New Possibilities in Human-Machine Interaction

András A. Tóth, Balázs Tusor[2], Annamária R. Várkonyi-Kóczy[1,2]

[1]Institute of Mechatronics and Vehicle Engineering, Budapest Tech
[2]Integrated Intelligent Space Japanese-Hungarian Laboratory
[1]koczy@mit.bme.hu, [2]bamota@gmail.com

Abstract: Nowadays, with the never unseen spreading of computer controled applications, human-machine interaction has also become an important issue. Soft computing based models built in machines make possible for humans to keep and apply their natural way of thinking and behavior when communicating with machines. A good example can be the Intelligent Space (iSpace) which is a new operational paradigm according to which the active space itself possesses the intelligence, distributed in the whole space. The usual aim of iSpace is to improve the living conditions of humans, creating an intelligent environment for higher quality, natural, and easy to follow lifestyle. In this paper, a new intuitive interface is presented which helps to control the iSpace by simple and natural hand gestures and movements.

Keywords: Ubiquitous computing, Intelligent Space, Man-Machine Interaction, Smart Environments, Image Processing, Intuitive User Interface

1 Introduction

Today, with the spread of machine intelligence, "smart environment" became a popular tool for humans collecting information, bearing, forming the environment, getting assistance etc. *Intelligent Space* (*iSpace*) is an intelligent environmental system offering ambient intelligence for improving the comfort and safety of everyday life as well as for achieving personalised healthcare and independent living for disabled persons. The main and ultimate goal of such systems is to build an environment that is human centered, comprehends human interaction, and satisfies them [1]. This means that the system should be easy to use for the people in it: they should be able to express their will through intuitive actions and there should be no need for them to learn how the system is to be used.

Intelligent Space (iSpace) [2] is a special intelligent implementation of the *Ubiquitous Computing* paradigm [3], which can be any smart area, such as a room,

a railway station, an underpass, road crossing, or even a town, etc. equipped with intelligent sensors and agents. The main feature of the iSpace is that the intelligence itself is not present in the agents but it is distributed in the whole space. Thus, the architecture of the artificial agents, such as robots, is quite simple as they are coordinated by the intelligent sensors.

Another important feature of the Intelligent Space is its capability of observing what is happening in it and to build models of the environment. In case of necessity the iSpace is also capable to interact with the environment in order to achieve some kind of change or give information to its users.

ISpace applications, existing and under development, aim at such tasks as the monitoring of physiological functions of humans, the positioning and tracking of humans [1], the localization of mobile robots [1], the control of robots [1], and finding paths for them by using itineraries taken by people [4], etc.

In this paper, authors introduce a new human-machine interface to iSpace applications which is intuitive and easy to use. By this, users become able to issue orders to the iSpace assuming simple gestures of their hands and/or issuing movements with them.

The rest of this paper is organized as follows: Section 2 describes the most important features and the architecture of the iSpace. Section 3 details the set up and the operation of the proposed man-machine interface. Section 4 analyzes the performance of the experimental system. Finally, Section 5 concludes the paper and outlines possible improvements.

2 The Intelligent Space

2.1 Purposes and Features of the iSpace

Intelligent Space (iSpace) is an intelligent environmental system originally developed at Hashimoto Lab, University of Tokyo, Japan. The main and ultimate goal of the Intelligent Space is to build an environment that comprehends human interaction and satisfies them [1]. This means that the system should be easy to use for the people in it: they should be able to express their will through intuitive actions and there should be no need for them to learn how the system is to be used.

Another important requirement is that the system should be human centered, that is it should not be disturbing for people: e.g. there should be no need of portable devices in order to interact with the space, and the installation of intelligent devices into an existing area should not alter that area overly. Beyond supporting

humans, the iSpace should provide an interface also to its artificial agents, e.g. to the robots providing physical services to the humans using the iSpace.

The most characteristic feature of the iSpace is that the intelligence is distributed in the whole space, not in the individual agents around. The other fundamental capability of the iSpace is that it is able to monitor what is going on in the space and to build models based on this knowledge. The system is also capable to react with its environment and provide information or physical services to its users.

The iSpace is an active information space: that means that the clients are able to request information from the system which is provided to them by the active devices. The iSpace is thus a so called *soft environment*: it has the capability to adapt itself to its clients [2].

2.2 Architecture of the iSpace

There are several requirements that the hardware and software architecture of the iSpace has to satisfy. As stated in [4] these are modularity, scalability, ease of integration, low cost, and ease of configuration and maintenance.

- Modularity is important in order to ensure the possibility of run-time reconfiguration when a component is added or removed from the system.
- Scalability is also relevant, because the iSpace must be able to adapt itself to environments of various sizes and shapes; furthermore it should be possible to integrate local systems into larger ones.
- Ease of integration means that it should be easy to integrate existing intelligent components into the system.
- Low cost is to be understood on the components: it is important, because the iSpace is built of many smaller components, thus the economic factor is determined by the cost of the components.
- Finally, it is expected that it should be easy to set up and maintain an existing system. Thus, the iSpace should also have the capability to learn by itself.

Software tasks are implemented as distributed individual processes and can be categorized into three types: *sensor and actuator servers*, *intermediate processing,* and *application processes*.

The task of the sensor and actuator servers is the preprocessing of the information and the delivery of the preprocessed information to the network. Sensor fusion, temporal integration, and model building occurs at intermediate processing level. The tasks performed at this level might require real-time capabilities, thus components performing them should be located near the sensor. Finally, the application processes are those that perform the actual applications of the iSpace. At this level only low amount of data is processed and a slower reaction time is

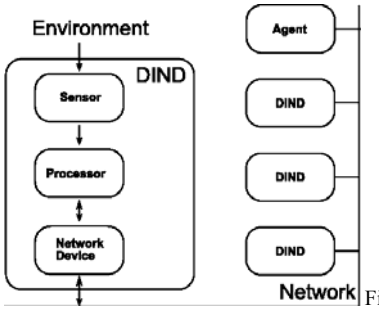

Figure 1
Structure of a DIND

sufficient. Functions at this level are required to be easily portable and maintainable by the user.

The architecture satisfying all these requirements is described in [1]. The basic elements of the iSpace are the artificial clients and the distributed intelligent sensors. The formers include robots, which provide the physical services to the human users of the iSpace, and the monitors, which furnish them information.

DIND (*Distributed Intelligent Networked Device*) is the term for the intelligent sensor, which is the basic building element of the iSpace. This device consists of three main components: a sensor which monitors the dynamic environment, a processor, whose task is to deal with sensed data and to make decision, and a communication part through which the device is able to communicate with other DIND's or with the artificial agents of the iSpace. These parts are shown in Fig. 1.

The advantage of the architecture based on the DIND's is that the iSpace can easily be constructed and modified by using them. Indeed, an existing space can be turned into an intelligent space just by installing DIND's. The modularity of the architecture makes the renewal of functions easy, and thanks to the network an update can be effectively applied to all DIND's. The other advantage of the network is that it facilitates resource sharing.

3 Human-Machine Interaction Based on Hand Gestures and Hand Movements

3.1 Overview of the System

The comfortability of the way of communication between humans and the iSpace is of vital importance from the point of view of usability of the iSpace. The more natural the interaction is, the wider the applicability can be. Because of this, we

decided to use one of the basic human talents, coordinated complex moving of the hands, for communication. The idea is that after that the sensors of the iSpace detect and separate the hands of the humans, the intelligence of the system determines the gestures and movements and translates them to desired actions.

At current stage of the development, we applied simplifications. The procedures described in this section assume that the whole field of view of the sensor is filled with a homogeneous background and the user makes hand gestures and movements before this background. No other parts of the body or other objects are visible on the image delivered by the sensor; however the absence of the hand is permitted.

Hand gesture recognition and the three dimensional modeling of the hand works with the input of two cameras. For this procedure it is presumed that the hand is fully visible on both input images or not visible at all and that the hand is steady. A further assumption is that stretched fingers are well separated and that the hand stays parallel to the cameras.

Hand tracking and movement classification uses only the input of one camera. In the case of these procedures we assume that the hands are moving slowly or they are steady. Hands visible on the picture never touch or overleap with each other. Furthermore, at most two hands are to be seen on the input sequence.

In this way the software part of the system consists of two main components: the first one processes two images of a stereo camera pair and has two functions: it classifies hand gestures and yields three dimensional locations of feature points of the hands. The second takes only the video stream of one camera as an input and performs the tracking of a slowly moving hand. Furthermore, it tries to match the shape of the movement with one of the predefined shapes. In the current implementation two predefined shapes are considered: a line and a circle but in the future other shapes will be defined, as well. The workflows of these two components are illustrated in Figs. 2 and 3, respectively.

The first step of both components is the extraction of skin regions. To achieve this, histogram back projection is performed as described in [5]. The next steps to retrieve areas containing skin include applying a threshold to the back projected image and locating the connected components, discarding those having the area below a given threshold.

3.2 The 3D Model

The three dimensional model of the hand captured by the sensors consists of the spatial location of the feature points of the hand. Three steps are executed in order to build the model: first feature points, peaks and valleys, are extracted in the images (see also [6]). Then, these feature points are matched separately, using the

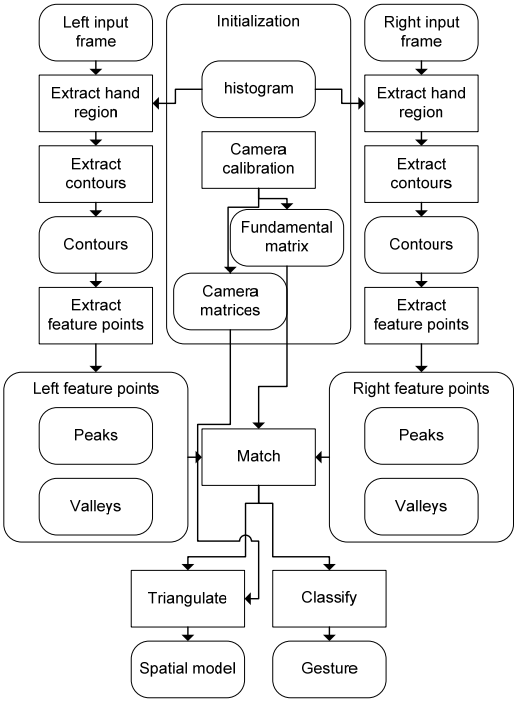

Figure 2
Gesture estimation and three dimensional reconstruction workflow

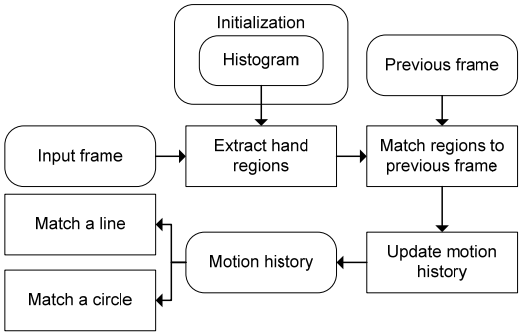

Figure 3
Block diagram of the hand tracking and movement estimation

fuzzy logic based matching algorithm described in [7]. Finally, the three dimensional coordinates are calculated using the known camera matrices (see [5]). The *Direct Linear Transform* (DLT) method is adopted in order to perform the triangulation which will yield the three dimensional coordinates of the feature points.

3.3 Hand Gesture Estimation

The matched peaks and valleys (see [5]) are used not only to build a spatial model but to estimate the hand gesture, as well.

Currently, three hand gestures have been defined which are called hand gesture A, B and C, respectively. Gesture A represents a hand with all five fingers stretched out. Gesture B is a closed fist. Finally, thumb, index and middle fingers are stretched out in case of gesture C, whilst the other two are bent. For each predefined hand gesture the number of peaks and valleys is stored and the count of respective feature points on the processed frame is compared to the stored count of the predefined gestures, as summarized in Table 1. The hand gesture estimator can yield five types of result which are: no hand on the picture, gesture A, gesture B, gesture C, other type of gesture.

For hand gesture recognition fuzzy neural networks and fuzzy inference can be used as well. The idea is based on having the coordinate model of the detected hand transformed to a fuzzy hand gesture model by circular fuzzy neural networks, which can be identified with fuzzy inference. The idea of circular fuzzy neural networks is based on the fuzzy neural networks proposed in [8], but with some modifications. Instead of using a full-wired network, connecting all input values to each hidden layer neuron, we only connect 9 certain inputs (which are 3 adjacent coordinate triplets) to each hidden layer neuron. The topology between the hidden and output layer neurons hasn't been changed. This reduction in wiring caused dramatic decrease in the time required for training, while the accuracy of the network hasn't been decreased. The last part of the hand posture identification procedure is the fuzzy reasoning based classification. In this part we compare the resulting fuzzy hand posture model to all fuzzy hand posture models stored in the rulebase. This method unites the learning ability of neural networks and thinking and ability of fuzzy systems, resulting in a robust, accurate and fast identification system with the ability of handling uncertainty.

3.4 Hand Tracking

In order to track the motions of the hand a simplified approach based on the one presented in [9] is developed. We consider only the center of gravity of the blobs, which can be computed as follows:

$$x_c(B) = \frac{\sum_{x,y \in B} x \cdot b(x,y)}{\sum_{x,y \in B} b(x,y)} \quad y_c(B) = \frac{\sum_{x,y \in B} y \cdot b(x,y)}{\sum_{x,y \in B} b(x,y)} \tag{1}$$

where (x_c, y_c) denote the center of gravity of the blob and $b(x, y)$ is the intensity at a given position.

Table 1
Expected and accepted peak and valley count for each type of hand gesture.

Gesture	**A**	**B**	**C**
Expected peaks	5	0	3
Expected valleys	4	0	2
Accepted peaks	At least 5	Exactly 0	Exactly 3
Accepted valleys	At least 3	Exactly 0	At least 1

The reason for taking only center of gravity into account is that color information is unnecessary, as all blobs represent hands and have thus the same color. Their size will also be approximately the same, as they are about the same distance from the camera. Finally, speed is neglected because it was assumed that tracked hands move slowly.

The tracking procedure works in the following way: in each frame it tries to match the current blobs to those of the previous frame by finding the minima of the match-score matrix of the distances of the blobs of the current frame and the previous one. The same ID is assigned to the matched blobs throughout the whole sequence. A new unique ID is assigned to unmatched blobs (which are supposed to belong to hands that have just entered the scene). The sequence of the centers of gravity is stored for each blob. This sequence is referred to as the *motion history* of the blob. When a hand leaves the scene, no blobs in the next frames will be matched to it again, and thus its motion history ceases to grow.

3.5 Movement Recognition

The recognition of predefined movements is achieved on a rule based manner by using some statistical parameters of the motion history yielded by the tracking procedure.

For each motion type to be estimated, appropriate thresholds for the statistical parameters have been set. Each motion history belonging to a hand still visible on the image is matched (until the first steady point) to each of the threshold-sets, and if all constraints of a given threshold-set are met, the appropriate motion is predicted. Thus, the possibility that more than one motion type is predicted for the same movement is not intrinsically excluded. The considered parameters for linear motion are the sum and variance of the angles between two subsequent motion vectors, to which upper thresholds have been set and one of the sums of the shift in directions x or y has to be above a lower threshold.

In case of the circular motion the parameters taken into consideration are the following: the maximum and minimum values of the angle between the motion vector and axis x must be above and below an upper and lower bound, respectively. The average of this angle also has to be below a given upper

threshold. The averages and sums of shift in directions *x* and *y* have all to be below an upper bound.

4 The Experimental System

4.1 Hardware

We have built an experimental set up for testing and analyzing the performance of the new interface. The hardware part of the testing system consists of two web cameras (Genius Look 316) connected to a PC (CPU: AMD Athlon 64 4000+, 1 GB RAM, OS: Windows XP). The cameras are located at one end of a table. At the other end a homogeneous background is mounted. The image processing algorithms are running on a PC and are implemented using OpenCV [10]. For the camera calibration the Camera Calibration Toolbox for Matlab has been adopted [11]. The experimental setup is illustrated in Fig. 4.

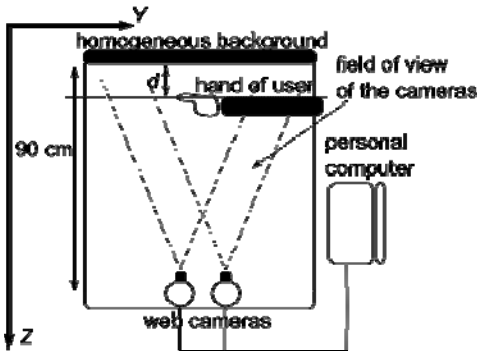

Figure 4
The experimental setup (d is the distance of the tip of the index finger from the homogeneous background)

4.2 Functional Test

We have worked with two types of tests. The purpose of the functional tests is to get the ratio of correct answers of the implemented modules when the given preconditions are met. The first measurement tests whether the hand gesture classifier based on the peaks and valleys count is able to differentiate between the predefined hand gestures A, B, and C and some negative gestures which have different count of peaks and valleys using 50 samples for each gesture type. It was found that this procedure gives acceptable results if the hand lies apart from the

homogeneous background and does not lie too near the camera. (In the first case, the contour might become noisy due to the hand region touching the shadow while in the latter one, peaks and valleys may fail to be detected.) The results of the measurement series satisfying these constraints are summarized in Table 2.

The second test aims at determining the accuracy of the three dimensional reconstruction. As also stated in [12], due to the lack of ground truth, it is more convenient to measure the jitter of the reconstruction rather than the absolute

Table 2
Results of the hand gesture prediction.

Input/Response	A	B	C	Negative
A	46	0	0	4
B	0	50	0	0
C	0	0	45	5
Negative	0	0	0	50

Table 3
Jitter measurement results for d=32 cm. All the values in the table are shown in mm

Direction	Maximum difference	Deviation
X	2.75	0.80
Y	3.46	0.80
Z	13.30	3.71

position itself. We measured the jitter at three different locations (d=22 cm, d=32 cm, d=42 cm, where d is the distance of the tip of the index finger as shown in Fig. 4), using 20 samples at each location. It was found that the spatial reconstruction gives little jitter in the directions parallel to the camera planes (i.e. directions X and Y), and it yields higher jitter in the direction perpendicular to the plane though the values are still usable. Results for d=32 cm are summarized in Table 3. Here Deviation means the standard deviation of the vector of the given coordinate component

$$devX = \sqrt{\frac{\sum_{i=1}^{N}\left(X_i - avg(X)\right)^2}{N-1}} \tag{2}$$

where $\boldsymbol{X}$ denotes the vector of the X coordinates, avg is the average of the X coordinates, and N is the number of the valid samples. At d=42 cm there were three samples where the peak of the index finger failed to get detected. Deviation for Y and Z components is calculated similarly.

The third test demonstrates the performance of tracking. The tracker assigns an ID to each yet unidentified blob entering the scene, which should be kept until the

object corresponding to that blob leaves the scene. The tracker meets this requirement in case of the predefined working conditions.

The last test measures the performance of the hand movement classifier. Both movements were performed 16 times from various starting points and in various directions. The linear movement classifier recognized linear motion 13 times whilst circle classifier gave the right result 12 times. Thus, the classifiers have an acceptable performance.

4.3 Timing Tests

The timing tests measure the time necessary to compute the steps described in the previous sections. It is important that these procedures work at real time speed in order to be used for the intuitive interface.

For the first subsystem (see Fig. 2) the considered steps were contour extraction, peaks and valleys localization, feature point matching, and computation of the spatial points. Each of the first three steps was found to operate in the order magnitude of 10 ms, which is about the frame grabbing rate of commercial 15-30 Hz web cameras. The time needed by spatial reconstruction was significantly shorter: it was found to be in the order of magnitude of 10 μs.

The steps considered for the second subsystem (see Fig. 3) are blob extraction, blob tracking, and testing for circular and linear movements. Magnitude of blob extraction time is the same as that of frame grabbing, blob tracking and shapes detecting time was 100 μs and 10 μs, respectively.

In order to measure the time the standard C routine *clock()* was used. As the granularity of this timer is larger than the measured intervals, the timings were obtained by averaging more measurements.

Conclusions

In this paper, the concept and the experimental setup of a new man-machine interface are introduced. This interface makes able humans to control iSpace by simple hand gestures and hand movements.

Acknowledgments

This work was sponsored by the Hungarian National Scientific Fund (OTKA 78576).

References

[1] J-H. Lee, K. Morioka, N. Ando, H. Hashimoto: “Cooperation of Distributed Intelligent Sensors in Intelligent Environment,” IEEE/ASME Transactions On Mechatronics, Vol. 9, No. 3, 2004

[2] J-H. Lee, H. Hashimoto: “Intelligent Space,” In Proc. of the International Conference on Intelligent Robots and Systems, IROS 2000, Vol. 2, pp. 1358–1363

[3] M. Weiser, “The Computer for the Twenty-First Century,” Scientific American, 1991, pp. 94–104

[4] G. Appenzeller, J.-H. Lee, H. Hashimoto: “Building Topological Maps by Looking at People: An Example of Cooperation between Intelligent Spaces and Robots,” Intelligent Robots and Systems, Vol. 3, No. 7, 1997, pp. 1326–1333

[5] A.R. Várkonyi-Kóczy, A.R., A.A. Tóth, “ISpace – a Tool for Improving the Quality of Life,” Journal of Automation, Mobile Robotics & Intelligent Systems, Vol. 3, No. 4, 2009, pp. 41-45

[6] S. Malik, “Real-time Hand Tracking and Finger Tracking for Interaction,” CSC2503F. Project Report, 2003

[7] A.R. Várkonyi-Kóczy: “Autonomous 3D Model Reconstruction and Its Intelligent Application in Vehicle System Dynamics,” In Proc. of the 5th International Symposium on Intelligent Systems and Informatics, SISY 2007, Subotica, Serbia, Aug. 24-27. 2007, pp. 13-18

[8] Ishibushi, H., Tanaka, H., “Fuzzy neural networks with fuzzy weights and fuzzy biases,” In Proc. of the IEEE Neural Network Conf, San Francisco, 1993., Vol. 3., pp. 1650-1655

[9] S. S. Intille, J. W. Davis, and A. F. Bobick, “Real-Time Closed-World Tracking,” In Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June 1997, pp. 697-703

[10] G. Bradski, T. Darrell, I. Essa, J. Malik, P. Perona, S. Sclaroff, C. Tomasi et al, Intel OpenCV Library, software available online at: http://sourceforge.net/projects/opencvlibrary

[11] J-Y. Bouguet, Camera Calibration Toolbox for Matlab, software available online at: http://www.vision.caltech.edu/bouguetj/calib_doc/

[12] J. Segen, S. Kumar, Shadow Gestures: 3D Hand Pose Estimation Using a Single Camera, In Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1999