ISPACE – A TOOL FOR IMPROVING THE QUALITY OF LIFE

Annamária R. Várkonyi-Kóczy, András A. Tóth

Abstract:

Intelligent Space or iSpace is a new kind of computing system aiming at improving the environments of humans, creating a natural and easy to use solution. Its main feature is that the intelligence is not implemented separately in the actors, but it is distributed in the whole space. In this paper, we summarize the concepts of iSpace and some of those methods which can advantageously be used for developing a new man-machine interface in iSpace applications.

Keywords: *Ubiquitous computing, intelligent Space, Image Processing, Intuitive User Interface*

1. Introduction

A new paradigm in user-machine interaction is the so called "*Ubiquitous Computing*" approach introduced in [1]. A Ubiquitous Computing system consists of more devices communicating with each other, thus it is a distributed system. As their name suggests, Ubiquitous Computing systems are present everywhere in an environment such as an office or a home and their task is to enhance that environment. The main goal of Ubiquitous Computing systems, however, is to offer such an interface to the user that should be so natural and easy to interact with, that users become unaware of the fact that they are using a computing system.

An implementation of the Ubiquitous Computing paradigm is the *Intelligent Space (iSpace)* [2], which is an area, such as a room, 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.

In this paper, authors discuss the main features of iSpace and also summarize methods which will be used in developing a new human-machine interface 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 paper is organized as follows: Section 0 describes the most important features and the architecture of the iSpace. In Section III we give an overview of the methods that are used in the interface. Finally, Section IV concludes the article.

2. The Intelligent Space

A. 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 [3]. 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: for instance 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. Thus, there is little logic in the robots which are controlled by the iSpace. 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 physical services are provided by the robots.

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].

There are several applications currently developed or planned for the iSpace. These include the positioning and tracking of humans [3], the localization of mobile robots [3], the control of robots [3], and finding paths for them by using itineraries taken by people [4].

B. 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 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 [3]. 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.

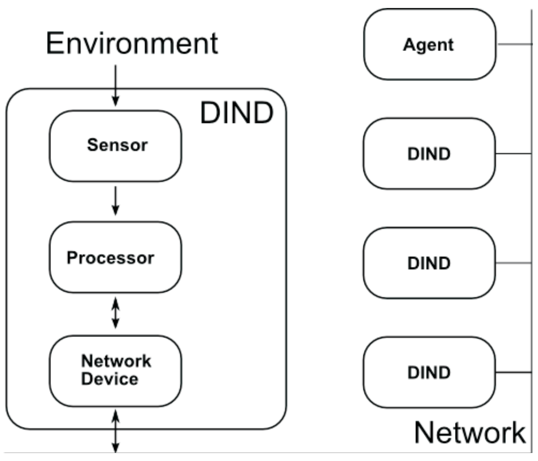

Fig. 1. Structure of a DIND.

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. Overview of the methods helping in developing iSpace by a new intelligent interface

A. Camera Calibration

Cameras taking input pictures can be described by means of so called extrinsic and intrinsic parameters.

The former mean the position and orientation of the camera in space. The latter are those parameters that are independent of locations, i.e. focal length, distortion, and skew. The transformation for the so called pin-hole camera model can be described in terms of these parameters:

$$x = P \cdot X = K \cdot R \cdot [I \,|\, -C] \cdot X \tag{1}$$

where x denotes the image location of the point, X stands for the spatial location, and P is the *camera matrix*. The matrices in the product yielding the camera matrix are the *camera calibration matrix* (K), the *rotation matrix* between the camera coordinate frame and the world coordinate frame (R), the identity matrix (I), and the center of the *camera coordinate frame* (C). The camera calibration matrix consists of the following values:

$$K = \begin{bmatrix} a_x & s & p_x \\ 0 & a_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

where (p_x, p_y) represent the image center, a_x and a_y stand for the horizontal and vertical size of pixels, and s denotes the skew. For a more elaborate description of camera parameters see e.g. [5].

The process of estimating these parameters is commonly referred as *camera calibration* in literature. There are more known approaches for calculating these parameters. One possible way is based on the correspondence between spatial- and picture points. Another possible solution commonly adopted relies only on picture points. In this latter case, however, the points have to satisfy some other constraint, for instance they have to lie on the same plane in a predefined pattern.

B. Histogram Based Skin Detection

Image data can be stored using various parameters. The well-known RGB color space uses red, green, and blue color components. The components of the HSV (Hue, Saturation, Value) color space in contrast uses quantities that are more related to the every day color description. Hue is the parameter we would define as "*the color*", saturation means the concentration, and value is the measure of brightness. As stated in [6] this color space is less sensitive to lighting changes than RGB, and the hue value is invariant for the various skin "colors". In fact, it is only the saturation that increases for dark skinned people.

The *histogram back projection* procedure [6] locates areas with a given color model. Let's consider HSV:

$$backproj(x,y) = H(hue(x,y)) \tag{3}$$

where $backproj$ means the single channel backprojection image, H is the hue-histogram, and hue is the single channel hue plane of the input image. This means that given the color model (i. e. the histogram) of a given object (in this case human skin), for each pixel on the image the value of the corresponding histogram bin is written back to the image. Thus the probability distribution of skin is obtained.

C. Feature point extraction

A common procedure in literature adopted to extract feature points is corner detection. Some known methods are those of Harris [7], Förstner [8], He and Yung [9], the SUSAN Corner Detector [10], and the Fuzzy Corner Detector [11].

Another approach, which considers curvature extrema, is the extraction of *peaks* and *valleys* as described in [12].
This latter algorithm works in the following way: First, the contour points of the blobs are extracted in counter clockwise order. After this, the *k-curvature* (see e.g. [12]) for each contour point is computed as follows:

$$k(i) = \arccos\left[\frac{C[i-k]-C[i]}{\|C[i-k]-C[i]\|} \times \frac{C[i+k]-C[i]}{\|C[i+k]-C[i]\|}\right] \tag{4}$$

where $C[i]$ denotes the picture coordinates of the i^{th} contour point. Thus the k-curvature is defined as the angle between the vectors $(C[i], C[i-k])$ and $(C[i], C[i+k])$. Contour points whose k-curvature is less than a given threshold are classified as peak/valley candidates. For these points the rotation direction of the above mentioned vectors is also computed based on the sign of the cross product of the two vectors, thus the convexity of the corners can be determined.

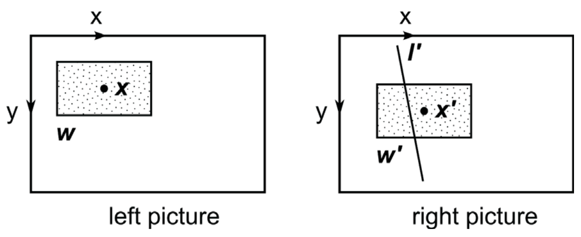

Fig. 2. Environments of matched points. l' denotes the epipolar line corresponding to x.

$$convexity(p) = \text{sgn}[(C[i-k]-C[i])\times(C[i+k]-C[i])]\,. \tag{5}$$

$convexity(p)$ determines whether the p^{th} extremum is convex or concave. Feature points at convex extrema of the contour are referred to as *peaks* whereas those at concave extrema are referred to as *valleys*. Since this procedure usually produces more candidate points in the intervals containing real peaks and valleys, for each interval one real peak/valley has to be chosen, discarding the others.

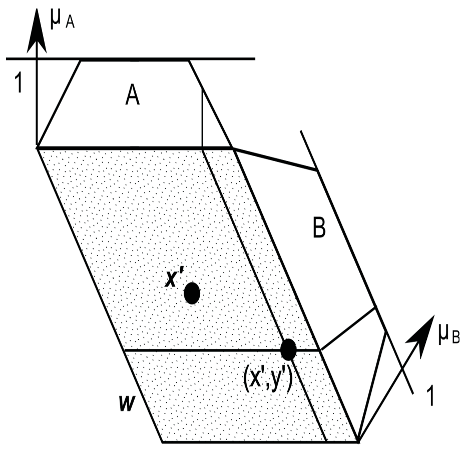

Fig. 3. Weighting fuzzy functions of the environments.

D. Feature point matching

Feature points in a stereo image pair can reliably be matched using the epipolar constraint

$$x'^{T} \cdot F \cdot x = 0 \tag{6}$$

where x' and x stand for the corresponding points in the right and left image, respectively and F denotes the fundamental matrix, which is a matrix mapping points in one of the stereo images unambiguously into a line in the other stereo image, and a similarity measure of the neighborhoods of the feature points (see [5] and [13]).

The algorithm described in [13] is a fuzzy logic based matching algorithm, which works in the following way: first the candidate pairs (i.e. points lying within a given fuzzy neighborhood of their corresponding epipolar lines) are located. Then for each candidate point pair detected in the stereo image pair, the sum of differences of their neighborhoods weighted by a two dimensional fuzzy set is calculated:

$$\sum_{\substack{x,y\in w\\ x',y'\in w'}} \left|I(x,y) - I'(x',y')\right| \cdot \mu_a(x',y') \cdot \mu_b(x',y') \tag{7}$$

where $I(x,y)$ and $I'(x',y')$ mean the intensity in the left and right images, respectively and μ_a and μ_b denote the membership functions used as weighting functions. The environments of the feature points are denoted as w and w'. These values are shown in Figs. 2 and 3.

The pairing which yielded the smallest sum of weighted difference is considered to be matching.

E. Three Dimensional Reconstruction

In order to locate the matched feature points in space, the relative position of the cameras must be known beforehand. This can be achieved by calibrating the cameras and extracting the extrinsic parameters. Then the position of points can be computed using one of the methods described in e.g. [5].

The *Direct Linear Transform (DLT)* is based on the fact that the projection equation (1) can be transformed in the following way:

$$A \cdot X = 0, \tag{8}$$

where X stands for the location of the spatial point and A is defined as:

$$A = \begin{bmatrix} x \cdot p^{3T} - p^{1T} \\ y \cdot p^{3T} - p^{2T} \\ x' \cdot p'^{3T} - p'^{1T} \\ y' \cdot p'^{3T} - p'^{2T}. \end{bmatrix} \tag{9}$$

Here (x,y) and (x',y') denote the picture coordinates of the matching feature points, p^{iT} and p'^{iT} stand for the i^{th} and jth rows of the left and right camera matrices respectively. Taking the *Singular Value Decomposition (SVD)* of A yields the coordinates of the spatial point corresponding to the image points:

$$A = U \cdot D \cdot V^T \tag{10}$$

where U, D and V are the matrices yielded by the SVD. The last column of V represents the coordinates of the spatial point.

F. Blob Tracking

There are various known approaches in the literature for implementing the tracking of moving objects on a video sequence.

One of them is the *Closed World Tracking*, proposed in [14]. As stated there, *"a closed-world is a region of space and time in which the specific context of what is in the region is assumed to be known"*. Here *context* means a method for representing knowledge in a dynamic multi-object tracking problem. The solution matches blobs in subsequent frames by comparing their size, position, speed and color. For each of these features it calculates the so called *match-score matrix* which is a matrix containing the distances in feature space of the objects and the blobs in a frame. Then, in order to obtain the best matches, it finds the minima of the weighted sum of the four match-score matrices.

Another approach for tracking a moving object is the *CAMSHIFT (Continuously Adaptive Mean Shift)* algorithm [6] based on the mean shift algorithm which is a gradient based searching procedure that operates on a back projection. It iteratively moves a fixed size search window from a given start position, until movement converges to zero. In each step, the new location of the center of the search window is the mode of the area under the window. The CAMSHIFT algorithm enhances the mean shift algorithm by dealing with the temporal change of the probability distribution. This procedure also iteratively moves the search window like the CAMSHIFT but it also changes its size according to the zeroth moment of the area under the search window. The advantages of the CAMSHIFT procedure are its low computational cost and its robustness with relation to outliers.

A third tracking method is the hybrid tracking algorithm proposed by the authors of the iSpace (see [15]). This procedure also relies on the probability distribution. One iteration of the hybrid tracking algorithm consists of three steps: in the first one the new position of the blob is estimated based on its previous position and its speed. The next step is the measurement of the position achieved by using the mean shift algorithm. The area located by the mean shift procedure is compared to the object model by their Bhattacharyya distance, which is also described in [15]. If the Bhattacharyya distance exceeds a given threshold, the mean shift window is temporarily enlarged and the searching procedure is repeated from the position of the previous frame. This part of the algorithm deals with the case when the tracker loses the tracked object for instance due to a sudden change in the speed or direction of the movement.

4. Coclusions

In this paper the concept of iSpace is summarized. Different image processing and computer vision methods are also presented which can serve as a basis for developing a hand gesture and -movement controlled man-machine interface in iSpace. The new interface is natural and comfortable to use, thus can contribute to the improvement of the quality of life of humans in smart environments.

ACKNOWLEDGMENTS

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

AUTHORS

Annamária R. Várkonyi-Kóczy* - Institute of Mechatronics and Vehicle Engineering, Budapest Tech, Integrated Intelligent Space Japanese-Hungarian Laboratory.
E-mail: koczy@mit.bme.hu.
András A. Tóth - Integrated Intelligent Space Japanese-Hungarian Laboratory. E-mail: bamota@gmail.com.
* Corresponding author

References

[1] M. Weiser, *The Computer for the Twenty-Frst Century*, Scientific American, 1991 pp. 94104,.

[2] Lee J-H., Hashimoto H., "Intelligent Space". In: *International Conference on Intelligent Robots and Systems 2000* (IROS 2000), vol. 2, 2000, pp. 1358-1363.

[3] Lee J-H., Morioka K., Ando N., Hashimoto H., "Cooperation of Distributed Intelligent Sensors in Intelligent Environment", *IEEE/ASME Transactions on Mechatronics*, vol. 9, no. 3, 2004.

[4] Appenzeller G., Lee J.-H., Hashimoto H, "Building Topological Maps by Looking at People: An Example of Cooperation between Intelligent Spaces and Robots", *Intelligent Robots and Systems*, vol. 3, issue 7, 1997, pp. 1326-1333.

[5] Várkonyi-Kóczy A.R., A Tóth, *ISpace - a Tool for Improving the Quality of Life*, Journal of Automation, Mobile Robotics and Intelligent Systems. Special issue for Inter-Academia 2009, vol. 3, no. 4, 2009,

[6] Várkonyi-Kóczy A.R., *"Autonomous 3D Model Reconstruction and Its Intelligent Application in Vehicle System Dynamics"*. In: 5th International Symposium on Inte-

lli-gent Systems and Informatics (SISY 2007), Subotica, Serbia, 24th-27th August 2007, pp. 13-18.

[7] Intille S.S., Davis J.W., Bobick A.F., "Real-Time Closed-World Tracking". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 1997, pp. 697-703.

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

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

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

[11] Várkonyi-Kóczy A.R., "Fuzzy Logic Supported Corner Detection", *Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology*, vol. 19, issue 1, 2008 pp. 41-50.

[12] Malik S., *Real-time Hand Tracking and Finger Tracking for Interaction*, CSC2503F. Project Report, 2003.

[13] Várkonyi-Kóczy A.R., "*Autonomous 3D Model Reconstruction and Its Intelligent Application in Vehicle System Dynamics*". In: *5th International Symposium on Intelligent Systems and Informatics (SISY 2007)*, Subotica, Serbia, 24th-27th August 2007, pp. 13-18.

[14] Intille S.S., Davis J.W., Bobick A.F., "Real-Time Closed-World Tracking". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 1997, pp. 697-703.

[15] Kazuyuki M., Lee J-H., Kuroda Y., Hashimoto H., "*Hybrid tracking Based on Color Histogram for Intelligent Space*, Artificial Life and Robotics", vol. 11, no. 2, 2007, pp. 204-210.