New Approach for Interactive Communication Between Distant Places Using Intelligent Space

Péter Szemes, Grégory Audureau, Béla Takarics, Péter Korondi

Budapest University of Technology and Economics
szemes@get.bme.hu

Abstract: Nowadays people can make video conferences at different points of the world with the help of technology. A group of Hungarian artists called Vizuális Művek came up with the idea to create a way to make people feel each other through a brand new video-conference system. The Intelligent Space concept was used as the technological background. In this paper, the system making the video conference more interactive is described.

Keywords: video conference, intelligent space, image processing, edge and skin color detection

1 Introduction

1.1 A Brief History of the Intelligent Space

Hashimoto Lab. in The University of Tokyo has proposed 'Intelligent Space' since 1996. At the beginning it consisted of two sets of vision cameras and computers with a home made 3D tracking software, this was written in C and tcl/tk under Linux. Later, a large-sized video projector (100 inches) was added to the Intelligent Space as an actuator. Mobile robots were located in the Intelligent Space for supporting people as well as for being supported. Vision cameras and computers sets were arranged around an entire room and it changed into the Intelligent Space [1].

1.2 Concept of the Intelligent Space

Intelligence Space is a space (room, corridor or street), which has distributed sensory intelligence (various sensors, such as cameras and microphones with intelligence, haptic devices to manipulate the space) and it is equipped with actuators. Actuators are mainly used to provide information and physical support to the inhabitants. This is done by speakers, screens, pointing devices, switches or

robots and slave devices inside the space. The various devices of sensory intelligence cooperate with each other autonomously, and the whole space has high intelligence. Each agent has sensory intelligence. The intelligent agent has to operate even if the outside environment changes, so it needs to switch its role autonomously. The agent knows its role and can support man. Intelligent Space re-composes the whole space from each agent's sensing information, and returns intuitive and intelligible reactions to man. In this way, Intelligent Space is the space where man and agents can act mutually.

1.3 DIND (Distributed Intelligent Networked Device)

In order to realize the Intelligent Space, sensors are located, which recognize space. However, we cannot change normal environment without considering economical and laborious problems. Moreover, appearance should be considered prudentially. Thus, it should be restricted to the range, which does not bring big influence on the existing environment. Based on these, Distributed Intelligent Network Device (DIND) is proposed (Figure 2), which is composed of three basic elements. The elements are sensor, processor (computer) and communication device. DIND is a small device based on three functions that the dynamic environment, which contains people and robots, is watched by the sensor, information is processed to be known easily by the clients by the processor and the DIND communicates with other DINDs through networks [2].

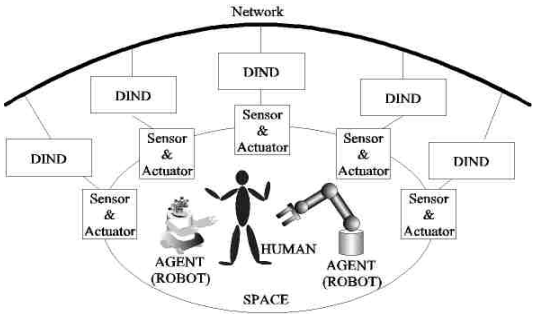

Figure 1
The structure of the DIND

2 The Animation: Exostra Project

Exostra was one of the many kinds of machines used in the theaters of the ancients. Exostra was a machine by means of which things which had been concealed behind the siparium, were pushed or rolled forward from behind it, and thus became visible to the spectators. we do not know much about this machine,

but it was pushed forward upon rollers and was used to exhibit to the eyes of the spectators the results or consequences of such things – e.g., murder or suicide – as could not consistently take place in the proscenium, and were therefore described as having occurred behind the siparium or in the scene.

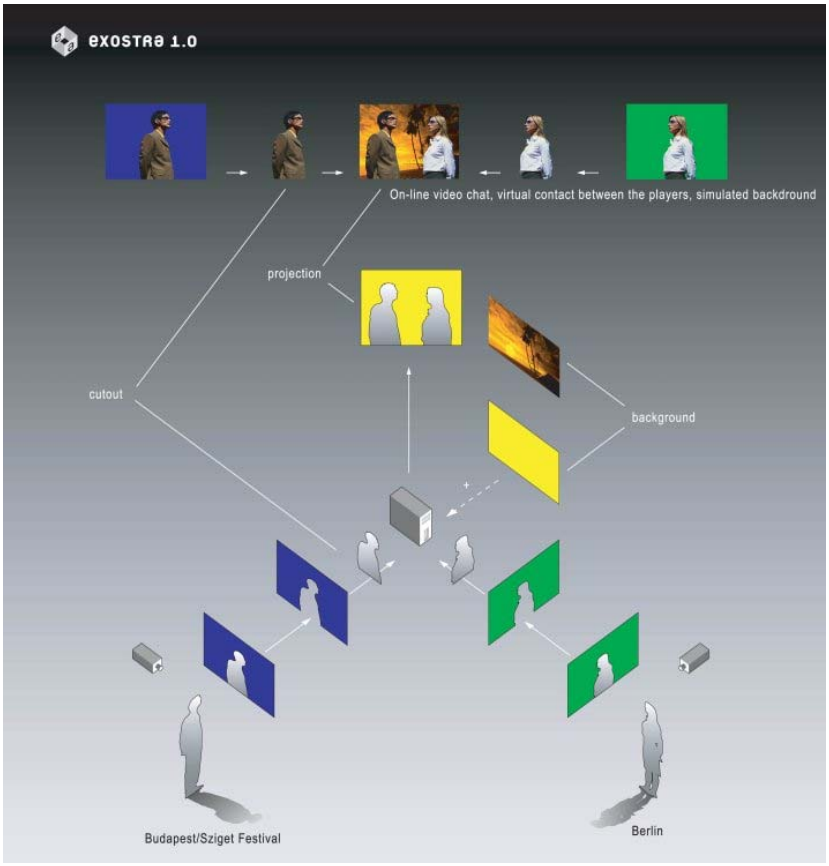

Figure 2
The theorectical structure of the system

2.1 The Finality of the Project

This project has been imagined first by a group of Hungarian artists called Vizuális Művek. Originally, they wanted to create a way to make people feel each other through a brand new video-conference system.

In it, the two protagonists can visually interact with each other. Here is a description of the goal of the project. This interaction must be realized by

composing a new image based on the two original images grabbed by the two cameras used. Of course, this implies both of the users are in front of their own camera.

To do so, several questions have to be answered:

- Which software(s) support(s) the conference?
- How are the users detected on each image?
- How is the final image created?
- How is the “contact” detected?

2.2 The Hardware

To set up this system, we used two computers and two Sony EVI-D30 cameras.

2.3 The Software

Player is a network server for robot control. Running on a robot or a computer, Player provides a clean and simple interface to the robot's or PC's sensors and actuators over the IP network. The client program talks to Player over a TCP socket, reading data from sensors, writing commands to actuators, and configuring devices on the fly.

Player supports a variety of hardware. The original Player platform is the ActivMedia Pioneer 2 family, but several other robots and many common sensors are supported. Player's modular architecture makes it easy to add support for new hardware, and an active user/developer community contributes new drivers. For more information about Player, refer [3].

Player runs on Linux (PC and embedded), Solaris and BSD.

It has been chosen because of its structure which suits perfectly with an Intelligent Space. The two devices connected to Player are the two cameras. A driver already exists in this software for the Sony EVI-D30, however it has to be studied and modified in order to apply the desired image processing to the grabbed images. An other solution consists on creating a brand new driver, able to apply the image processing on the images grabbed by the original video driver.

The following image illustrates the common structure of a Player.

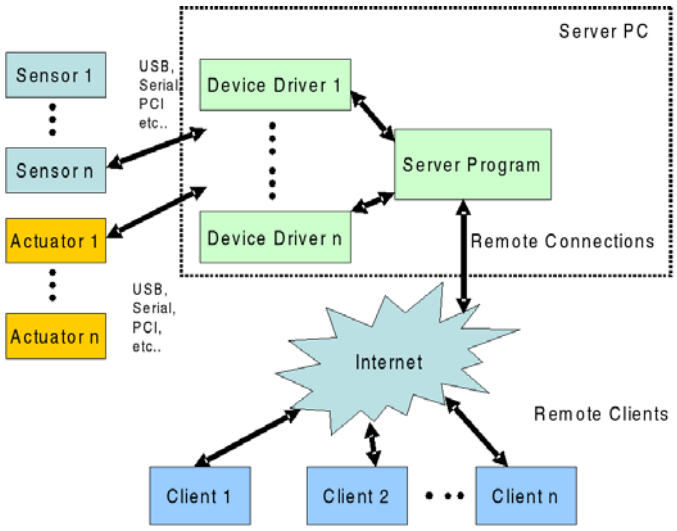

Figure 3
The common structure of Player

3 The Designed Structure of the System

Basically, the system is organized like this. One Player server is running on each distant computer. The driver used on these machines supports the image processing. A Player client (Videoplayer) displays the output image, by connecting at the same time the two servers, and grabbing the treated data. Therefore, the composition of the final output is realized on the client side. Figure 4 illustrates this principle.

We can notice, that the first two computers with the cameras form DINDs, introduced in section 3. This proves, that the Intelligent Space technology and concept can be used in a very wide range of different fields, for example for telemanipulation.

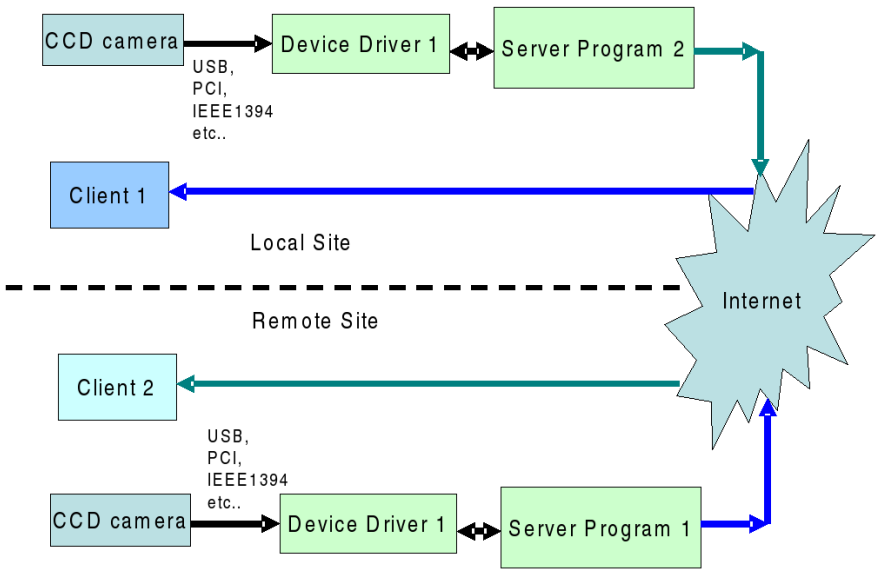

Figure 4
The designed structure of Player

3.1 Drivers and Plug-ins

In order to implement, the image processing on the server's side, it is necessary to write a brand new specific driver, in the Player's project. The Linux system which supports video under Linux is called Video4Linux. That is why the original driver used in Player to support acquisition of video is called *camerav4l.cc*. The way people has to follow to do so is described in the Player's documentation, however it has been very useful to study the structure of the original video4linux driver in order to understand the way data are treated on the server side. Indeed, to realize the image processing, we have to work on the images and the structure of the corresponding data's in this C++ driver.

The are two distinct types of drivers in Player:

- *Static drivers* have their code in the main Player distribution and they are statically linked into the server. Generally speaking, such drivers will be added by the lead developers.
- *Plug-in drivers* are shared objects that are loaded at runtime (like loadable modules in the Linux kernel). They are the recommended method for all new, experimental or third party drivers.

Logically, we decided to create a Plug-in driver. This did not take too much time. Indeed, as we had to add functions to the existing grabbing process, we just had to

reuse the original one, that is to say, changing few variables and adding preprogrammed functions, specific to a plug-in driver.

4 Image Processing

Now that we know more about the environment and the system, we can concentrate on the image processing. Two main artistic directions were designed during the project. The first one is based on background subtraction and skin color detection, the other one is based on edge detection. The theory of background subtraction and skin color detection image processing are given in [7]. Since [7] was written at the sema department, we do not want to repeat it, in this paper we concentrate on the edge detection algorithm.

4.1 Edge Detection

The third image processing imagined to detect someone standing in front of the camera is the edge detection. The two previous techniques, theoretically, did not implied to have any specific background on the back of the user. However, here we can easily image the various problem caused, if we do not use a good background.

For instance, in an office or using a plain background:

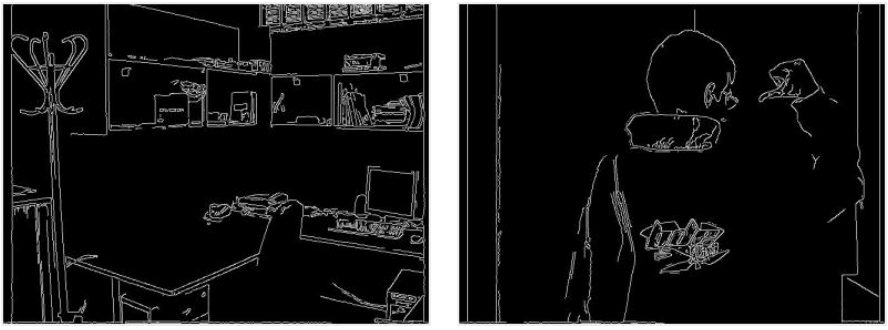

Figure 7
Edge detection examples

A lot of algorithm exist in order to process an edge detection. For instance: Sobel, Prewitt, Roberts, Frei-Chen , Laplacien or Canny. We used this last one, because it was the one implemented in the IntelOpenCV project [6]. So we did not had to implement it by myself. Canny is one of the most efficient process among all the previous ones.

With edge detection everything depends on a pixel and its neighbors. A comparison of that pixel's brightness and its neighbor's brightness will ultimately determine if the pixel is part of an edge or just an ordinary pixel. When a pixel's value is within a certain range of its neighbors it is most likely not part of an edge. When the pixel's value differs from its neighbors and is not within the range it is usually considered to be a part of an edge. The change in pixel value from pixel to its neighbors is usually done in a smooth manner.

"The Canny operator was designed to be an optimal edge detector. It takes as input a gray scale image, and produces as output an image showing the positions of tracked intensity discontinuities."

"The Canny operator works in a multi-stage process. First of all the image is smoothed by Gaussian convolution. Then a simple 2-D first derivative operator is applied to the smoothed image to highlight regions of the image with high first spatial derivatives. Edges give rise to ridges in the gradient magnitude image. The algorithm then tracks along the top of these ridges and sets to zero all pixels that are not actually on the ridge top so as to give a thin line in the output, a process known as non-maximal suppression. The tracking process exhibits hysteresis controlled by two thresholds: T1 and T2, with T1 > T2. Tracking can only begin at a point on a ridge higher than T1. Tracking then continues in both directions out from that point until the height of the ridge falls below T2. This hysteresis helps to ensure that noisy edges are not broken up into multiple edge fragments."

5 The Output Image

Once, the two people on each side of the video conference have been detected in one of the way we have previously developed, we want to create the final output image.

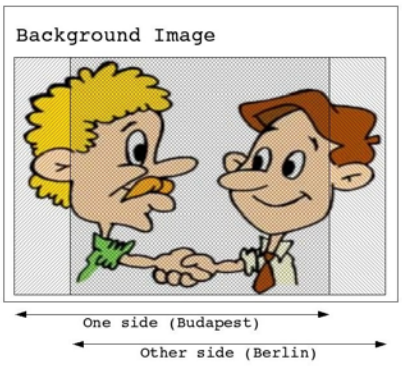

Figure 8
The output image

This image illustrates the way the two detected members of the conference can be displayed on the same image, on each side of the final picture and can interact together on the center of the output. Here, the two men shake their hand!

6 Implementation and Results

6.1 Implementation

To realize such an output, we used the Videoplayer software. This software is essentially used as a client for Player to display the output of a camera connected to Player. We can sum up the different steps which lead to the desired output:

1. Copy the chosen background in the frame.
2. Choose a specific color, considered as the background.
3. Scan the first image pixel-by-pixel, and if is not the background color we have previously chosen, then it should be copied to the frame.
4. Redo the same process with the second image.
5. Scan the part of the output where the two grabbed images cover each other and find the common pixels.
6. Alterate or transform the output frame depending on the number of common pixels or transform these pixels and their neighbors.
7. Display the final frame.

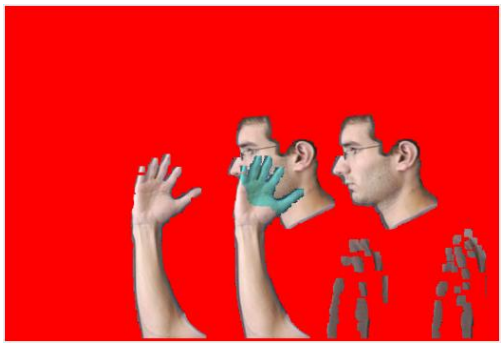

Figure 9

An interaction example

These images have 800x600 resolution. Unfortunately, on these examples, the two images grabbed are the same instead of two different, however we can appreciate

the result. Here, the background launched is a whole red image, but we can use any kind. Finally, we can see that the common pixels have their color changed. On the first image, the pixels become black instead of white, on the second image the common pixels have their color reversed, that is why we can see this green part on the image. This process is very basic, but once we have detected which parts people have in common, we can easily imagine any type of interaction.

6.2 Results

The results of the image processing are already shown in the previous sections, where the theory was discussed. Two major problems are going to be discussed about image processing in this section.

The first problem is the light condition, which can influence the output result in a bad way. It is mainly because of the HSV color space. When colors are too close to black or white, it is hard to differentiate the colors, as it comes from the structure of the HSV color space. Fortunately, we use the image processing for video conference. This means, that the environment is specially designed for this, the light conditions in his workshop can be precisely adjusted for the optimal output.

The second problem with skin color detection is, that problems arise, when the people wear clothes, which color is close to the skin color. Of course, the system will recognize the clothes as skin, so the output will be very wrong.

The solution for this problem is similar to the solution of the first problem. As we can set the light conditions at the video conference, we can paint its wall to the optimum color, similarly, we can give people special clothes to wear. In the tests we found that light blue color is the best for skin color detection. Of course, if we check the HSV color circle, we find that blue is the color at the opposite side from skin color.

By taking these steps, these two major problems are eliminated.

Some other image processing characteristics have to be mentioned also, such as the speed of the processing, or the frame rate what we can achieve. Using separate computers for the two cameras, the approximate maximal frame rate is around 10 frames per second. Since the operator is moving relatively slowly, we can say that this frame rate is enough.

Besides image processing, we have to say a few words about the whole system itself. The most important issue is the required Internet bandwidth needed to transfer the images. As we already mentioned, the achieved frame rate is 10 fps. The average sizes of the compressed images are about 30 kB. This implies, that if we want to transfer 10 images every seconds, which is the minimum for the video conference to be enjoyable, the minimal required bandwidth is 2 Mbits in both ways, which is quiet high. We can say, that this is the weakest point of the system,

since we cannot connect every point of the world with this bandwidth, and also the price of such an Internet connection is high.

Conclusions

The introduced system, which integrates different techniques to achieve the final result was deigned and implemented at the Budapest University of Technology and Economics. The whole system is realized as two DINDs, known from the Intelligent Space. The background removal, skin and edge detection algorithms were designed and implemented into the system. The first interactive effect implemented into the system is the change of color when two people touch each other, but many other effects are also imaginable.

Acknowledgement

The authors wish to thank the JSPS Fellowship program, National Science Research Fund (OTKA T046240), Control Research Group and János Bolyai Research Scholarship of Hungarian Academy of Science for their financial support and the support stemming from the Japanese-Hungarian Intergovernmental S & T Cooperation Programme.

References

[1] J.-H. Lee, H. Hashimoto. *Intelligent Space – Its Concepts and Contents*, Advanced Robotics Journal, Vol. 16, No. 4, 2002

[2] P. Korondi, H. Hashimoto. *Intelligent Space as an Integrated Intelligent System,* Electrical Drives And Power Electronics International Conference, Slovakia, 24-26 September, 2003

[3] http://playerstage.sf.net

[4] M. Piccardi. *Background Subtraction Techniques: a Review,* Faculty of Engineering UTS, 2004

[5] Xiu-Na Xu, Zhe-Ming Lu. *Real-time Face Detection Based on Skin Color Model.* Department of Automatic Test And Control, Harbin Institute of Technology, China

[6] http://www.intel.com/technology/computing/opencv/index.htm

[7] A. Gaudia, P. Szemes, B. Takarics, P. Korondi. *Recognizing Unusual Behavior for Intelligent Space.* 6th International Symposium of Hungarian Researchers on Computational Intelligence, 2005 Budapest