

PRISM Laboratory, Dept. of Computer Science, University College Dublin (UCD), Belfield, Dublin 4, Ireland
{Brian.Duffy, Gregory.OHare, Ruadhan.ODonoghue, Colm.Rooney, Rem.Collier}@ucd.ie

Abstract. This paper advocates the application of VRML in the visualisation of social robotic behaviour. We present the Virtual Reality Workbench, which via the Social Robot Architecture integrates the key elements of Virtual Reality and robotics in a coherent and systematic manner. To support this, we deliver a development environment, Agent Factory, which facilitates rapid prototyping and visualisation of social robot communities.

Introduction

The use of Virtual Reality has developed from the traditional game playing metaphor to application as diverse as e-commerce and virtual communities. This paper proposes the use of VRML in the visualisation of intelligent agent communities. We present the Virtual Robotic Workbench, which by monitoring the mental states of agents in a multi-agent system, can display and update a VRML representation of the agents environment. An obvious application of such technology is robotics.

The *Social Robot Architecture* (SRA) combines reactivity, deliberation and social ability to enable robots to deal competently with complex, dynamic environments. We demonstrate how the Virtual Robotic Workbench can present a virtual window into the robots' environment, allowing for remote experimentation and thus providing insights into the inconsistencies between a robot's mental image of an environment and the physical counterpart. Figure 1, below, presents an architecture, which seamlessly integrates, real world robots, multi-agent development tools, and VRML visualisation tools into a coherent whole.

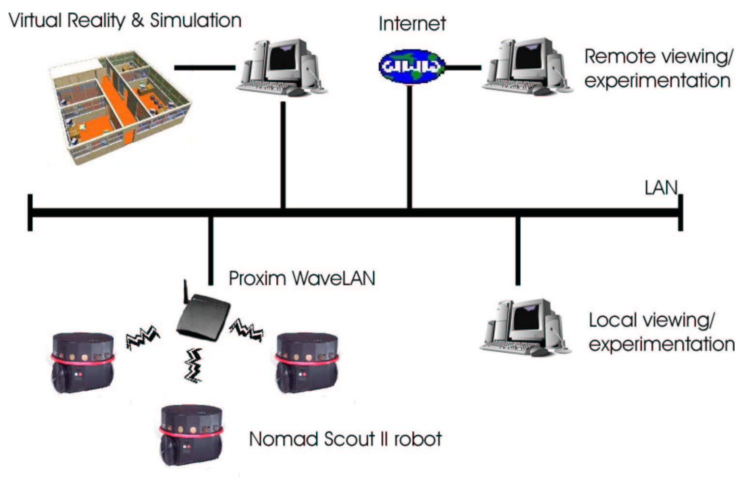

Virtual Reality & Simulation
Internet
Remote viewing/ experimentation
LAN
Proxim WaveLAN
Local viewing/ experimentation
Nomad Scout II robot

Fig. 1. Social Robot: The Coherent Whole

Virtual Reality

VRML (Virtual Reality Modelling Language) is a recent advancement in Internet technologies [TSW+96]. VRML allows for the development of dynamic 3D worlds, which can be viewed through a web browser. VRML 2.0 provides the user with many tools facilitating scene animation and scene update based on user input. Some applications of the technology are as follows:

Virtual communities

A substantial area of VRML research is the provision of software tools which enable user immersion in a *connected virtual community*. Several such VR toolkits are available which enable the creation and deployment of multi-user interactive virtual communities. One such tool is VNET [VNET], which is a VRML and Java based 3D Virtual World client/server system, freely available under the GNU Public Licence.

E-commerce

ViSA (Virtual Shopping Agent Architecture) [OO99] is representative of a new generation of e-commerce system that enables the user to truly enter the retail arena and participate in the virtual shopping experience. The ViSA System offers: immersion within a 3-D shopping environment and virtual community of shoppers; intelligent assistance in all stages of product procurement; contextualised and personalised shopping experience for individual shoppers.

Robotics

Many exemplary systems demonstrate the use of VR for simulation and visualisation within robotics. Here we consider but four subsystems.

Workspace® [Workspace] from Robot Simulations Ltd. is one such tool. It allows the user to model and display simulations of workcell layouts involving conveyors, AGV's and the evaluation of their performance. System programming can take place before robot installation or while an existing workcell is in operation. Complex layouts can be tested in complete safety either in an office environment or on the shop floor. Simulation highlights potential problems before they happen, allowing quick and easy modifications.

The RHINO-project [BCF+98] a joint project between the Institut für Informatik III of the Rheinische Friedrich-Wilhelms-Universität Bonn and the Carnegie Mellon University (USA). The central scientific goal of the RHINO project is the analysis and synthesis of computer software that can learn from experience. The team believes an essential aspect of future computer software will be the ability to flexibly adapt to changes, without human intervention. The ability to learn could soon enable robots like RHINO to perform complex tasks, such as transportation and delivery, tours through buildings, cleaning, inspection, and maintenance.
While giving tours in the Deutsches Museum, RHINO can be observed and even tele-operated through the Internet via a virtual reality medium. RHINO will make available on-line camera images recorded in the museum.

The DLR VRML 2.0 Robot [Roh97] is a robot simulation system that enables the telemanipulation of real robots via WWW. It provides a detailed graphical model of the robot that can be manipulated intuitively using the mouse. Basic features are forward and backward kinematics with various interaction possibilities for motion in joint space or Cartesian space. Routes can be programmed and edited. The software also handles the connection and control of various tools.
It is anticipated the VRML interface will support the operation one of the KUKA robots.

RESOLV Project [LGL+98] provides a working model of its Autonomous Environmental Sensor for Telepresence (AEST). This robot tours the inside of a building and automatically creates a 3-D map of the interior compete with surface texture information. The 3-D reconstruction module produces two models; a geometrical model suitable for conventional CAD systems, and another composed of triangular meshes suited to graphical visualisation. Both representation use VRML format and are thus viewable with any WWW browser.

In the next sections we present the Social Robot Architecture and the use of Virtual Reality for the visualisation of social robot communities as opposed to individual robots.

The Social Robot Architecture

The *Social Robot* Architecture aims at achieving team building and collaborative behaviour through the judicious synthesis of the reactive model with that of the deliberative model. The architecture (figure 2) is comprised of four discrete layers: physical, reactive, deliberative developed using Agent Factory, and social.

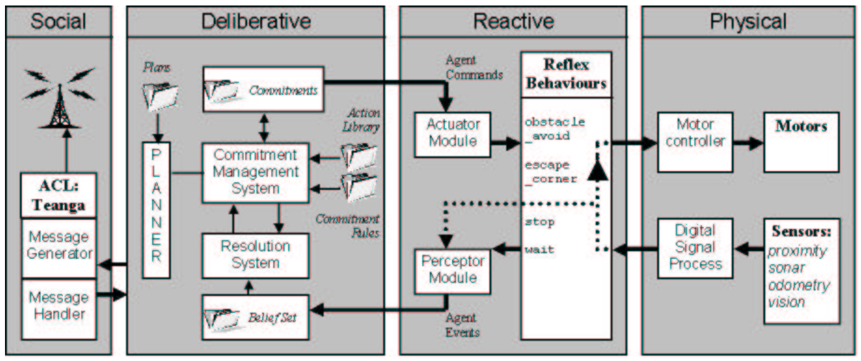


Fig. 2. The Social Robot architecture: The Robot Agent

Physical: Robots in terms of this research may take the form of either that of a physical entity, specifically the Nomad Scout II or a simulated entity (Nomadic Technologies XRDev robot).
Reactive: A series of fundamental reflex behaviours are implemented at this level. The sensory information is processed resulting in clear *agent_events* and communicated to the deliberative level. *Agent_commands* are received from the deliberative layer.
Deliberative: This comprises of a Belief Desire Intention (BDI) [RG91] architecture developed through Agent Factory. The perception process deals with converting agent events into beliefs and adding them to the belief set providing the agent with an up to date model of its current perceived situation and results in the agents' commitments being updated accordingly. Pre-existing commitments are analysed and those pertaining to the current time frame honoured resulting in either a communicative act being sent to the social level or a physical act being passed to the actuators via the reactive level.
Social: Our agents interact via an Agent Communication Language (ACL), entitled Teanga.

More detailed information on *The Social Robot Architecture* and its applications are given in [DCO+99].

Agent Factory

Agent Factory has been developed to facilitate the rapid prototyping of Multi-Agent Systems. The system offers an integrated toolset that supports the developer in the instantiation of generic *agent structures* that are subsequently utilised by a pre-packaged agent interpreter that delivers the BDI machinery. Other system tools support interface customisation and agent community visualisation.

In creating an agent community three system components must be interwoven, those of *agents*, a *world* and a *scheduler*.

The *agent* is the core computational unit underpining Agent Factory, it combines a series of attributes that represent and support the Mental State model of an agent, a set of methods (the actuators), a set of perceptors, an Agent Communication Language (ACL), and a Commitment Revision Strategy. This design is then executed using a generic *Agent Interpreter*. The *scheduler* controls execution of the community, using an algorithm that exploits parallelism where possible. Finally, the *world interface* acts as a medium between the problem domain, the community it is being developed for, and the other components of the Agent Factory System.

The creation of an agent community is facilitated by the *Agent Factory Development Environment*, which provides a *Component Library* and a selection of tools for the rapid prototyping of agent communities. The Component Library is built from Component Design Hierarchies (CDH) that extend the standard Object Hierarchies in the OOP Paradigm.

The *Agent Factory Run-Time Environment* provides the support necessary for the release of a completed Multi-Agent System. This environment comprises of a *Run-time Server* and an *Agent Interpreter*. The Run-time Server offers two main services: access to non-agent components of the system (a controller & some worlds), and a set of tools for viewing and interacting with the agent community. The Agent Interpreter provides the mechanisms by which the agents may be executed and visualised. Access to these environments is provided both locally through Graphical User Interfaces (GUIs) and remotely through the World Wide Web (WWW) via a purpose built *Web Server*.

The Agent Factory System has been discussed more completely elsewhere in the literature [CO99].

The Virtual Robotic Workbench

One of the key tenants of our research has been the provision of multiple views of multiple robot systems. The primary view is the physical perspective of the Nomad Scout II's navigating the physical world. The secondary, more abstract view, is a virtual reality perspective provided via the Virtual Robotic Workbench, which delivers a 3-D VRML world via the Internet (figure 3).

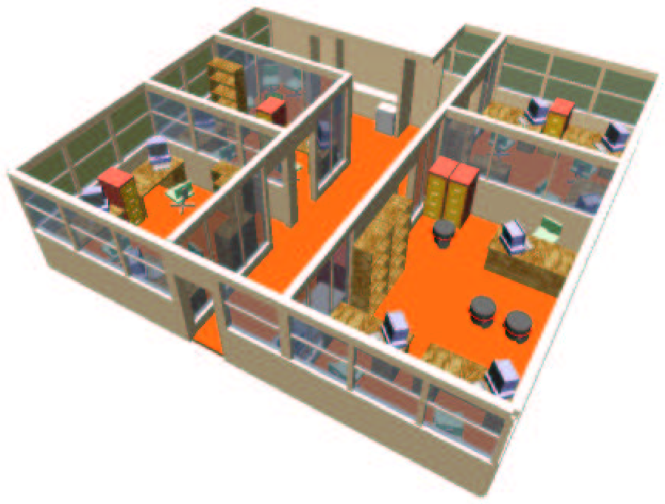


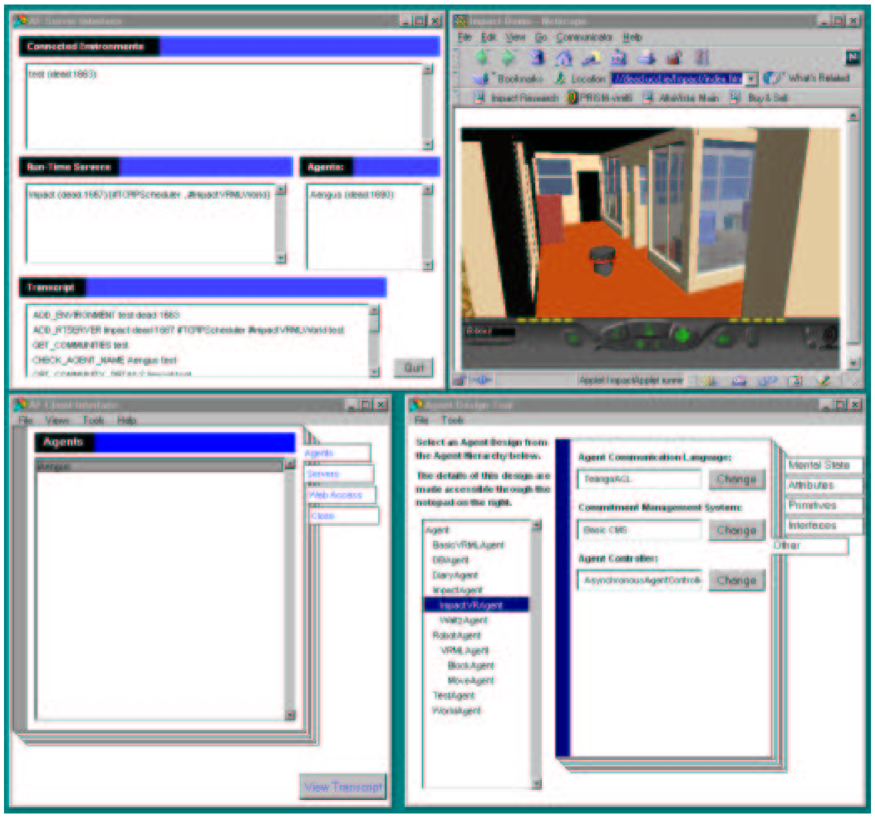
Fig. 3. Virtual reality view of the IMPACT research environment

Herein we harness the advantages of using virtual environments, by directly relating virtuality and reality in a seamless manner. This permits multiple views, information hiding and abstraction, system interaction, and behaviour scrutiny via snapshots and recordings. A *Virtual Robotic Workbench* provides a medium through which researchers can design robot experiments remotely and without recourse to the purchase of expensive robotic entities in the first instance. Researchers can articulate their experiments across a Java interface whereby they tune certain key workbench parameters, namely:

- The Customisation of the Environment
  - The Number of Robots;
  - The world within which they are to be situated;
- The Customisation of the Robots
  - Their Name;
  - The visualisation avatar associated with each robot;
  - Mapping virtual robots to physical robots;
  - The behaviours ascribed to a given robot;
  - Their Initial Location within the selected world;
- The Task
  - The selection of the shared goal;

Figure 4 depicts the Virtual Robotic Workbench interface by which these parameters are specified. Once the experiment has been crafted the subsequent activation results in the observable behaviour of the robots across any suitably configured web browser. The behaviour of the robots is governed by the Social Robot Architecture with the higher level functions directly furnished through Agent Factory. As such, the navigation and behaviour of the robotic avatars can be seen by utilising the Agent

Factory Visualiser. Detailed treatment of the Agent Factory Visualiser is presented elsewhere in the literature [OCC+98].



```
ADD_ENVIRONMENT test dead 1683
ADD_RTSERVER Impact dead 1687 #TCPIPScheduler #ImpactVRMLWorld test
GET_COMMUNITIES test
CHECK_AGENT_NAME Aengus test
```

Fig. 4. The Virtual Robotic Workbench

The benefits of the application of virtual reality to robotic experimentation are manyfold. Perhaps, the most obvious is that it allows for remote behaviour observation from multiple perspectives.

The Virtual Reality Visualiser

Within the Social Robot Architecture, the Run-Time Server performs the particular role of housing the machinery for the delivery of the *Virtual Robotics Workbench*. This workbench facilitates the remote specification and subsequent observation of Social Robotic experiments. Within the context of Agent Factory, the Virtual Robotic Workbench utilises an existing tool, the Agent Factory Visualiser (AFV) to present a 3D view of the agents. An earlier version of this tool is discussed in [OCC+98]. The

Visualiser tool facilitates the presentation of Virtual Reality views of the agent community commissioning standard web browsing technologies. It is comprised of three components:

- The AFV Server: This component is plugged into a Run-Time Server and acts as a proxy server between the agents and the AFV Clients. The server maintains lists of objects depicted in the Virtual Reality Scene.
- The AFV Client: This is a Java Applet embedded into a HTML page. The applet is responsible for the maintenance of a current 3D view of the agent community. This page is viewed within a standard Web browser that has support for the VRML.
- A Web Server: The third component is a standard web server. This is required to host the AFV Client in order that it may be downloaded as necessary.

We now discuss the interplay between these components. The key component of this system is the AFV Client, which is responsible for the delivery of the VR view of the agent community. As indicated earlier, the AFV Client is a combination of standard web technologies. A client interface is comprised of a VRML Scene which is updated through a Java Applet. Upon loading the AFV Client, it is the responsibility of the user to connect the client to a suitable AFV Server. After the AFV Client and Server are connected, the next step is for the Server to send the URL of the VRML scene to be loaded (i.e. the agents arena).
The update of the VRML Scene is achieved through an interface provided as standard within the VRML97 specification. This interface is called the External Authoring Interface (EAI). The purpose of the EAI is to enable the modification of VRML scenes through a pre-defined library of functions. These functions are made available through a collection of Java classes that are used within the AFV Client. Available functionality includes the ability to create additional VRML objects within the VRML scene, and to modify parts of the VRML scene.
Henceforth, those agents that are to be displayed within the VRML scene are appropriately configured, and subsequently appear. This configuration necessitates permits the subsequent *tapping* of the *agent_event* queue as described later. From this point on, the agents' movements are mirrored in the VRML Scene.

Figure 5 illustrates how an agent's virtual position may be updated based upon its physical position. *Agent_events* are triggered within the reactive layer of the SRA and transmitted to the deliberative level. These events relate to the detection of landmarks such as corners of a room, doorways etc. For example, the detection of a landmark results in system messages such as the following being sent: '`AGENT-EVENT LANDMARK corner AT 5.32 4.14`'. This states that a corner was detected at position (5.32, 4.14) in the agent's local coordinate system.

Each agent handles *agent_events* about that event. However, the update of the Virtual Robots' position does not occur through any deliberative action on the part of the agent. Instead, a *tap* is placed upon the event queue, which listens for a landmark *agent_event*. Upon detection of this event, the coordinates are taken and converted to the absolute coordinate system used by the Virtual Agents. Subsequently a system message informs the Visualiser to update the position of a given virtual robot. Such a message takes the form of: ‘`MOVE_OBJECT <name> <x> <y> <orientation>`’ informing the Visualiser to move the object with the given name to position `(<x>, <y>)`, and to rotate it to the given orientation. The AFV Server receives this message and propagates it to all the AFV Clients currently connected. These clients receive the message and update their VRML scenes accordingly.

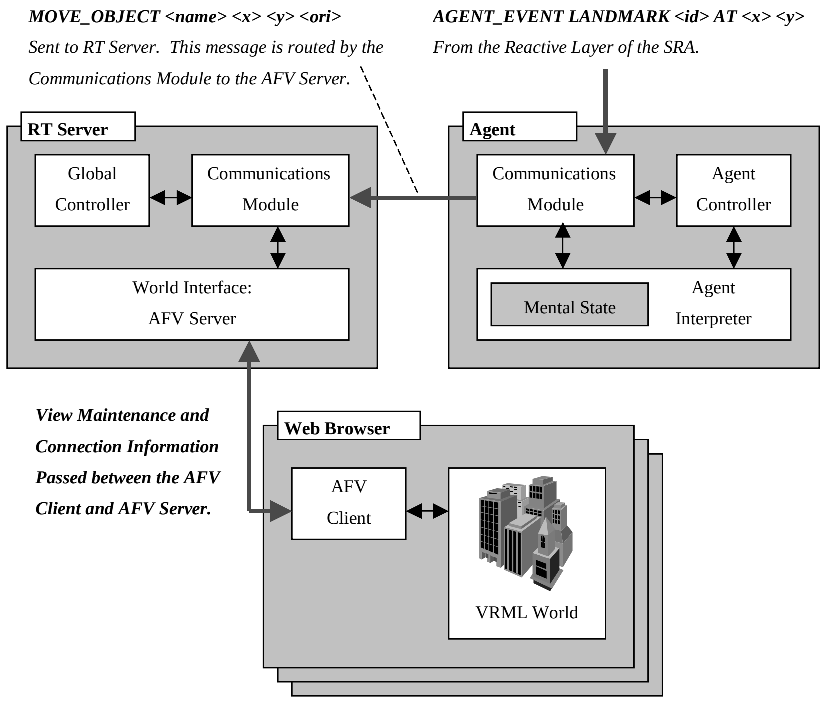

MOVE_OBJECT <name> <x> <y> <ori>
Sent to RT Server. This message is routed by the Communications Module to the AFV Server.
AGENT_EVENT LANDMARK <id> AT <x> <y>
From the Reactive Layer of the SRA.
RT Server
Global Controller
Communications Module
World Interface: AFV Server
Agent
Communications Module
Agent Controller
Mental State
Agent Interpreter
View Maintenance and Connection Information Passed between the AFV Client and AFV Server.
Web Browser
AFV Client
VRML World

Fig. 5. Position updating within the VR Workbench

This results in a real-time link between the real robot and its virtual counterpart. In the next section we examine how to maintain a sufficient degree of positional accuracy to make this link viable.

Experimentation

Robot experiments are characterised by firstly selecting a world, subsequently situating robot(s) in this world, and finally ascribing behaviours to these robots. In this approach the odometric information sent by the real-world robot to Agent Factory is supplemented with sonar and visual information that may indicate detected environmental features, and relative distances from these features. Such sensor fusion enables the robot avatar position update not only by mirroring the uncertain real-world coordinate position updates, but also by matching current sensory information with the VRML world to reduce the uncertainty of the robot's position.

Of course, we do not aspire to have a completely accurate real-time synchronisation between real and virtual robots. The VRML view is primarily a visualization tool and therefore it is sufficient to update and recalibrate the virtual world stochastically.

The problems associated with obtaining robot position accurately without the use of tailored environments (by providing landmarks etc.) are rife [LMK+97] [Saf96] [DN99] [SC93] [RL97] [HR96]. Robot research has extensively documented problems associated with the cumulative positional error in the robots' own odometric sensor readings render such sensors inadequate as exclusive sources of long-term positional information. Given the robots' internal position information (x_r,y_r) the problem is to ensure that the robot avatar is in *approximately* the corresponding location (x_v,y_v) in the virtual world (see figure 8).

The first step in achieving this synchronization task is to send the actual robot position update to Agent Factory. This event takes the form of `position_update Agent(x1,y1)`. This coordinate is based on the robots internal odometry system and may therefore harbour inaccuracies. Based on this information the robot avatar position is provisionally updated. However, to avoid the certain inconsistencies that would arise between the two worlds, we employ several recalibration techniques, which operate on both a local and a global basis.

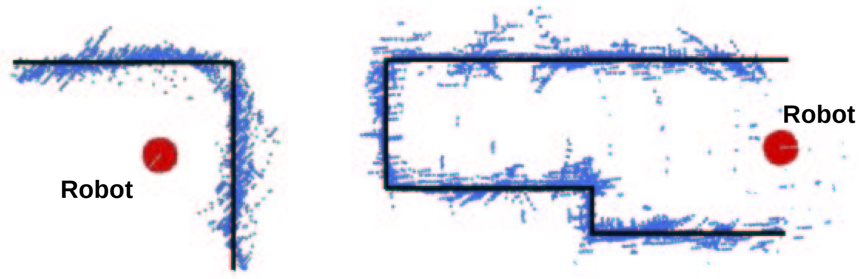
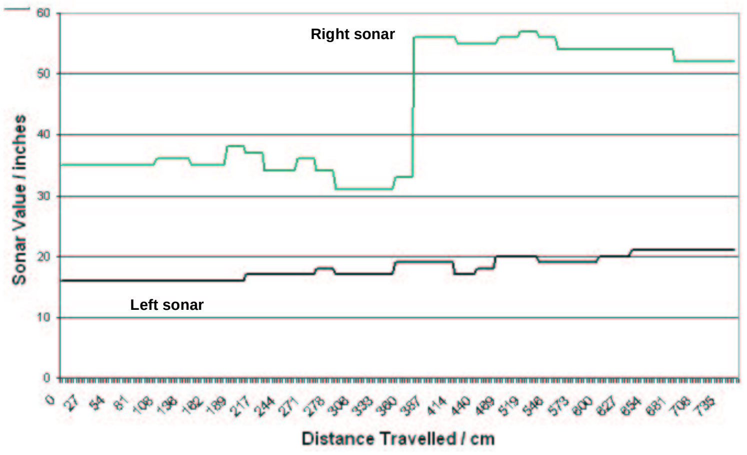


Fig. 6. (a) Corner detection, (b) Wall detections

The world is comprised of predefined locations. On passage between these locations, an event is sent to Agent Factory. For instance, as the real robot passes through a door (recognized by both visual analysis and by its characteristic sonar signature) an event is sent such as *door(bui,sonar_number,distance)* which corresponds to Bui's belief that there exists a door at distance *distance* from sonar *sonar_number*. On the occurance of such an event, we may update the avatar position accordingly, by zeroing both the real and the virtual position coordinates. Global recalibration is thus achieved.

On entering a new location, we can retrieve the robot's position with a certain degree of accuracy. We may carry out local recalibration based on the various features which may be detected within in a specific location. Common features that can be recognised using sonar or vision (or both) include doors, walls, primary corners, desks etc. Figure 6 shows corner and wall detection from sonar data. These features may be

classed as primary location cues towards achieving local recalibration for accurate visualization through the VR medium. This results in a globally relevant perspective of the environment through VR. Having recognized such a feature, and given that we have an estimate of true robot position, we may update the position estimate by *fitting* the detected feature to the corresponding feature in the VRML model.
Here we merely present initial feature extraction results.



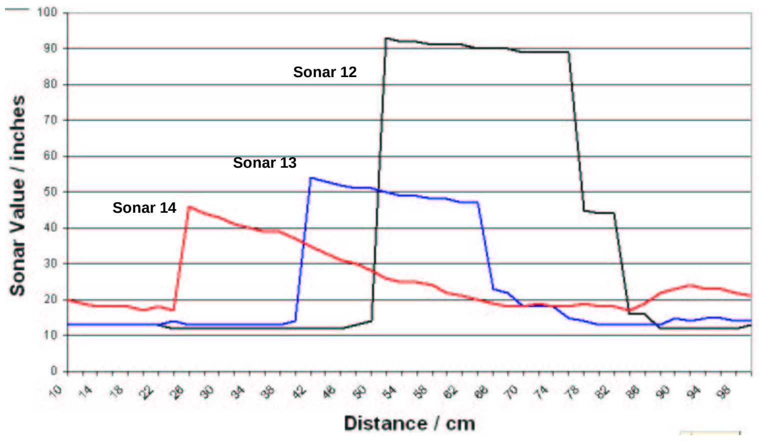


Fig. 7. Sonar signatures: (a) corridor; (b) door

Figure 7(a) graphs the sonar readings from the left and right sides of the robot as it travels parallel with the corridor walls. Figure 8(a) depicts the (real-world) corridor in which a robot experiments was performed, while figure 8(b) depicts the VRML representation of this corridor. The relationship between the graph in figure 7(a), and the views of the corridor should be obvious. The rather large jump in the graph of the *right sonar* corresponds to the increase in distance from the right wall as the robot passes the filing cabinet on *its* right hand side, whereas the flat plot of the *left sonar* corresponds to the straight wall the robot was travelling parallel with. It is relatively easy to fit a line to the plot of the left sonar, and extract the event `wall_from(`x_1`,` y_1`,` x_2`,` y_2`, sonar_number, distance)`, where x_1,y_1 and x_2,y_2 represent the perceived start and end points of the wall, which is distance *distance* from sonar *sonar_number*. In most cases the task of matching this to the VRML representation of this wall is feasible, and having achieved this the robot position and orientation can be fine-tuned.

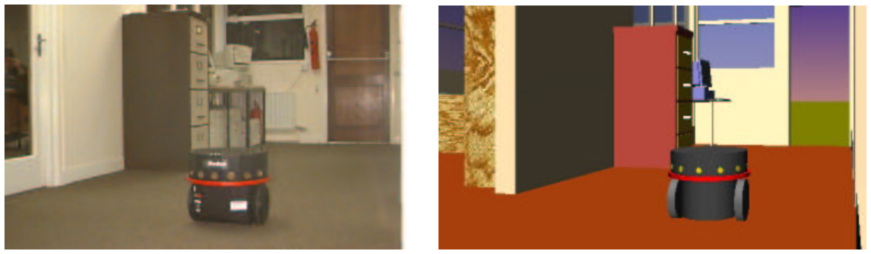


Fig. 8. The virtual reality and physical reality in parallel

Figure 7(b) is a graph of the readings from three of the robot's sonar as it passes an open door. This sonar footprint is characteristic of an open door. First sonar 14[1] jumps as the free-space in the new room is detected. Next sonar 13 makes a similar jump, and finally sonar 12, the side sonar jumps up too, indicating that the door is directly to the side of the robot. While the value from sonar 12 jumps, the values from sonar 13 and sonar 14 fall off again. This *sonar signature* reappears each time a door is passed. It differs from a corner sonar typical corner signature in that sonar 13 and sonar 14 decrease again in the door case. In addition to recognising the door through its sonar signature, we have successfully employed visual analysis for door recognition to provide corroborative evidence of the presence of a door. A library of sonar signatures can therefore be utilised for feature recognition.

Stochastic Synchronisation Between Worlds

The trade off between perception and representation has been extensively documented within AI literature. In a conventional sense mobile robots sense (perceive) their environment as they navigate through it. The emphasis is clearly upon perception rather than representation. Within the Social Robot Architecture we redefine this trade off. Perceptions are transformed into beliefs about the environment within which the robot exists. As such, perceptions underpin a subsequent thin representation of the environment. Individual robot perceptions and corresponding beliefs may subsequently be propagated to fellow robots in keeping with the social nature of our architecture. Thus, given the inherently myopic nature of our Nomads,

[1] Sonar 12 is directly on the side perpendicular to the wall with sonars 13 22.5° and 14 45° to the front respectively

an inexact and incomplete representation of the environment is derived through the augmentation of other partial views communicated by fellow robots. Currently, Agent Factory agents are gullible and as such beliefs are received in good faith and directly incorporated into the mental state.
When we contrast this with our virtual robots then the perception representation trade off is somewhat different. Our virtual robots have currently no perception capability. They do however have a fairly rich representation of the virtual environment, a direct replica of its real counterpart. By taking receipt of Agent Factory events the world is refreshed to reflect robot transformations. A vast amount of sensory data is culled by the Nomads and filtered through key feature footprints, which encode patterns of sonar data synonymous with particular features. Upon recognition the associated event is generated, dispatched to Agent Factory, and the next cycle the belief update effected. Subsequent to this the Virtual world is updated. As such a synchronisation of sorts is achieved. Rather than this being continuous it is stochastic in nature. There are a multitude of problems associated with achieving the former. For our purposes, the approximate depiction of robot behaviour in virtual robot experiments viewed and expressed through the medium of the Internet, the latter proves adequate.
To date we have concentrated upon the flow of information from the real world to the virtual. A counter flow of data could be harnessed and as yet we have not truly investigated this. Given the richer representational model held by the virtual environment (i.e. a building comprised of floors, in turn comprised of rooms interconnected by corridors and populated with objects and robots) upon recalibration of the robot position, beliefs about the immediate location could be funnelled back to the robot agents. For example the existence of a door to the immediate left. This bi-directional information flow seems to offer mutual benefit.

Discussion and Conclusions

Within this paper we have characterised our research on Social Robotics and specifically the interplay between virtual and real environments for Social Robot experimentation. The creation of the Virtual Robotic Workbench offers a medium through which cost effective robot experiments may be conducted. In particular we access robot behaviours in a variety of virtual buildings.

Within the context of this work we regard robots as active intelligent objects. We would envisage that smart buildings would be comprised of a collection of intelligent interacting components some of which might be static (Heating Subsystem, Lighting Subsystem, Entry Controller) or dynamic (robots, AGVs, Lifts). Robot agents in the context of the Social Robot Architecture permit effective collaboration through their social nature. We advocate an agent community for the delivery of the intelligence necessitated in such smart buildings. Thus collaboration between a lift agent and a robot agent and entry control agent could prove crucial for floor migration. Collaboration between the lighting control agent and a robot agent could facilitate lighting adjustment for onboard frame grabbing facilities for visual perception.

The application of social robotics within the context of smart environments and buildings is endless. Two possible scenarios are, the self-cleaning building and mail delivery robot teams. At appropriate times a team of social cleaning robots could collaboratively effect an exhaustive vacuuming activity. Alternately a team of mail delivery robots could plan delivery schedules for internal mail dispatch.

Acknowledgements

We gratefully acknowledge the support of Enterprise Ireland through grant No.'s SC/96/621 COMEDY and SP/97/08 IMPACT (Initiative for Mobile Programmable Agent-based Computing Technologies).

References

[BCF+98] Burgard, W., Cremers, A.B., Fox, D., Hähnel, D., Lakemeyer, G., Schulz, D., Steiner, W., and Thrun, S. "The Interactive Museum Tour-Guide Robot", Proceedings of the 15th National Conference on Artificial Intelligence (AAAI'98), Madison, Wisconsin, 1998

[CO99] Collier, R.W., O'Hare, G.M.P., "Agent Factory: A Revised Agent Prototyping Environment", 10th Irish Conference on Artificial Intelligence & Cognitive Science, 1-3 Sept., 1999 University College Cork, Ireland

[Col96] Collier, R. "The realisation of Agent Factory: An environment for the rapid prototyping of intelligent agents", M.Phil., Univ. of Manchester, 1996

[DCO+99] Duffy, B.R., Collier, R.W., O'Hare, G. M. P., Rooney, C.F.B., O'Donoghue, R.P.S. "SOCIAL ROBOTICS: Reality and Virtuality in Agent-Based Robotics", in Proceedings of Bar-Ilan Symposium on the Foundations of Artificial Intelligence: Bridging Theory and Practice (BISFAI), 1999.

[DN99] Duckett, T., Nehmzow, U., "Self-localisation and autonomous navigation by a mobile robot", Proc. Towards Intelligent Mobile Robots 99 (Bristol). Tech. Report Series, Dept. Computer Science, Manchester University, Rep. No. UMCS-99-3-1. 1999.

[HR96] Harris, K.D. and Recce, M. (1996) Localisation and goal-directed behaviour for a mobile robot using place cells in Proc. Sintra Workshop on Spatiotemporal Models in Biological and Artificial Systems, IOS Press, Amesterdam.

[LGL+98] Leevers, D., Gil, P., Lopes, F. M., Pereira, J., Castro, J., Gomes-Mota, J., Ribeiro, M. I., Gonçalves, J. G. M., Sequeira, V., Wolfart, E., Dupourque , V., Santos, V., Butterfield, S., Hogg, D., Ng, Kia. "An Autonomous Sensor for 3D Reconstruction" 3rd European Conference on Multimedia Applications, Services and Techniques Berlin-Germany, 26-28 May 1998 ECMAST'98

[LMK+97] Lambrinos, D., Maris, M., Kobayashi, H., Labhart, T., Pfeifer, R., and Wehner, R. (1997). "An autonomous agent navigating with a polarized light compass", Adaptive Behavior, 6, 131- 161.

[OCC+98] O'Hare, G.M.P., Collier, R., Conlon, J. and Abbas, S., "Agent Factory: An Environment for Constructing and Visualising Agent Communities", Proc. AICS98, 1998

[OJ96] O'Hare, G.M.P., Jennings, N.R., (Editors.), Foundations of Distributed Artificial Intelligence, Wiley Interscience Pub., New York, 1996, 296 pages. ISBN 0-471-00675

[OO99] O' Sullivan, G., O' Hare, G.M.P., Coughlan, C., 1999. "ViSA: Virtual Shopping Agent Architecture" Submitted to 8th Euromicro Workshop on Parallel and Distributed Processing Rhodos, Greece, January 2000.

[RG91] Rao, A.S., Georgeff, M.P., “Modelling Rational Agents within a BDI Architecture”, Prin. Of Knowl. Rep. & Reas., San Mateo, CA., 1991

[Roh97] Rohrmeier, M. “Telemanipulation eines Roboters via Internet mittels VRML2.0 und Java”, Diplomarbeit, Institut für Robotik und Systemdynamik, Technische Universität München

[RL97] P.Ramos, F. Lobo Pereira, "Localisation system for an Autonomous Mobile Platform" Proceedings of the ISIE'97 - IEEE International Symposium on Industrial Electronics, Guimarães, Portugal, 7th-11th July, 1997.

[Saf96] Saffiotti, A., “Robot navigation under approximate self-localisation”, *Robotics and Manufacturing Vol.6, Procs. 6th ISRAM Symposium, P589-594, 1996.*

[SC93] Scheile, B., and Crowley, J. L., "Certainty Grids: Perception and Localisation for a Mobile Robot", IRS ‘93, Zakopane, Poland, July 1993.

[TSW+96] Tittel, E., Scott, C., Wolfe, P., Sanders, C., (1996). Building VRML Worlds. Obsborne ISBN 0-07-882233-5, July 1996.

[VNET] VNET: http://ariadne.iz.net/~jeffs/vnet/

[Workspace] http://www.rosl.com/brochure.htm