DOI: 10.15439/2015F166
ACSIS, Vol. 5

Enhanced movement tracking with Kinect supported by high-precision sensors

Tomasz Pałys
Cybernetics Faculty at the Military University of Technology
ul. S. Kaliskiego 2,
00-908 Warsaw, Poland
Email: tomasz.palys@wat.edu.pl

Witold Żorski
Cybernetics Faculty at the Military University of Technology
ul. S. Kaliskiego 2,
00-908 Warsaw, Poland
Email: wzorski@wat.edu.pl

***Abstract*—The paper presents a proposal of sensor fusion combining data derived from a Kinect device and high-precision sensors. The main idea was to enhance the tracking of a human arm in order to obtain precise coordinates. The Kinect plays the role of a calibration device and sensors' data are used in kinematics equations for enhanced tracking of the arm. This way the resulting information has less uncertainty and can be directly transmitted to another system, e.g. robotics one. The system has been implemented using kinematics-based approach with transformation operators as well as quaternions. In terms of accuracy the main result is that the Kinect performs very similar with sensors.**

I. Introduction

THE idea of creating a device that allows to recognize human body shape in details without having to hold any remote controller on it inspired Microsoft to create the Kinect device (2010). It was a very novel and extraordinary approach to the challenge. Soon, the Kinect attracted researchers in many fields, especially from robotics and computer vision, as well as some communities that have released various SDK for Kinect allowing to use it as a kind of measurement tool. Since the lunch of the Kinect for Windows a lot of papers have been devoted to the scientific application of this device.

Some papers may serve as tutorials, e.g. for calibration of the Kinect imaging sensor [12], or help while building a new Kinect-based system [9], or source of references [4] for Kinect-based computer vision researchers, covering topics which include: preprocessing, object tracking and recognition, human activity analysis, hand gesture analysis, and indoor 3-D mapping, and many others.

In the beginning, researchers presented novel approaches to human detection using depth information taken by the Kinect [21], methods of obstacles detection where the Kinect was used as a capturing device [14], methods to quickly and accurately predict 3D positions of body joints from a depth image [17].

Later, considered problems became more complex or sophisticated. The Kinect was used to evaluate noise, accuracy, resolution, and latency of tracking skeletons; and measure the range in which the person being tracked must be in order to achieve these values [7]. Another tested application of the Kinect was a virtual bilateral Man-Machine interaction through the Kinect sensors, allowing the user to control and monitor Windows programs by gestures without a need of peripherals use [1].

There are even attempts to use the Kinect in the field of medicine or clinical rehabilitation as a monitoring device [18]. Another interesting use of the Kinect are methods for calibration using streaming information from depth cameras [19]. This is still a challenge in the case of hands generic precision and agility due to complexity of fingers joints.

As we consider the Kinect, the area of robotics is very popular in some purposes, for instance, enhancing the navigation of mobile robots, object detection and recognition, scene reconstruction, 3D inspection and others. The good example is an automated vehicles inspection by scanning and dynamically exploring regions of interest over an automotive vehicle body under visual guidance [11]. In such applications the very important aspect is quality of sensors [10] responsible for 3D data acquisition [13].

At present, there are many available solutions dedicated to context-aware robotics system [8], [16], e.g. for orientation determination of pedestrian motions by using an inertial measurement unit module and a complementary separate-bias Kalman filter [22].

In this paper we would like to propose a system based on the Kinect assisted by a set of high-precision sensors, that would be responsible for enhance movement tracking of a human arm with the accuracy suitable in the field of robotics. The idea of such a task is presented in Fig. 1, where we can see a man wearing sensors – one on the waistline and three on the natural links of his right arm. The arm's configuration is mapped by the system. The Fig. 2. gives us a general view about the system and its components.

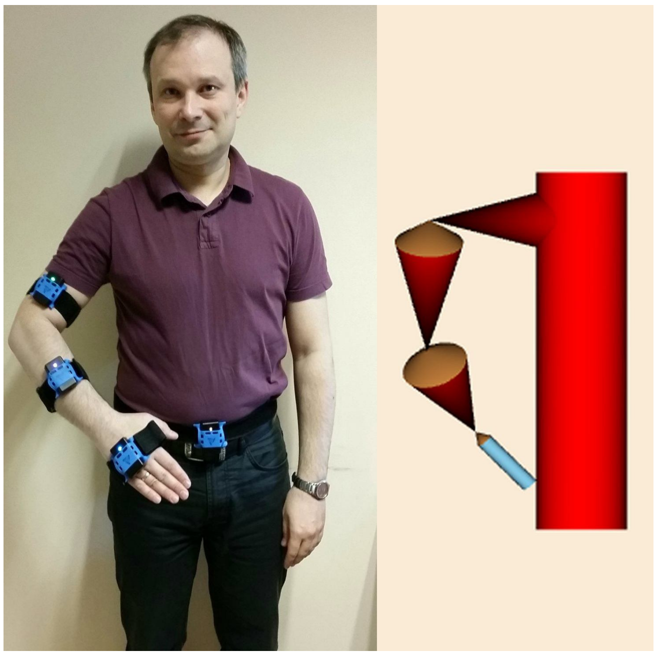

Fig. 1. The idea of mapping a human arm

II. Presentation of the System

The used computer vision system consists of an x86 computer equipped with the Kinect for Windows v2 device [23], which is supported by the YEI 3-Space sensors [24]. Microsoft Visual Studio 2012 and .NET Framework 4.5 are on the basis of applied software environment. Fig. 2 shows a visual conception of the used computer vision system.

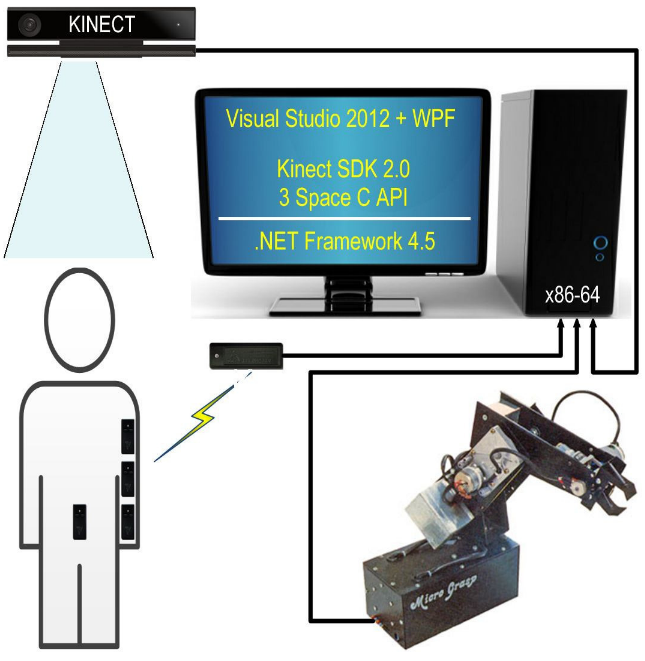

Fig. 2. The system conception

The presented system is based on the second-generation Kinect for Windows (well described in [3]) with an improved motion tracking functionality over its predecessor, a wider field of view, a high definition camera, and the ability to track up to six bodies at once. The Kinect can capture audio, color, and depth environment features, and process the depth data to generate skeleton data.

The YEI 3-Space Sensor (Fig. 3) is a miniature, high-precision, high-reliability, Attitude and Heading Reference System (AHRS) / Inertial Measurement Unit (IMU) with USB 2.0 communication interfaces in a single unit. The AHRS/ IMU uses a gyroscope, an accelerometer, and compass sensors in conjunction with advanced processing and on-board quaternion-based algorithms to determine orientation relative to an absolute reference or relative to a designated reference orientation in real-time. The gradient descent calibration process and high update rates increase accuracy and greatly reduce and compensate errors. The YEI 3-Space Sensor USB unit features are accessible via a well-documented open communication protocol that allows access to all sensor data and configuration parameters. Versatile commands allow access to raw sensor data, normalized sensor data, and filtered absolute and relative orientation outputs in formats incl. quaternions and Euler angles (pitch/roll/yaw). We a going to use both, Euler angles and quaternions, in order to compare results and also to verify our implementation this way.

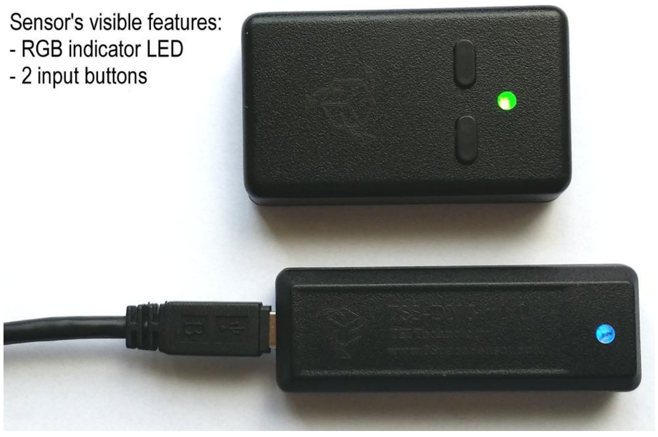

Fig. 3. The YEI 3-Space Sensor and the communication dongle

III. Software Environment

Considering Visual Studio 2012 as the primal software environment we can distinguish three essential layers of the system: “application layer” on the top, “management layer” in the middle, and “data layer” at the bottom. The structure of the used software is presented in Fig. 4.

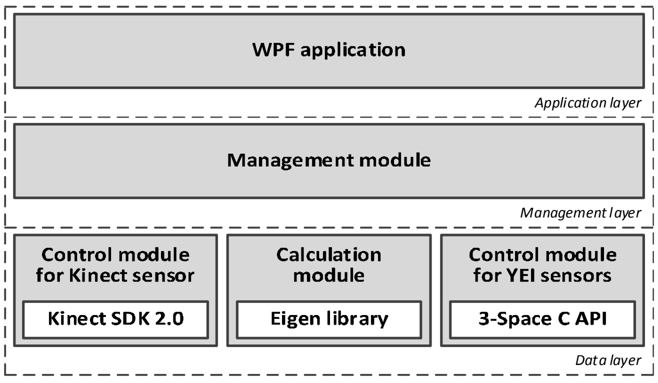

Fig. 4. The structure of the software environment

The application layer stands for the created WPF application [15]. It is the main element responsible for every activity aspect of the system, particularly for the GUI.

The management layer includes management control system tools placed in the management module library.

The set of libraries on the bottom layer includes: a control module for the Kinect sensor (it uses Kinect SDK 2.0), a calculation module (it uses the Eigen library ver. 3.2.2), and a control module for YEI sensors (it uses 3-Space C API ver. 2.0.6).

A. Microsoft Visual Studio as a development tool

The use of Visual Studio 2012 as a development tool was involved by the possibility to fully utilize the Kinect for Windows SDK and assets of the Windows Presentation Foundation that provides a consistent programming model for building applications with a graphical subsystem. An additional aspect was the integration possibility of Microsoft .NET Common Language Runtime (managed code) with the 3-Space C API library and the Eigen library (unmanaged code).

B. The Software Development Kit for Kinect

The Kinect for Windows SDK enables to create applications under Visual Studio 2012 (.NET Framwork 4.5 is required) that support advanced gesture and voice recognition using Kinect sensor technology on computers running modern Windows. To develop speech-enabled Kinect applications, the Microsoft Speech Platform SDK must be installed.

C. The Eigen library

The Eigen is a high-level C++ open source library of template headers for linear algebra, matrix and vector operations, numerical solvers; this facilitates to achieve high performance of required calculations. In particular, the Eigen has also a possibility to solve linear systems using several types of decompositions, and includes several very useful classes and functions for image processing.

D. The 3-Space API

The use of the 3-Space API was inevitable due to YEI sensors. This API is available for C/C++ and Python languages and can be used to write independent software for YEI sensors. Because of the chosen Visual Studio environment the 3-Space C API has been adopted.

IV. KINECT'S ROLE

As mentioned earlier and suggested in Fig. 1, the assumption is that the system is able to perform tracking of a human arm with the use of Kinect and sensors. To be exact, the Kinect is use mostly at the stage of calibration and later for verification purposes. In short, to perform the calibration of the system a person wearing four sensors have to take a posture shown in Fig. 5. (the arm must be aligned to ensure proper calibration), and next press a button on sensor S_1. At this moment sensors are reset, and the system reads coordinates of points P_1 to P_4 from the Kinect. Starting from this point, the Kinect plays the role of the secondary sensor, suitable for verification purposes.

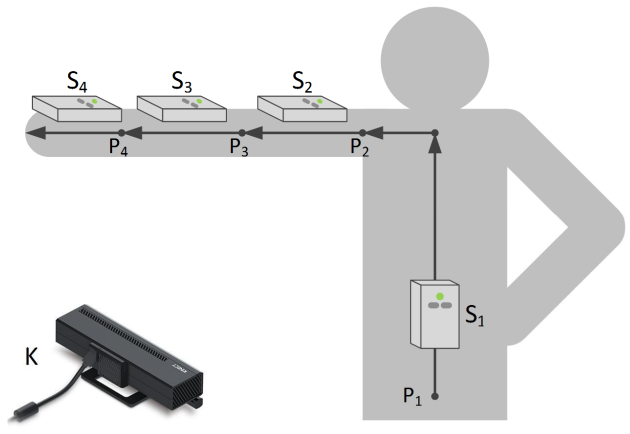

Fig. 5. The calibration of the system

V. KINEMATICS SOLUTIONS

The use of sensors affixed to a human stature involves a need for an adequate mathematical instrument appropriate to perform required transformations on received data. The obvious one derives from robotics, where the kinematics plays the role of a fundamental description tool [6].

In order to describe the location of each link (a person's body part) relative to its neighbors a frame (a coordinate system) has been affixed to each link; see Fig. 6.

A. Transformation operators

In the field of robot kinematics transformation operators allow simple and clear matrix representation of rotations and translations [2]. Let us start from coordinates of point P_1 (see Fig. 5) received from the Kinect during calibration procedure:

$$^{K}P_1 = [p_1^x, p_1^y, p_1^z, 1]^T . \qquad (1)$$

Next, we can use other coordinates (points P_2, P_3, P_4) received from the Kinect to calculate crucial distances between sensors (see Fig. 6):

$$\begin{cases} x_{21} = p_2^x - p_1^x \\ y_{21} = p_2^y - p_1^y \end{cases}, \quad x_{32} = p_3^x - p_2^x, \quad x_{43} = p_4^x - p_3^x. \qquad (2)$$

Now, we can create transformation operators that are adequate to the situation presented in Fig. 6. The assumption is, that we are able to receive Euler angles (α,β,γ) from sensors. The next thing is that we use differences of corresponding angles, e.g. $\alpha_3 = \alpha_{S3} - \alpha_{S2}$.

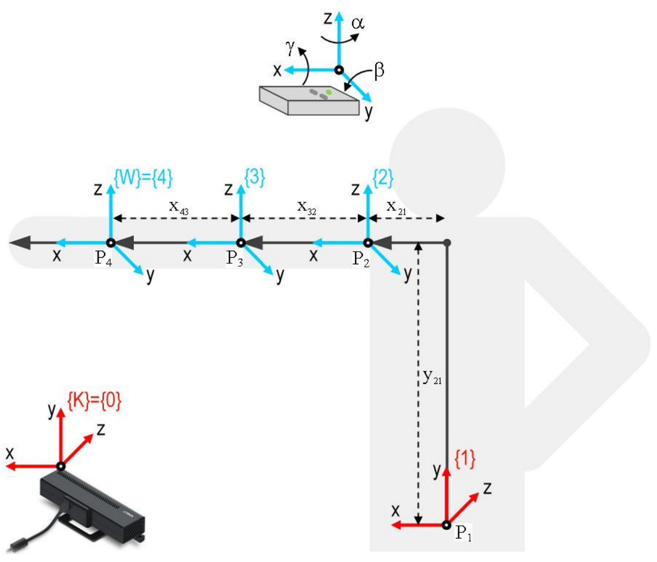

Fig. 6. A model of the human skeleton as a kinematic chain (see Fig. 5)

Taking into account our assumptions as to angles the set of transformation operators is as follows:

$$ {}^0_1T = {}^K_1T = \begin{bmatrix} 1 & 0 & 0 & p_1^x \\ 0 & 1 & 0 & p_1^y \\ 0 & 0 & 1 & p_1^z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3)$$

$$ {}^1_2T = \begin{bmatrix} 1 & 0 & 0 & x_{21} \\ 0 & \cos(-\pi/2) & -\sin(-\pi/2) & y_{21} \\ 0 & \sin(-\pi/2) & \cos(-\pi/2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} c\alpha_2 c\beta_2 & c\alpha_2 s\beta_2 s\gamma_2 - s\alpha_2 c\gamma_2 & c\alpha_2 s\beta_2 c\gamma_2 + s\alpha_2 s\gamma_2 & 0 \\ s\alpha_2 c\beta_2 & s\alpha_2 s\beta_2 s\gamma_2 + c\alpha_2 c\gamma_2 & s\alpha_2 s\beta_2 c\gamma_2 - c\alpha_2 s\gamma_2 & 0 \\ -s\beta_2 & c\beta_2 s\gamma_2 & c\beta_2 c\gamma_2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (4)$$

$$ {}^2_3T = \begin{bmatrix} c\alpha_3 c\beta_3 & c\alpha_3 s\beta_3 s\gamma_3 - s\alpha_3 c\gamma_3 & c\alpha_3 s\beta_3 c\gamma_3 + s\alpha_3 s\gamma_3 & x_{32} \\ s\alpha_3 c\beta_3 & s\alpha_3 s\beta_3 s\gamma_3 + c\alpha_3 c\gamma_3 & s\alpha_3 s\beta_3 c\gamma_3 - c\alpha_3 s\gamma_3 & 0 \\ -s\beta_3 & c\beta_3 s\gamma_3 & c\beta_3 c\gamma_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (5)$$

$$ {}^3_4T = \begin{bmatrix} c\alpha_4 c\beta_4 & c\alpha_4 s\beta_4 s\gamma_4 - s\alpha_4 c\gamma_4 & c\alpha_4 s\beta_4 c\gamma_4 + s\alpha_4 s\gamma_4 & x_{43} \\ s\alpha_4 c\beta_4 & s\alpha_4 s\beta_4 s\gamma_4 + c\alpha_4 c\gamma_4 & s\alpha_4 s\beta_4 c\gamma_4 - c\alpha_4 s\gamma_4 & 0 \\ -s\beta_4 & c\beta_4 s\gamma_4 & c\beta_4 c\gamma_4 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (6)$$

Thus, the link transformations can be multiplied together to find the single transformation that relates frame {W} to frame {K}:

$$ {}^K_W T = {}^0_4T = {}^0_1T\, {}^1_2T\, {}^2_3T\, {}^3_4T \qquad (7)$$

Finally, we can calculate coordinates of points from P_2 to P_4 with respect to frame {K} in the following way:

$$\begin{cases} {}^K P_2 = {}^0_1T\, {}^1_2T \cdot [0,0,0,1]^T \\ {}^K P_3 = {}^0_1T\, {}^1_2T\, {}^2_3T \cdot [0,0,0,1]^T \\ {}^K P_4 = {}^K_W T \cdot [0,0,0,1]^T \end{cases} \qquad (8)$$

In the case of a robotics system we will prefer coordinates of point P_4 (i.e. for the wrist) with respect to frame {1} that can be calculated in the following way:

$$ {}^1 P_4 = {}^1_2T\, {}^2_3T\, {}^3_4T \cdot [0,0,0,1]^T \qquad (9)$$

This is the case when the arm is fully independent of the Kinect as long as the calibration is done.

B. The quaternions

The quaternions [5], a number system that extends the complex numbers, are members of a noncommutative division algebra. Dual-quaternions may formulate and solve a problem more concisely, more rapidly and in fewer steps, with result more plainly to others, and practice with fewer lines of code effortlessly debugged. There is no loss of efficiency – dual-quaternions can be just as efficient (if not more efficient) than using matrix methods. Additionally, there are some very important reasons for using dual-quaternions: singularity-free, un-ambiguous, shortest path interpolation, etc. This mathematical tool becomes very popular in the computer world, especially in computer graphics – even the used Kinect and sensors are quaternion-based devices. In a general case dual-number quaternions [20] allow compact representation of both rotations and translations. This mathematical tool was integrated into the current system with little coding effort.

In short, a dual-quaternion consists of two quaternions: q_r, q_d (eight elements). These two quaternions are called the real part and the dual part:

$$q = q_r + q_d \varepsilon , \qquad (10)$$

where ε is an additional dual number.

The dual-quaternion can represent only rotation by angle φ if the dual part is set to zero:

$$q_R = [\cos(\frac{\varphi}{2}) - (\hat{v}_x i + \hat{v}_y j + \hat{v}_z k)\sin(\frac{\varphi}{2})][0,0,0,0] , \qquad (11)$$

where $\hat{v}$ is a unit axis.

Analogously, to represent only translation by vector $[t_x, t_y, t_z]$, the real part is set to an identity and the dual part represents the translation:

$$q_T = [1,0,0,0][0, \frac{t_x}{2}, \frac{t_y}{2}, \frac{t_z}{2}] . \qquad (12)$$

Now, combining the rotational and translational transforms into a single unit quaternion to represent a rotation followed by a translation we get:

$$q = q_T \times q_R . \qquad (13)$$

The following equation defines how we can transform point p into point p', using the received unit dual-quaternion:

$$p' = q \cdot p \cdot q^* , \qquad (14)$$

where q and q^* represent a dual-quaternion transform and its conjugate.

In the considered system we can directly use quaternions received from the Kinect and sensors. However, if we have only Euler angles (α, β, γ), there is an easy way to receive the quaternion responsible for the rotation (see Eq. 11):

$$\begin{cases} q_x = \cos(\gamma/2) - i\sin(\gamma/2) \\ q_y = \cos(\beta/2) - j\sin(\beta/2) \\ q_z = \cos(\alpha/2) - k\sin(\alpha/2) \end{cases}$$
$$q_R = [q_z \cdot q_y \cdot q_x][0,0,0,0] . \qquad (15)$$

VI. Results

Results we obtained seem to be quite satisfying. Both tools, transformation operators and quaternions, were independently implemented and gave exactly the same values, what may be considered as a confirmation of a proper solution (implementation). Additionally, the coordinates received from the Kinect are concurrent with those calculated by the system – this has been checked in many cases; a sample is presented in Fig. 7–9, respectively for coordinates X, Y, and Z. The considered figures present wrist coordinates (the values are in metres) received during a person activity for an example period of 8 seconds.

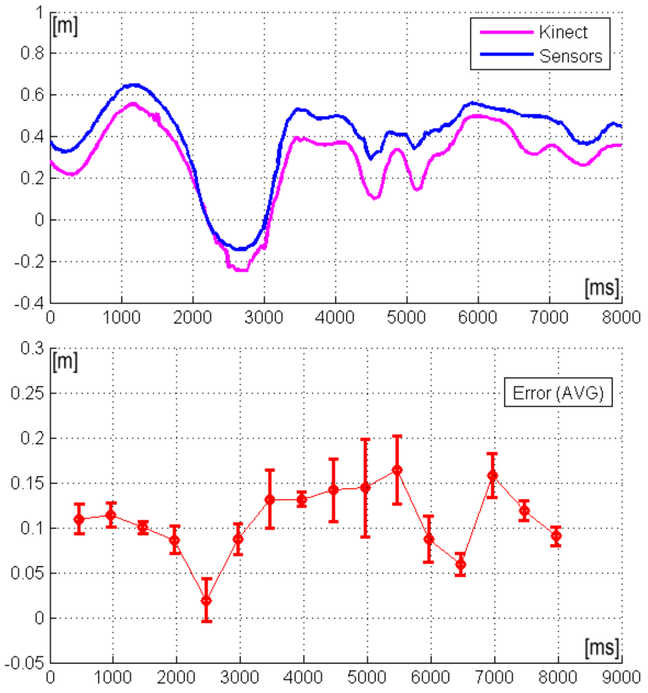

Fig. 7. Wrist coordinate X – values and the average error

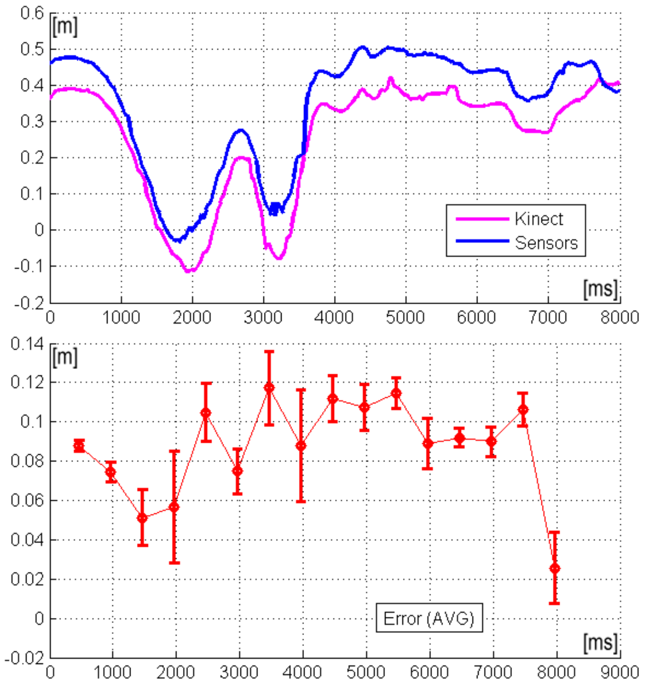

Fig. 8. Wrist coordinate Y – values and the average error

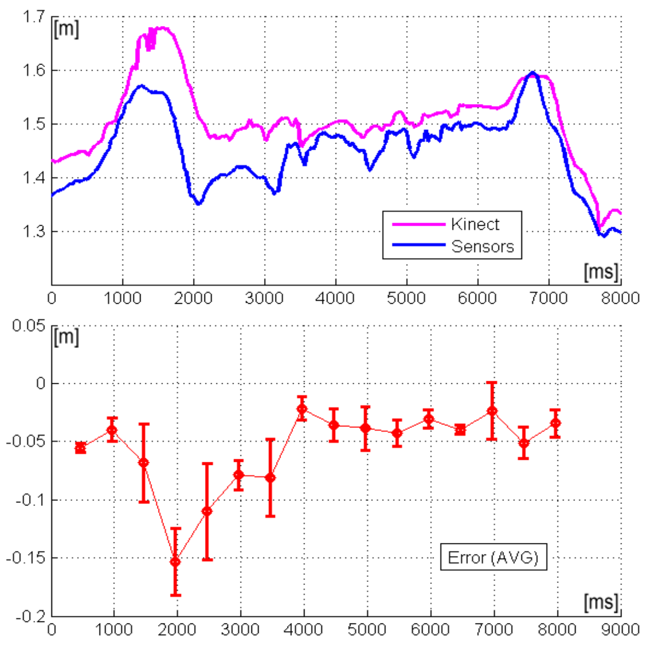

Fig. 9. Wrist coordinate Z – values and the average error

VII. Conclusion

Authors are aware that the idea of the Kinect device supported by additional sensors is a strong negation of the Kinect nature. Nevertheless, we need to accent that the Kinect served mainly as a fine calibration device and for some verification purposes. Additionally, the Kinect fails for some "configurations" of a human posture, environment objects are serious obstacles, and the space range of the device is very narrowed. This set of adversities has been reduced by the use of the set of four high-precision sensors.

Some gains of the work are worth to be mentioned as advantages of the system: a high level of accuracy limited mainly by the process of calibration, immunity to terrain obstacles, and wide operating range.

We plan to improve the system accuracy, further simplify and revamp the process of calibration, and remove some technical limitations. We want to use only quaternions, even in the case of data received from the Kinect and sensors as they are native for these devices. We would like to perform some robotics experiments in order to face unknown.

References

[1] Andaluz V., Gallardo C., Santana J., Villacres J., *Bilateral Virtual Control Human-Machine with Kinect Sensor*, Andean Region International Conference, 2012, pp. 101-104. http://dx.doi.org/10.1109/Andescon.2012.32

[2] Craig J. J., "*Introduction to Robotics: mechanics and control*", 2nd ed., Addison-Wesley Publishing Company, 1989. http://dx.doi.org/10.1016/0005-1098(87)90105-1

[3] Fankhauser P., et al., *Kinect v2 for Mobile Robot Navigation: Evaluation and Modeling*, IEEE International Conference on Advanced Robotics, 2015 (the paper is accepted).

[4] Jungong H., Ling S, Dong Xu, Shotton J., *Enhanced Computer Vision with Microsoft Kinect Sensor: A Review*, IEEE Transactions on Cybernetics, Vol.43(5), 2013, pp. 1318-1334. http://dx.doi.org/10.1109/TCYB.2013.2265378

[5] Kenwright B., *A Beginners Guide to Dual-Quaternions: What They Are, How They Work, and How to Use Them for 3D*, WSCG 2012 Communication Proceedings, pp.1-13.

[6] Kim Jung-Ha, Kumar V. R., *Kinematics of robot manipulators via line transformations*, Journal of Robotic Systems, Vol.7(4), 1990, pp. 649–674. http://dx.doi.org/10.1002/rob.4620070408

[7] Livingston M., Sebastian J., Zhuming Ai, Decker J., *Performance measurements for the Microsoft Kinect skeleton*, Virtual Reality Short Papers and Posters, 2012, pp. 119-120. http://dx.doi.org/10.1109/VR.2012.6180911

[8] Luo R.C., Bo-Han Shih, Tsung-Wei Lin, *Real time human motion imitation of anthropomorphic dual arm robot based on Cartesian impedance control*, IEEE International Symposium on Robotic and Sensors Environments, 2013, pp. 25-30. http://dx.doi.org/10.1109/ROSE.2013.6698413

[9] Ming A., Enomoto K., Shinozaki M., Sato R., Shimojo M., *Development of an entertainment robot system using Kinect*, 8th Europe-Asia Congress on Mechatronics, 2014, pp. 127-132. http://dx.doi.org/10.1109/MECATRONICS.2014.7018621

[10] Murawski K., *Measurement of Membrane Displacement with a Motionless Camera Equipped with a Fixed Focus Lens*, Metrology and Measurement Systems, Vol.22(1), 2015, pp. 69-78. http://dx.doi.org/10.1515/mms-2015-0011

[11] Nakhaeinia D., Fareh R., Payeur P., Laganiere R., *Trajectory planning for surface following with a manipulator under RGB-D visual guidance*, IEEE International Symposium on SSRR 2013, pp. 1-6. http://dx.doi.org/10.1109/SSRR.2013.6719365

[12] Pagliari D., Menna F., Roncella R., Remondino F., Pinto L., *Kinect Fusion improvement using depth camera calibration*, International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XL-5, 2014, pp. 479-485. http://dx.doi.org/10.5194/isprsarchives-XL-5-479-2014

[13] Pinto A., Costa P., Moreira A., Rocha L., *Evaluation of Depth Sensors for Robotic Applications*, IEEE International Conference on Autonomous Robot Systems and Competitions, 2015, pp. 139-143. http://dx.doi.org/10.1109/ICARSC.2015.24

[14] Rakprayoon P., *Kinect-based obstacle detection for manipulator*, IEEE/SICE International Symposium on System Integration (SII), 2011, pp. 68-73. http://dx.doi.org/ 10.1109/SII.2011.6147421

[15] Ren Yu, Gubing Lu, Feng Lu, *A Method of Rotation Transformation for 3D Object by Changing Camera Attributes in WPF*, International Conference on Information Engineering and Computer, 2010, pp. 1-4. http://dx.doi.org/10.1109/ICIECS.2010.5678267

[16] Rosado J., Silva F., Santos V., Zhenli Lu, *Reproduction of human arm movements using Kinect-based motion capture data*, IEEE International Conference on Robotics and Biomimetics, 2013, pp. 885-890. http://dx.doi.org/10.1109/ROBIO.2013.6739574

[17] Shotton J., Fitzgibbon A., Cook M., Sharp T., *Real-time human pose recognition in parts from single depth images*, IEEE Conference on Computer Vision and Pattern Recognition, 2011, pp. 1297-1304. http://dx.doi.org/10.1109/CVPR.2011.5995316

[18] Tao G., Archambault P. S., Levin M. F., *Evaluation of Kinect skeletal tracking in a virtual reality rehabilitation system for upper limb hemiparesis*, International Conference on Virtual Rehabilitation, 2013, pp. 164-165. http://dx.doi.org/10.1109/ICVR.2013.6662084

[19] Vicente A., Faisal A., *Calibration of kinematic body sensor networks: Kinect-based gauging of data gloves "in the wild"*, IEEE International Conference on Body Sensor Networks, 2013, pp. 1-6. http://dx.doi.org/10.1109/BSN.2013.6575526

[20] Walker M. W., Shao L., Volz R. A., *Estimating 3-D location parameters using dual number quaternions*, CVGIP: Image Understanding Vol.54(3), 1991, pp. 358-367. http://dx.doi.org/10.1016/1049-9660(91)90036-O

[21] Xia Lu, Chen Chia-Chih, Aggarwal J. K., *Human detection using depth information by Kinect*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2011, pp. 15-22. http://dx.doi.org/10.1109/CVPRW.2011.5981811

[22] Zhang R., Reindl L., *Pedestrian motion based inertial sensor fusion by a modified complementary separate-bias Kalman filter*, Sensors Applications Symposium, 2011, pp. 209-213. http://dx.doi.org/10.1109/SAS.2011.5739766

[23] https://www.microsoft.com/en-us/kinectforwindows/

[24] http://www.yeitechnology.com/yei-3-space-sensor