An Human-Computer Interface for a Robotic System

J. Madeiras Pereira¹, P. Gil¹ and F. Lopes¹

¹Instituto de Engenharia de Sistemas e de Computadores, Lisboa, Portugal

Abstract

The ACTS Project RESOLV aims to develop an integrated approach to the construction of textured 3D scene models from laser range data and visual images. This approach has been realised within a prototype device for 3D reconstruction, known as the AEST—an autonomous mobile robot carrying a scanning laser range finder and a video camera. The AEST is intended fully automate the creation of indoor virtual environments by navigating automatically between positions at which range data and video images are captured. This vehicle must be remotely operated via a human-computer interface, being the WWW the chosen media for its implementation. The paper introduces this fully integrated multipurpose web-based interface.

1. AEST Software Architecture

The AEST (Autonomous Environmental Sensor for Telepresence) incorporates an autonomous mobile platform with a hybrid sensor, consisting of a scanning laser range finder and a video camera. The system is intended to fully automate the creation of models from the interiors of buildings (e.g. one floor of an office block) or from hostile environments (e.g. within a nuclear plant), by navigating between positions at which data is captured. Each capture point provides a separate model of the world, which is then merged with the others in order to generate a larger and more complete model of the scene [Leevers98, Sequeira98].

Embedded software performs several functions (see Fig.1), including triangulation of the range data and registration of video texture, registration and integration of data acquired from different capture points, and optimal selection of these capture points, ensuring that range data and video texture is acquired for all surfaces at the required resolution.

![Diagram](image)

Figure 1: Software architecture for the AEST

The software architecture of the robotic system [Gil98] is peculiar in terms of its organisation. Although the functionality is thoroughly distributed in different executable clients with well-defined tasks, there was the need to centralise all the shared information in a Host Server (HS). This server two major functions. Firstly, it provides every module with a communications framework which allows them to exchange messages between each other. Secondly, using the created framework, it is responsible for synchronizing and coordinating all the client modules, allowing them to converge into an application that reconstructs the environment surrounding the acquisition platform. It is the HS that starts and terminates all modules, whilst defining the order in which they start and finish their jobs. As well as these two major functions, the HS is also responsible for the manipulation of files before, during and after, a scan session, and for storing the sets of data needed.
2. Human-Computer Interface (HCI)

The HCI is a user interface, which allows the remote control over the Internet of the AEST. It acts as an indispensable part of the task distributed architecture implemented in the RESOLV project, connecting directly with the Host Server. The HCI works by interacting with the user, assembling his requests and sending them over HS, where they will be processed. Then, its actions will include the display of the results or requested data in a clear and understandable fashion. Some functions that may not require some kind of intervention from the HS are processed within the HCI internal architecture. The communication to and from the HS has been developed using asynchronous sockets, thus enabling the messages to be processed through the Socket Manager, just as if the HCI was a regular client (Fig. 2).

![Human-Computer Interface](image)

**Figure 2: The HCI within AEST software architecture**

The HCI design was based on the task-centred approach recommended by Clayton Lewis [Lewis94]. This kind of task orientation resulted in a design centred mainly in a set of primary commands:

- **New Model** – begins a new acquisition session, in order to create a new VRML model;
- **Open Model** – opens a previously saved VRML model;
- **Save Model** – saves the current VRML model;
- **View Model** – enlarges the current VRML model onto the workpad;
- **Extend Model** – enhances the model resolution in specific user-defined areas by processing a new reconstruction in the selected region;
- **Engineering Console** – provides an extensive and extendable panel with functions to be used for expert use of the platform, such as tele-operation, ultrasound readings, platform localisation, parameter settings, etc...
- **Last Iteration** – transmits the command relative to the last iteration of the current acquisition;
- **Stop Robot** – forces an emergency robot stop;

- **Close Application** – closes the application.

Although the commands have set the perspective of the HCI use, the prime directive for its development was to build it in a way to simplify any changes that may become necessary. Minor changes, such as the inclusion of a new command, must be previewed through a flexible design, able to reflect them in any of the elements that compose the interface. To uphold this objective, a modular structure was defined, through the definition of functional areas within the HCI (Figure 3).

![Functional areas of the HCI](image)

**Figure 3: The functional areas of the HCI**

All the commands were positioned in the main interactive area of the interface, the toolbar, in which each button represents a direct order to the application. The workpad is the result of the necessity to create an area with multiple functions and, thus, maximum flexibility. It can display diversified contents, such as the engineering console or the current VRML model. The preview area provides the dynamic contents of the HCI, relatively to what is going on with the acquisition. Two windows, one with a snapshot of the environment being captured, and another with the model as it is at a particular moment. These windows also allow the visualisation of a previously saved VRML model. A message box upholds the necessary feedback to users' requests. Movement in the RESOLV logo implies that the application is processing information, while a static picture corresponds to an idle condition.

In terms of technical approach to the problem, various solutions were adopted. Options taken to ensure optimal...
results include the use of the Netscape Frame Specification, ISAPI (Internet Server Application Programming Interface) and JavaScript.

This work has been carried out as part of the ACTS project RESOLV, involving partners from four countries.

References


3.3.3