Real-Time 3D Hand Gestures Recognition for Manipulation of Industrial Robots

T. Cerlinca[1], S. G. Pentiuc[1], V. Vlad[1]
[1]*Stefan cel Mare University of Suceava,*
13 University St., RO-720229 Suceava, Romania, phone: +40230524801
tudor_c@eed.usv.ro

***Abstract*—This paper presents an innovative system for industrial robot manipulation through 3D hand gestures. The interaction between the human operator and the robotic system is done via a dynamically created 3D virtual environment which is a precise copy of the robot's real working environment. The virtual environment enhances the manipulation of different objects and the selection of the desired action, in a natural way, only though hand gestures. Unlike most of the HRI based systems, ours is not meant for moving the robot or its mobile arm from one place to another, but to perform a specific task comprising of a list of specific operations. This kind of interaction has an important advantage: it eliminates the dead-times which are specific to the direct-interaction based systems. The gesture recognition algorithm processes depth maps grabbed from a stereoscopic camera system and uses Dynamic Time Wrapping to compute the similarity between the hand trajectories acquired in real time and those from the gestures dictionary.**

***Index Terms*—Real time systems, algorithms, human robot interaction, gestures recognition.**

I. INTRODUCTION

Nowadays, the Human Robot Interaction (HRI) is a very attractive research area with multiple and quite interesting applications to the industry field. The interaction between a human operator and a robotic system can be achieved in several ways, as follows: using the robot's internal control panel, using a remote controller, through voice commands, through a dedicated software application which allows remote control and even through hand gestures. As the voice commands and the hand gestures allow for a natural communication, there is an increasing research interest in designing and developing new methods and algorithms that will make the interaction as reliable as it can be. In the human-to-human communication, voice and hand gestures are practically inseparable: the hand gestures always strengthen the speaker's ideas; moreover, people sometimes communicate with each other only through hand gestures (e.g.: American Sign Language). In the HRI area, voice and gestures are usually seen as independent of each other, but some systems use a combination of them, aiming for an increased reliability. It has to be mentioned that most of the current HRI based systems are only meant for moving the robot or its mobile arm from one place to another. Only a relatively small number of HRI based systems approach the problem of controlling a robot through hand gestures so that it fulfils real tasks. Hand gestures recognition problem is not simple either in theory or in practice. Basically, this problem is reduced to the recognition of the hand trajectories which are acquired in real time. Depending on the requirements of the practical application, the hand gestures recognition can be done in 2D or even in 3D.

Currently, there are many methods and algorithms for 3D hand gestures recognition (a good survey is provided in [1, 2]), but some of them seem to have only a purely theoretical nature and therefore cannot be integrated into specific HRI applications. For example, some algorithms assume that all trajectories have the same length. It is obviously, that such a constraint limits the use of these algorithms. For instance, when a human operator makes the same gesture 1000 times, the associated trajectories will always have a different length. This drawback can be overcome by using a Dynamic Time Warping (DTW) based distance which also assures time and speed invariance. DTW was successfully used in [3] to control the movement of an omni-wheel robot through gestures. Yet, the system has few disadvantages as follows: it is not adaptive (for different environments, the threshold needs to be manually set) and the object being detected and tracked is not the human hand but a small red ball.

In [4] the authors propose a HRI based system for industrial robot manipulation through hand gestures. They used Levenshtein Distance on Trajectories (LDT distance) for trajectories matching and Hidden Markov Models (HMMs) for recognizing different action sequences. The hand detection is carried out using a spatio-temporal 3D model of the hand-forearm limb. Hand tracking is achieved with an improved version of the classical Shape Flow algorithm. The authors claim a 90% global recognition rate. An improved version of the classical Continuous Dynamic Time Warping (CDTW) algorithm is proposed in [5].

HMMs and Bayes classifiers were used in [6] to create a 3D gesture recognition system that is capable of recognizing 16 different classes of gestures performed with one or both hands. To detect and track the objects of interest (hands and head), the system analyses the colour component of the 2D

Manuscript received March 12, 2012; accepted April 30, 2012.

This paper was supported by the project "Progress and development through post-doctoral research and innovation in engineering and applied sciences- PRiDE - Contract no. POSDRU/89/1.5/S/57083", project co-funded from European Social Fund through Sectorial Operational Program Human Resources 2007-2013.

images and uses the Expectation-maximization (EM) algorithm.

A fully functional HRI based system which relies on a real-time 3D hand gestures recognition algorithm is presented in [7]. The authors also propose a human detection method which incorporates skin, face and leg detection. In [8] the authors propose a real-time 3D pointing gesture recognition algorithm which allows for a natural human-robot interaction. Hands are detected on the base of the colour distribution of the face region and the face was detected with an improved version of the well-known Viola and Jones detector. Tracking was achieved with a 3D particle filter and the estimation of the pointing direction relies on a cascade of HMMs.

An innovative haarlet-based 3D hand gestures recognition system which combines both RGB and ToF cameras is presented in [9]. In [10] the authors combine both gesture trajectories and acceleration signals aiming for a novel approach to 3D hand gestures recognition problem. Another novel approach which claims to be view-invariant is presented in [11]. All gestures are modelled through HMMs and their corresponding trajectories are assumed to be in a plane which is estimated through Least Squares Method. According to the authors, the recognition rate ranges from 93.4% and 93.7%.

This paper addresses the problem of controlling an industrial robot in a very natural way, through 3D hand gestures, so that it will be able to fulfil real tasks comprising of a list of specific operations.

II. SYSTEM ARCHITECTURE

The manipulation of the industrial robot through 3D hand gestures is a 7 stages process, as follows: image acquisition, object detection and 3D virtual environment construction, 3D hands detection and tracking, trajectory smoothing, hand gesture recognition, objects' manipulation within the 3D virtual environment and objects' manipulation by the industrial robot. The system architecture is presented in Fig. 1.

In the first stage, the system grabs a series of depth maps from the stereoscopic camera system and tries to detect the objects that will be manipulated by the industrial robot. The layout for this application comprises of a series of different objects which can also be of different shapes (cuboids, cylinders etc.) and sizes.

In the next stage, the system automatically creates a 3D virtual environment in which the user can interact with the detected objects through hand(s) gestures. The virtual environment is intended to be a precise copy of the real environment in which the industrial robot operates. For certain applications (e.g.: a food repository, where the objects are represented by big containers with food and/or drinks) it is hard if not almost impossible to detect all objects at once, therefore the layout needs to be manually constructed.

Once the virtual environment has been built, the human operator can interact with it through the hand gestures to accomplish the following main tasks: manipulation of objects (selection, grabbing, moving and leaving), rotation of the 3D scene and zoom in/out. The interaction ends when the human operator performs a specific gesture with both hands. At this point, the system iterates through the list of operations performed by the human operator in the virtual environment and creates a series of specific commands that will be transferred to the industrial robot. When the robot has finished receiving the list of operations that need to be done it will immediately start operating on the real layout, moving the real objects from one place to another. During this time, the system will freeze the interaction with the 3D virtual environment. Interaction will start again once the robot completed the task and the system receives a confirmation in this regard.

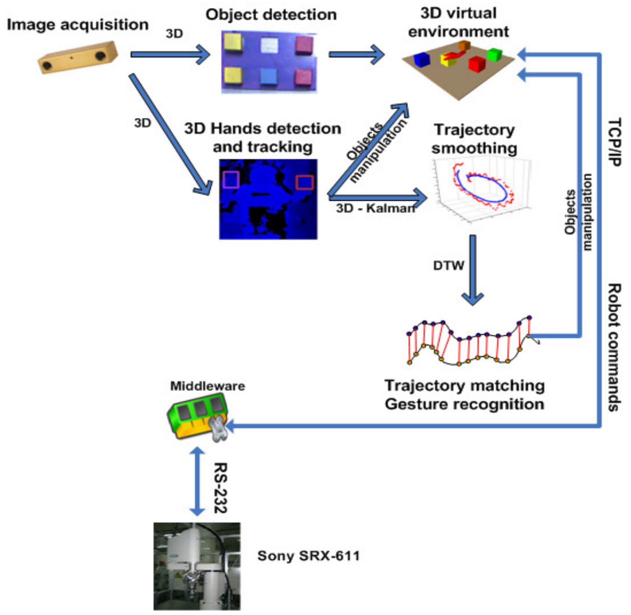

Fig. 1. System architecture.

The communication between the main system and the industrial robot is done via a specialized middleware. This way, the system does not need to know anything about the robot and the way it operates. It can be any robot which is capable of grabbing, moving and leaving objects at specific coordinates.

III. IMAGE ACQUISITION

All the algorithms that we have implemented within the industrial robot manipulation system (gestures recognition, 3D object detection etc.) process depth maps (3D information) grabbed from a stereovision camera system. In the experiments we have conducted, two different video input devices were used. The first one is Bumblebee2 IEEE-1394 FireWire stereovision camera system (provided by Point Grey Research) and the second one is the Microsoft Kinect sensor.

IV. 3D OBJECT DETECTION

As we already mentioned, the system we have developed automatically detects all the objects that will be manipulated by both the human operator (in the 3D virtual environment) and the industrial robot. Unlike the classical algorithms, ours not only that it detects 3D objects but it also processes only the depth information grabbed from a stereoscopic camera system. The depth map contains information related to the distance between different objects in the scene and the stereoscopic camera system. The algorithm consists of three stages, as follows:

1) Depth map grabbing;
2) 3D region growing. This is an original 3D extension to the classical region growing algorithm which operates on the color component of the 2D images. A detailed description of this algorithm can be found in [12]. In this stage, the algorithm detects all the objects in the repository, no matter what shape they may have. For each object, the algorithm detects the top-side shape and the distance to the ground. It has to be mentioned that the scope of this algorithm is not to provide a full 3D object reconstruction, but a basic 3D representation of each detected object so that the industrial robot will know how to work with it;
3) Shape detection. In this stage, the algorithm detects all the cuboids and cylindrical objects. The detection works fine no matter if the objects are rotated or not. Shape detection was achieved with the algorithm provided by the Open Computer Vision library (OpenCV).

This approach has few important advantages, as follows:

1) An object which shows an uneven distribution of the color component will be correctly detected and labelled [12] and not split into several smaller objects;
2) If deals with changes of lighting;
3) It gives a basic 3D representation of each detected object as follows

$$\left\{S_i, \left(x_{i1}, y_{i1}, z_i\right), \left(x_{i2}, y_{i2}, z_i\right), DG_i\right\}, \qquad (1)$$

where S_i is the shape of the object (cuboid or cylinder); $\left(x_{i1}, y_{i1}, z_i\right)$ and $\left(x_{i2}, y_{i2}, z_i\right)$ are the 3D coordinates for the upper-left and lower-right corners of the top-side shape; DG_i is distance to the ground.

The results of this stage are presented in Fig. 2. The first image shows an example of the robot's working environment: the board on which several cuboids are placed at different coordinates. The second image shows the filtered depth map and the third one, the top-side shape detection. The fourth image shows the 3D virtual environment which was built on the base of the detected objects.

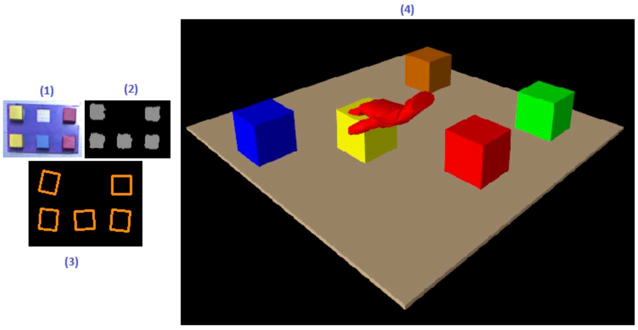

Fig. 2. Object detection. 3D Virtual environment.

V. 3D Hand Detection and Tracking. Trajectory Smoothing

This is probably one of the most important stages because all subsequent stages depend on the results provided by this one. For instance, if hands are not fast and accurately detected and tracked, their corresponding trajectories will be affected by errors and therefore, the gesture recognition algorithm may fail to correctly recognize the gestures performed by the human operator. Thus, a reliable 3D hand tracker is needed. As we already mentioned, we worked with two different stereovision camera systems and therefore with two different trackers (one for the Bumblebee2 and another one for Microsoft Kinect sensor).

Both trackers provide the 3D position of the hands in real time. They are also capable of removing the false detections by constantly analyzing the hands trajectories. The detection is considered to be erroneous when the following condition is fulfilled

$$3DDist\left(XYZ_{t_{curr}} - XYZ_{t_{prev}}\right) < ThMaxDist, \qquad (2)$$

where $XYZ_{t_{curr}}$ is the current 3D hand position; $XYZ_{t_{prev}}$ is the 3D hand position for the last successful detection; $ThMaxDist$ is the maximum allowed distance between the 3D coordinates of two consecutive detections. This threshold is not static but dynamically computed on the base of the timestamp $t_{curr} - t_{prev}$.

The first algorithm, which promotes one of our original ideas is mainly based on a classical 2D image segmentation algorithm namely, region growing [12]. The algorithm is capable of detecting the hands from long distances and it also deals with changes of lighting [12]. Although it provides good results, the algorithm has a small disadvantage: it needs an additional stage of head detection and tracking which may slow down the entire process. A detailed description of this algorithm can be found in [12].

The second algorithm which works with Microsoft Kinect sensor provides the best results when the distance between the human operator and the sensor ranges from 1.2 to 3.5 meters. If the human operator is located too close to the sensor the results tend to be noisy. The detection results are shown in Fig. 3 and Fig. 4.

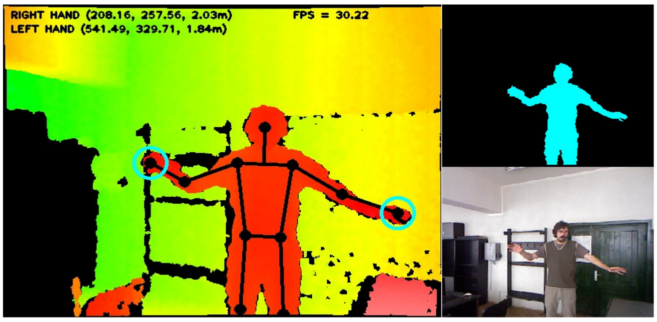
RIGHT HAND (208.16, 257.56, 2.03m)
LEFT HAND (541.49, 329.71, 1.84m)
FPS = 30.22

Fig. 3. Hands detection. Example no. 1.

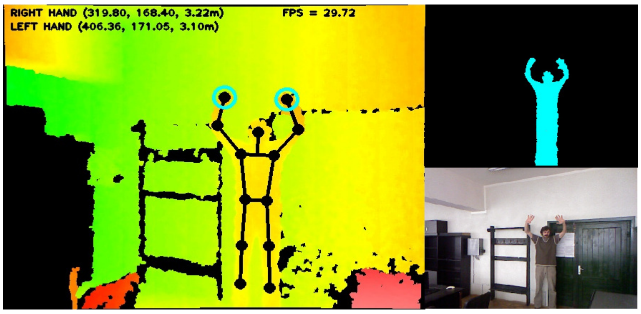
RIGHT HAND (319.80, 168.40, 3.22m)
LEFT HAND (406.36, 171.05, 3.10m)
FPS = 29.72

Fig. 4. Hands detection. Example no. 2.

We also analyzed the 3D trajectories acquired in real time by both algorithms and for different gestures and it seems the second algorithm provides slightly more accurate results. However, the hand trajectories are not yet suitable as inputs for the gesture recognition algorithm. In order to smooth the trajectories, in the next stage, we apply a 3D Kalman filter. The result of the smoothing process for a simple trajectory which was acquired in real time is shown in Fig. 5. The original trajectory is marked with red points and the smoothed one with blue ones.

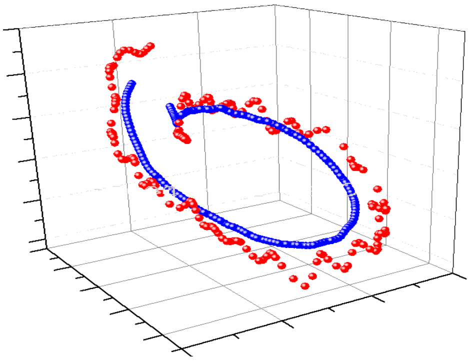

Fig. 5. Trajectory smoothing.

VI. 3D Hand Gestures Recognition

Within the framework of our system, the hand gestures are used for two different purposes, as follows: to manipulate different objects in the 3D virtual environment and to initiate/complete a specific action. As for the first purpose, we have implemented a 3D virtual mouse which operates in the virtual environment and is entirely controlled through hands. In order to be able to deal with the three basic operations (hand movement within the virtual environment, selecting/moving an object and leaving the selected object at the current coordinates of the virtual mouse) we have used 3 different 3D virtual models of the hand, as shown in the Fig. 6. An example of how an object can be manipulated within the virtual environment is given in Fig. 2.

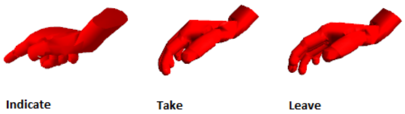

Fig. 6. 3D virtual models of the hand.

To cope with the second goal (selection of an action) we have first created a dictionary of 8 different gestures and then created a training set by acquiring 300 sample hand trajectories for each gesture. These are the reference trajectories underlying the hand gestures recognition. Each trajectory in the training set is represented as follows

$$R_k = \left[C_k, \left\{P_1, P_2, ..., P_m\right\}\right], \qquad (3)$$

where C_K is the gesture represented by the trajectory; $\left\{P_1, P_2, ..., P_m\right\}$ are the trajectory points.

The dictionary contains the following gestures:

1) START INTERACTION - initiates the interaction with the virtual environment so that the human operator can start handling the 3D objects;
2) STOP INTERACTION – stops the interaction and sends all the commands to the industrial robot;
3) START/STOP ROTATION – initiates/stops the rotation of the virtual environment;
4) START/STOP ZOOM – initiates/stops the zooming of the virtual environment;
5) SELECT OBJECT – selects an object from the virtual environment;
6) PLACE OBJECT – places the selected object to the current coordinates of the virtual mouse.

The gesture recognition problem, which ultimately reduces to that of trajectories' matching, raises few important issues, as follows:

1) Gestures generally differ in regard to the starting position. When making the same gesture several times, the starting position will be always different, which means each time there will be a different offset $\left[t_x, t_y, t_z\right]$ to the starting position of the reference trajectory. The offset will propagate through all the points in the trajectory and therefore will affect the trajectories comparison result. The higher is the offset, the bigger will be the distance between two trajectories which correspond to the same gesture. Generally, this problem can be solved through a simple 3D translation as follows

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_Y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}; \qquad (3)$$

2) Gestures are performed at different speeds. This issue affects only those systems in which the machine being controlled by gestures needs to operate at different speeds in accordance with the actual velocity of the hands being tracked;

3) Different lengths of trajectories. The length of a trajectory acquired in real-time will always be different from that of the reference trajectory. This problem can be solved through a 3D scale transformation as follows

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \qquad (4)$$

where S_x, S_y and S_z are the scale factors for each of the 3 axes.

The hand gestures recognition method that we have designed and developed uses the Dynamic Time Warping (DTW) algorithm for trajectories matching (ensuring time and speed invariance) and the k-nearest neighbor (k-NN) algorithm for classification. Given two trajectories, T of length n and R of length m (the first one acquired in real time and the second one, a reference trajectory from the gesture dictionary), the DTW algorithm will compute the similarity between R and T, by finding an optimal match between their corresponding points.

In the first step, the algorithm computes the distance matrix, as follows

$$DTW = \begin{bmatrix} d(T_1,R_1) & d(T_1,R_2) & \dots & d(T_1,R_m) \\ d(T_2,R_1) & d(T_2,R_2) & \dots & d(T_2,R_m) \\ \dots & \dots & \dots & \dots \\ d(T_n,R_1) & d(T_n,R_2) & \dots & d(T_n,R_m) \end{bmatrix}, \quad (5)$$

where $d(T_i,R_j)$ is the Euclidean distance between the points T_i and R_j. Based on this matrix, in the next step, the DTW algorithm will find the best path (with the minimum cost) which starts at $DTW[1,1]$ and ends at $DTW[n,m]$. The algorithm imposes the following constraint: the path, which is a sequence of matching points, needs to be monotonically and continuous. Thus, at each step the new DTW matrix will be computed as follows

$$DTW_{[i,j]} = DTW_{[i,j]} + \min\begin{pmatrix} DTW_{[i-1,j]} \\ DTW_{[i,j-1]} \\ DTW_{[i-1,j-1]} \end{pmatrix}. \quad (6)$$

When the algorithm finishes, the minimum cost will be find in $DTW[n,m]$. The trajectories' matching is illustrated in Fig. 7.

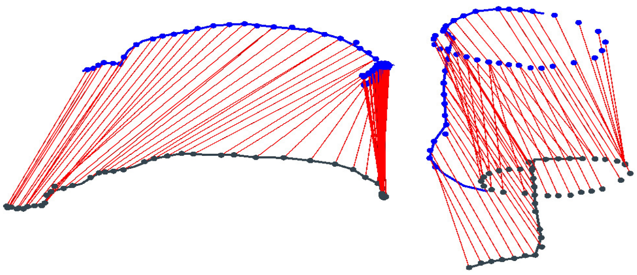

Fig. 7. Examples of DTW matching.

As we already mentioned, for trajectory's classification (gesture selection) we have used the k-NN algorithm which runs against the training set described in (2). First, the DTW algorithm determines the best match between T and each R_i trajectory belonging to the training set. Then, the k-NN algorithm selects the first K (usually, K=10) trajectories having the shortest distance to T. Finally, the algorithm chooses that gesture which is most frequent among the first K trajectories.

VII. THE INDUSTRIAL ROBOT

The assembly cell consists of a Sony SRX-611 industrial robot (Fig. 8) with 4 degree of freedom and a set of additional devices (used for transportation and fastening purposes) controlled by an Omron C200HX Programmable Logic Controller.

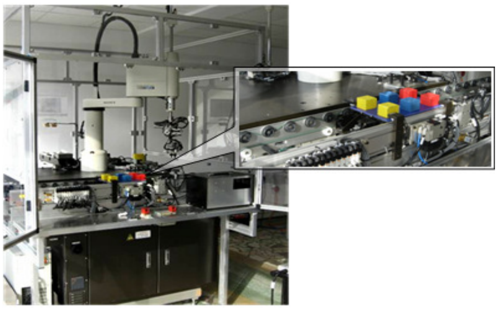

Fig. 8. Sony SRX-611 industrial robot.

The main system (comprising of a 3D virtual environment and a hand gestures recognition module) communicates with the cell's controllers via a hardware independent interface running on an embedded device of Elsist Netmaster type. The interface has the role to abstract the communication with controllers, providing the main system with a set of high level commands. These commands are structured as XML elements, in which the element's tag represents the command type and its attributes contain the associated data. For example, the command for moving a cuboid from a certain location to another one may have the following form: *<Move Source="1" Dest="2" />.*

The embedded device on which the hardware independent interface runs can be seen as an additional controller attached to the cell ensuring both physical communication interfacing as well as the logical one [13]. The hardware independent interface (HII) was developed using an IEC 61499 modelling approach. The HII was developed as a Service Interface Function Block (of HII_Sony_Cell type), encapsulated along with several communication function blocks into a composite FB. Fig. 9 presents the content of this FB.

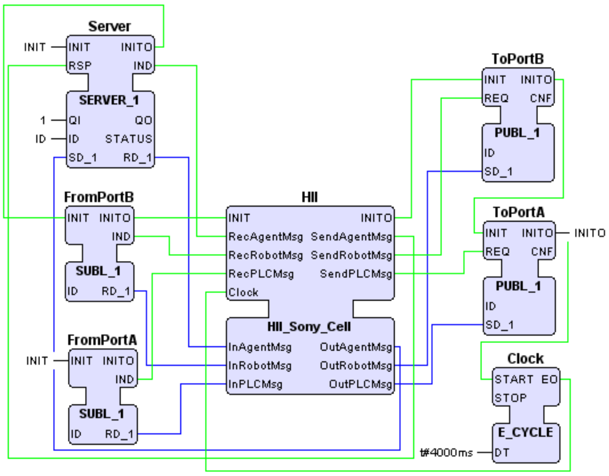

Fig. 9. Content of the composite FB encapsulating the hardware independent interface.

The Server FB allows receiving and sending of messages from/to the main system, while the FBs pairs {*ToPortA*, *FromPortA*} and {*ToPortB*, *FromPortB*} allows the HII to communicate with the cell's controllers via the serial ports (*A* and *B*) of Netmaster.

VIII. CONCLUSIONS

The HRI based system that we have designed and developed promotes the use of hand gestures for a more natural interaction between humans and industrial robots. The interaction's goal is not to move the robot or its mobile arm from one place to another, but to control the robot in such a manner so that it will be able to accomplish certain tasks. The interaction is done via a dynamically created 3D virtual environment. The 4 major advantages of our approach are:

1) It eliminates the dead-times which are specific to those systems which promotes a direct interaction;
2) The human operator does not need to be in the proximity of the industrial robot;
3) A sequence of operations which simulates an industrial process flow needs to be done only once. After that, the process can take place numerous times, within the robot's working environment;
4) It can be extended, so as to accomplish more complex tasks (e.g.: moving and placing containers in a real food warehouse).

Future work will be focused on the design and development of a more complex system which will allow users to manipulate different mechanical parts and to build real products, only through hand gestures.

REFERENCES

[1] C. Ankit, J. Raheja, D. Karen, R. Sonia, "Intelligent Approaches to interact with Machines using Hand Gesture Recognition in Natural way: A Survey", *International Journal of Computer Science & Engineering Survey (IJCSES),* vol. 2, no. 1, pp. 122–133, 2011. [Online]. Available: http://dx.doi.org/10.5121/ijcses.2011.2109

[2] G. Pragati, A. Naveen, S. Sanjeev, "Vision Based Hand Gesture Recognition", *World Academy of Science, Engineering and Technology,* no. 49, pp. 972–977, 2009.

[3] S. Indra, K. Gama, *Gesture Recognition Aplication based on Dynamic Time Warping (DTW) FOR Omni-Wheel Mobile Robot,* EEPIS Final Project, 2011.

[4] M. Hahn, L. Kruger, C. Wohler, F. Kummert, "3D Action Recognition in an Industrial Environment", in *Proc. of the 3rd International Workshop on Human-Centered Robotic Systems (HCRS 09),* Bielefeld, Germany, 2009, pp. 141–150. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-10403-9_15

[5] Y. Yang, Y. Li, "A New Descriptor for 3D Trajectory Recognition", in *Proc. of the Ninth International Symposium on Operations Research and Its Applications,* China, 2010.

[6] J. Agnes, M. Sebastien, B. Olivier, V. Jean-Emmanuel, "HMM and IOHMM for the Recognition of Mono- and Bi-Manual 3D Hand Gestures", *FG Net Workshop on Visual Observation of Deictic Gestures,* 2004.

[7] M. Van den Bergh, D. Carton, R. De Nijs, N. Mitsou, C. Landsiedel, et al., "Real-time 3D hand gesture interaction with a robot for understanding directions from humans", in *Proc. of the IEEE International symposium on robot and human interactive communication,* 2011, pp. 357–362.

[8] P. Chang-Beom, R. Myung-Cheol Roh., L. Seong-Whan, "Real-time 3D pointing gesture recognition in mobile space", in *Proc. of the 8th IEEE International Conference on Automatic Face & Gesture Recognition,* 2008, pp. 1–6.

[9] M. Van den Bergh, L. Van Gool, "Combining RGB and ToF cameras for real-time 3D hand gesture interaction", in *Proc. of the IEEE Workshop on Applications of Computer Vision,* 2011, pp. 66–72.

[10] C. Jun, X. Can, B. Wei, T. Dacheng, "Feature fusion for 3D hand gesture recognition by learning a shared hidden space", *Pattern Recognition Letters,* vol. 33, no. 4, pp. 476–484, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.patrec.2010.12.009

[11] Y. Ruifeng, C. Jun, L. Pengcheng, C. Guang, X. Can, X. Qi, "View invariant hand gesture recognition using 3D trajectory", in *Proc. of the 8th World Congress on Intelligent Control and Automation (WCICA),* 2011, pp. 6315–6320.

[12] T. Cerlinca, G. Pentiuc, "Robust 3D Hand Detection for Gestures Recognition", *Studies in Computational Intelligence,* vol. 2012, no. 382, pp. 259–264, 2012.

[13] V. Vlad, A. Graur, C. Popa, "Models and concepts for integration of classical manufacturing systems into holonic systems", in *Proc. of the 3rd International Symposium on Electrical Engineering and Energy Converters,* Suceava, 2009, pp. 131–136.