RGB-D Sensors in Social Robotics

Submitted: 16th October 2014; accepted: 10th December 2014

Michał Dziergwa, Paweł Kaczmarek, Jan Kędzierski

DOI: 10.14313/JAMRIS_1-2015/3

Abstract:
This article presents the results of a series of experiments carried out using a Kinect for Windows sensor coupled with dedicated software. The focus of this study is on the use of such devices in the field of social robotics. Two software packages are considered - Microsoft Kinect SDK and OpenNI coupled with NiTE library. Particular emphasis is placed on the parameters affecting the social competencies of a robot, such as the speed of detecting users, the accuracy of establishing position and orientation of a user or stability of the tracking process. Key characteristics of the evaluated software packages are identified and differences regarding their usage outlined in view of interaction oriented algorithms.

Keywords: *RGB-D sensors, social robotics, perception, human-robot interaction*

1. Introduction

Until recently, the perceptive capabilities of social robots were based mainly on RGB cameras. Data which can be extracted from a 2D image, with regard to social robotics applications, include the location and parameters of the users' faces, the color of their clothing or the character of their gestures. The use of RGB cameras, however, is limited due to their high sensitivity to light conditions. In addition, processing high resolution images requires vast amounts of computational power.

Another class of sensors that plays a significant role in social robotics is the distance sensor. This class of sensor encompasses such devices as optical range finders, sonar arrays or laser scanners. Such equipment is used not only for classic navigation, but also allow the robot to maintain a proper proxemic distance during interaction with humans.

3D sensors such as stereo cameras have been available on the market for a long time. However, their usage was limited due to their low resolution and high price. The real revolution started in November 2010, when Microsoft introduced Kinect. In addition to RGB data, this cheap and compact device provides a detailed depth image in the form of a point cloud. Kinect was initially created as a controller for the Xbox 360 but over time it began to be used for research purposes, mainly for human posture and balancing analysis [1], three-dimensional interior mapping [4], gesture detection and recognition [10] or the analysis of 3D facial models [5]. Examples of sensor application in social and service robotics can be found in [12,16]. The demand for an inexpensive depth sensor was so huge that in February 2012 Microsoft introduced a modified development version of the device, called Kinect for Windows. Kinect finally entered the Guinness Book of Records as the fastest selling consumer electronics device.

There has been a multitude of applications of Kinect in the field of social robotics. Speech recognition and analysis of its meaning are still the bottleneck of human-robot interactions and the visual abilities of robots need to complement these deficits. RGB-D data can be used to maintain appropriate proxemic distance and eye contact or to communicate with the robot via prearranged gestures (eg. waving or lifting the hand). Knowledge about the color and shape of objects held in the user's hand can also be useful in the course of the interaction. Tracking of face orientation can be used to determine where the user is staring, which allows the robot to share the user's attention.

The accuracy of RGB-D sensors, especially Kinect, was investigated in a number of articles. Their outcomes confirm the high precision of the sensor and its usefulness in perception of indoor environment. Kinect proved to be free from large systematic errors when compared with a laser scanning data [7]. Its random error increases quadratically and depth resolution decreases quadratically with increasing distance from the sensor, and reaches maximum error of 4 cm and minimum resolution of 7 cm at the maximum range of 5 meters [8]. Both Kinect and Asus Xtion, provide similar accuracy without regard to angle between the optical axis and the test subjects [3].

From the point of view of social robotics, Kinect's ability to accurately detect and track human posture is more important than the precision of determining position of a single point in the image. This article describes the course and results of a study aimed at the verification of Kinect in the context of the robot's social competencies. To achieve this goal, a Kinect sensor was integrated with the control system of the social robot FLASH [2]. The study consisted of several experiments focused on various parameters of the two main software packages allowing the creation of applications oriented towards interaction with humans – Kinect SDK [9] and OpenNI library [11] coupled with NiTE software.

2. Kinect

The following tests were carried out using the Kinect for Windows (as opposed to Kinect for Xbox) version of the sensor. The range of the device is

80–400 cm. It utilizes the structured light method which allows it to obtain an image with a resolution of 300x200 pixels. This image is then interpolated to a 640x480 resolution at a refresh rate of 30 frames per second. The resolution of the sensor's RGB camera is 640×480 at 30 frames per second. It is also possible to obtain an image with a higher resolution (1280x960) but the refresh rate drops to 12 frames per second. An array consisting of 4 microphones is also built into the sensor which allows it to detect the direction of sound. Kinect also features a tilt motor and a 3-axis accelerometer.

Currently, a new generation of the sensor is available – Kinect for Xbox One. The main advantage over the Kinect for Xbox and Kinect for Windows is the increased resolution of the RGB and depth image (1920x1080 and 512x424, respectively). This allows the detection of not only the limbs, but also fingers. The field of view of the optical system has also been improved.

As with any other product, there are now alternatives on the market, most prominently Asus Xtion and Asus Xtion Pro. Until recently, it was possible to buy an OEM version, which was produced by the company behind depth sensing technology – PrimeSense (Fig. 1). Currently, the company has been bought by Apple.

2.1. Microsoft Kinect SDK

Microsoft Kinect SDK consists of a set of drivers, libraries and sample programs, developed by the manufacturer of the sensor. They allow developers to create their own applications in C++, C# and Visual Basic. This software allows the detection of up to six users and tracking silhouettes of two of them. These silhouettes are represented as skeletons consisting of 20 joints. The software allows to switch the sensor into "near mode", in which the dead zone is reduced to 40 cm, which allows the sensor to operate in confined spaces. However, this feature comes at a price – the range of the device is also decreased to 300 cm. This option is available only for Kinect for Windows version of the sensor. Another feature of the SDK, called "seated mode" allows tracking of silhouettes of sitting people. Figure 2 presents the visualization of a detected silhouette. Microsoft SDK provides full access to the settings of the on-board camera, such as white balance, gain, shutter speed, etc. Besides algorithms dedicated to tracking silhouettes, SDK includes algorithms enabling detection of simple gestures and facial features in 3D. It also gives access the tilt motor of the sensor as well as accelerometer measurements. It is also worth mentioning that the SDK provides a ready-to-use implementation of algorithms for detecting sound direction, and comes with an integrated speech recognition software – *Microsoft Speech Platform*. Microsoft Kinect SDK is limited solely to Windows operating systems.

2.2. OpenNI/NiTE

OpenNI is an open source software developed by a non-profit organization comprised of, among others, ASUS (manufacturer of Xtion Pro, alternative to Kinect sensors) and PrimeSense (manufacturer of the depth measurement technology used in Kinect sensors). OpenNI serves only as a platform for acquiring data from various RGB-D sensors and presenting them in a unified way. This data need to be further processed using a wide set of libraries (each tailored for specific application) cooperating with OpenNI platform. Of particular importance in the field of social robotics is the NiTE project. It allows the detection and tracking of silhouettes of an unlimited number of people. The detected silhouettes are represented as skeletons composed of 15 joints. NiTE can also be used to detect basic gestures, track the user's hand and detect certain body poses. Access to the settings of the sensor is very limited compared to Microsoft Kinect SDK. OpenNI software is portable and uses external drivers for the Kinect sensor (driver supplied by Microsoft Kinect SDK for operation under Windows and the Freenect driver for operation under Linux). Figure 2 presents the visualization of silhouette detection.

3. Experiments

In order to evaluate the most significant parameters of the Kinect sensor, new modules were added to the existing social robot control system [6]. Their implementation was based on the above mentioned projects (Microsoft SDK, OpenNI/NITE). The control system of FLASH is based on URBI [15] software, whose operation relies on dynamically-linked libraries called UObjects, each responsible for executing a specific set of competencies. Two new UObjects were created – UKinect [14] and UKinectOpenNI2 [13]. Experiments described in the following chapter have been carried out based on scripts written in urbiscript language provided by URBI. A complete documentation of the system can be found on FLASH's website [2]. Experiments were carried out using a Kinect for Windows sensor, connected to a laptop with an i5-2410M processor, 8 GB of 1333 MHz DDR3 RAM and an SSD hard drive, running under a Windows 7 64-bit operating system. Measurements carried out with a measuring tape were taken as a reference. Their accuracy is sufficient from the point of view of algorithms utilized in social robotics. The Reader should note that figures presented in this section might have axes with the same units but different scales.

3.1. Square Walk

During the first study a user was to cross a preset path, shaped as a rectangle. The dimensions of the path were 1.29x2.58 m and at it's closest point it was situated 1.27 m from the Kinect. The longer side of the rectangle was parallel to the Z axis of the sensor, and the shorter side was parallel to the X axis. An image of the test room along with the path followed by the user can be seen in Figure 3a and the expected result of the measurement is shown in Figure 3b. In order for each lap to be identical, both the pace and the places where the user was to put individual steps (points marked in Fig. 3a) were also predetermined. During the experiment, 4 persons with different body posture and

Fig. 1. Kinect for Windows, ASUS Xtion Pro, PrimeSense OEM sensor

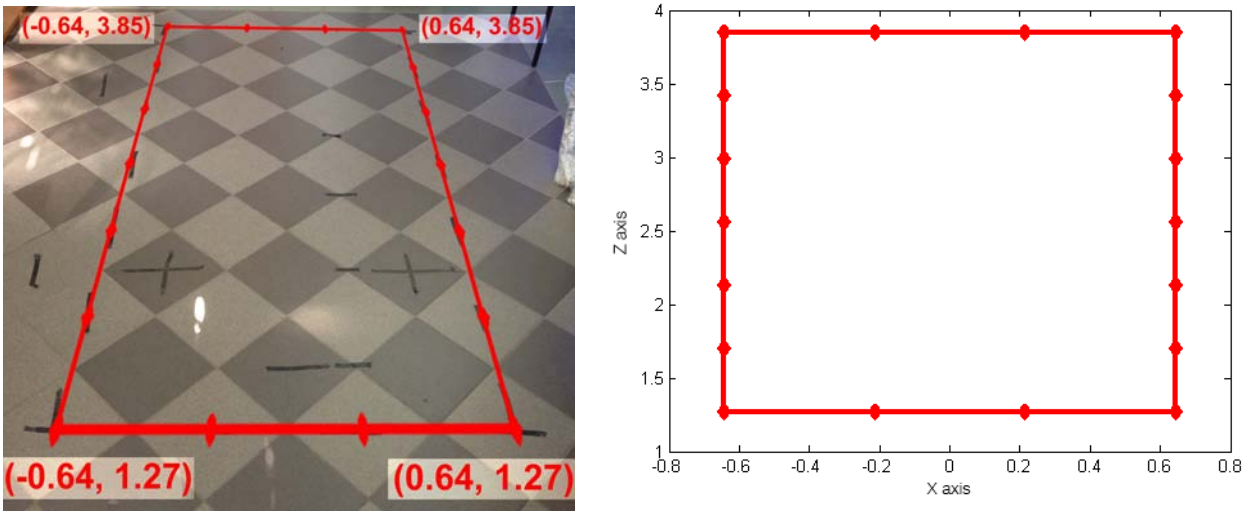
Fig. 2. Comparison of detected silhouettes (starting from the left: Kinect SDK, Kinect SDK "Seated mode", OpenNI/NiTE)

height completed five laps at a speed of 80 or 120 steps per minute.

Figures 4a and 4b represent the projections of the center of mass of the detected user onto the XZ plane for each of the four subjects during a single lap. Analysis of the results suggests that Kinect SDK is capable of greater accuracy in determining the position of a user than OpenNI. The dispersion of the results around the mean is also smaller in the case of Kinect SDK. For both libraries there were no significant differences between the test users, which suggests that the tests carried out are sufficiently reproducible and reliable. An important objective of this test was to identify the joints (possibly different for both libraries) that can be tracked to accurately determine the position of a person, and to investigate how the speed of the user's movement will affect the measurement accuracy.

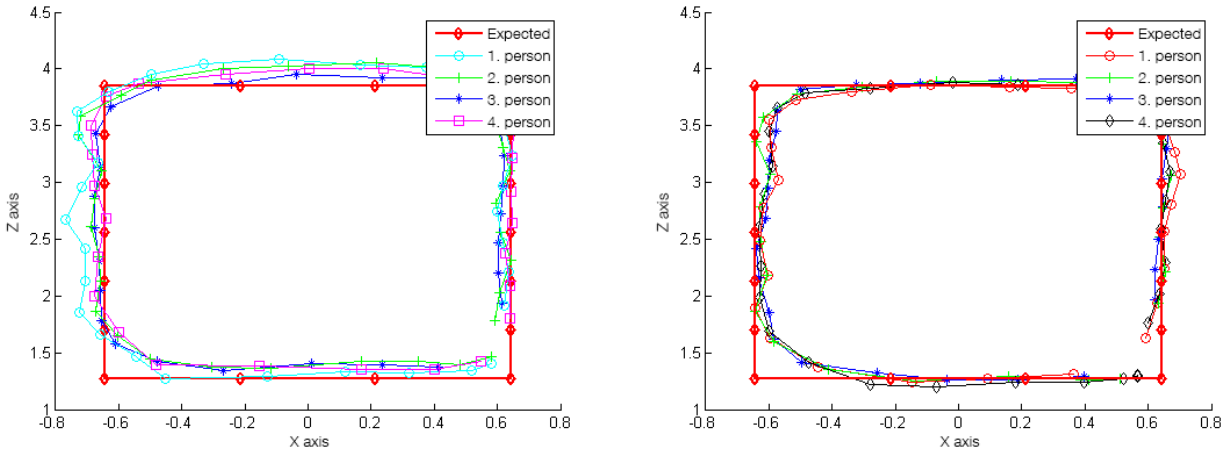
Figures 5a and 5b represent the path followed by the user's head, torso, left and right arm for both modules. An important observation is the discontinuity in head tracking noticeable in Figure 5a. During the experiment, in 330 out of 1472 samples for the slow walking speed, and 44 out of 1019 samples for fast walking speed, OpenNI was not capable of establishing the head's position. This problem did not, however, extend to other joints. Additionally, this phenomenon was not observed when using Kinect SDK.

Joints which according to the authors can accurately determine the position of a person are summarized in Table 1. It presents the average distance from the reference path and the measured height for the selected joints. When using OpenNI, these were neck and torso and when using Kinect SDK they include shoulder center and back. For both software platforms head and center of mass was also selected as well as the point located midway between the right and left arm (mean taken from arm positions). All of the mentioned joints are approximately situated in one line, perpendicular to XZ plane. Their projection onto this plane, should coincide with the walking path. Gathered data revealed the joints which reached the best accuracy of the measured position in the XZ plane and the smallest dispersion of results in the Y axis. For OpenNI, the best accuracy in the XZ plane was obtained for the center of mass of the detected character and the position of the torso. The accuracy of position in the Y-axis was comparable for all considered joints, with the exception of the user's center of mass, which showed a slightly higher levels of dispersion. In the case of Kinect SDK, the best accuracy in the XZ plane can be ensured by the center of mass of the detected user. The smallest standard deviation of the measurements in the Y axis was observed for the back and center of the segment connecting both arms. Based on the data, it can be safely assumed that walking speed has a negligible influence on the accuracy of measurements.



(a) Test room with highlighted path *(b) Expected results*

Fig. 3. Test path



(a) OpenNI, 2D view *(b) Kinect SDK, 2D view*

Fig. 4. Path of the center of mass during a single lap, 80 steps per minute (every tenth sample)

3.2. Orientation

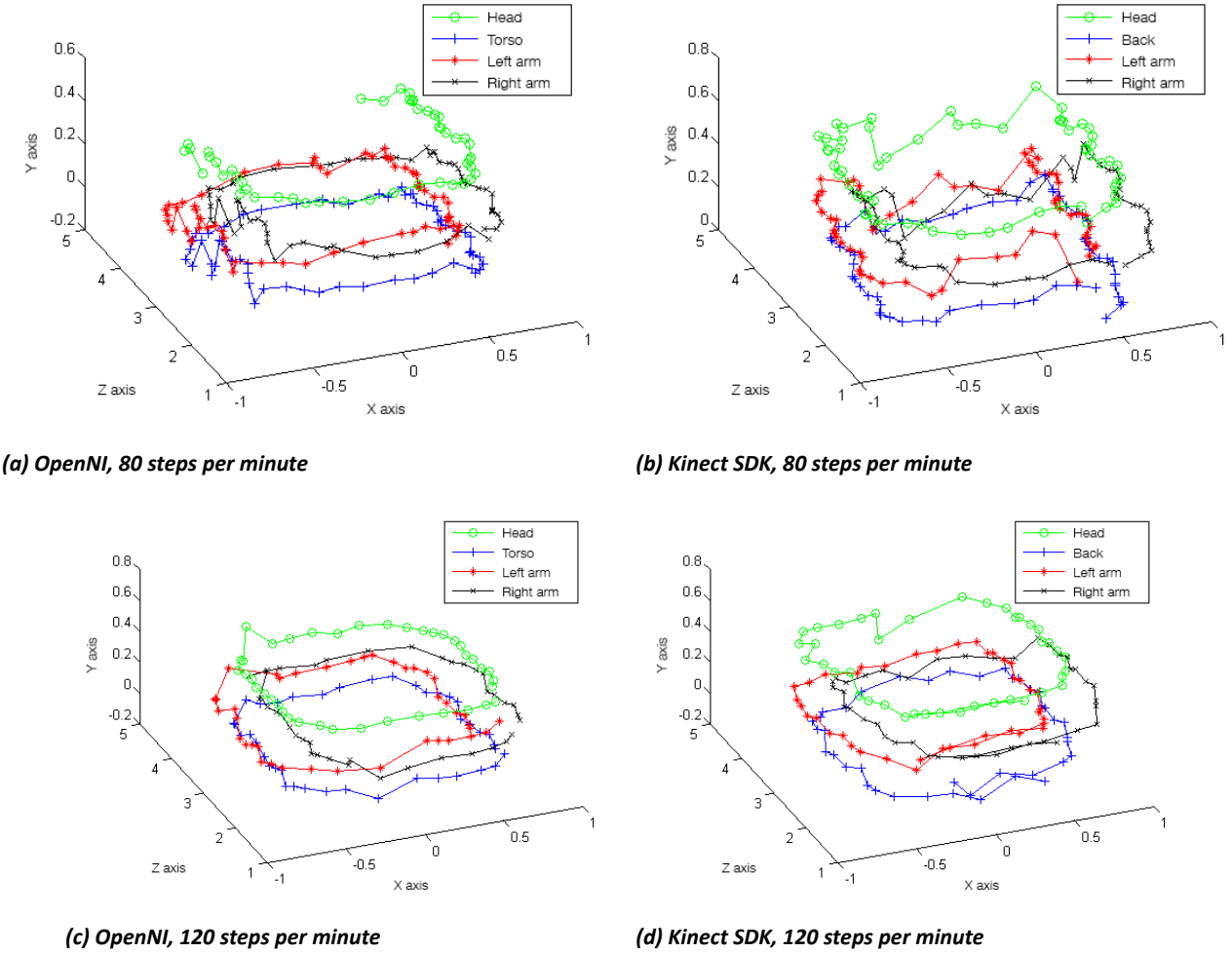
The information about the user's orientation is crucial from the point of view of human interaction with a robot. It allows to determine when a person is facing in the direction of a robot. The exact orientation is required in order to properly decode many social cues, especially when the robot is functioning in multi-agent environment. For comparison purposes, test were conducted where the user performed 30 revolutions in motion, following two paths shown in Figure 6.

The results of tests using OpenNI library are shown in Figures 7a and 7c whereas results obtained with Kinect SDK can be seen in Figure 7b and 7d. It is apparent that OpenNI correctly determines the orientation of the body, while Kinect SDK always assumes that the user is facing the device. A thorough analysis of the orientation in the vicinity of the points where it is expected to be $-\frac{\pi}{2}$ or $\frac{\pi}{2}$ (left and right vertex of the triangle in Figure 6) showed that OpenNI always properly determined where the user was facing, and the average error was only 0.0215 π radians for the first path and 0.0297 π radians for the second path. Kinect SDK never properly detected the orientation when the user's back was facing toward the device. This problem possibly originated from Microsoft focusing on Kinect as game controller. Such usage of the device would never require the user to be facing away from the sensor.

3.3. Detection Time

The next set of experiments was devised to analyze the time in which both libraries detect the presence of a person, but without any information about user's silhouette (henceforth called detection time) and the time required to begin tracking the detected user's skeleton (ie. provide information about positions of his/her joints, henceforth called tracking start time). There were 5 variants of the experiment:

1) User standing still 2.4 m from the sensor, whole



(a) OpenNI, 80 steps per minute

(b) Kinect SDK, 80 steps per minute

(c) OpenNI, 120 steps per minute

(d) Kinect SDK, 120 steps per minute

Fig. 5. Joint paths during first of the five laps (every fifth sample)

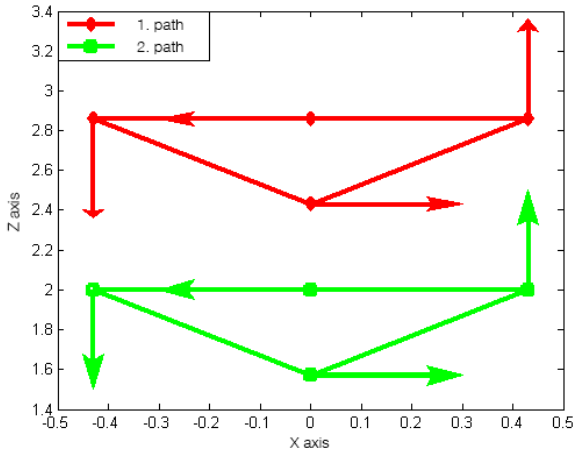


Fig. 6. Test paths – points reflect user's steps, arrows – body orientation

body in the device's field of view.

2) User stepping left and right, steps 43 cm apart, 100 steps per minute, whole body in the device's field of view.

3) User stepping left and right, steps 43 cm apart, 100 steps per minute, body from the neck down in the device's field of view.

4) User stepping left and right, steps 43 cm apart, 100 steps per minute, body from the knees up in the device's field of view.

5) User stepping left and right, steps 43 cm apart, 100 steps per minute, body from the waist up in the device's field of view.

For each of the variants mentioned above, 30 measurements were performed for both libraries. A mea-

Tab. 1. Accuracy of determining joint position based on walking speed (in meters)
Dev. – standard deviation, Pp. v. – peak-to-peak value, spm. – steps per minute

	Distance (projection onto XZ)			Height (Y value)		
	Mean	Dev.	Pp. v.	Mean	Dev.	Pp. v.
OpenNI						
Center of mass, 80 spm.	0.0118	0.0529	0.3250	–0.0549	0.0671	0.3126
Head, 80 spm.	–0.0110	0.0959	0.5342	0.4271	0.0424	0.3369
Neck, 80 spm.	–0.0044	0.0728	0.3944	0.1671	0.0548	0.3325
Torso, 80 spm.	0.0052	0.0608	0.3872	–0.0044	0.0566	0.3496
Mean from both arms, 80 spm.	–0.0044	0.0728	0.3944	0.1671	0.0548	0.3325
Center of mass, 120 spm.	0.0083	0.0608	0.3570	–0.0568	0.0794	0.3926
Head, 120 spm.	–0.0287	0.1131	0.5689	0.4179	0.0490	0.4892
Neck, 120 spm.	–0.0182	0.0877	0.4613	0.1702	0.0557	0.4521
Torso, 120 spm.	–0.0120	0.0755	0.4005	0.0073	0.0557	0.4287
Mean form both arms, 120 spm.	–0.0182	0.0877	0.4613	0.1702	0.0557	0.4521
Kinect SDK						
Centre of mass, 80 spm.	–0.0095	0.0990	0.2800	–0.0633	0.0566	0.6814
Head, 80 spm.	–0.0514	0.0775	0.4629	0.5397	0.0592	0.3868
Shoulder center, 80 spm.	–0.0519	0.0755	0.3788	0.3894	0.0510	0.3904
Back, 80 spm.	–0.0540	0.0707	0.3801	0.0615	0.0510	0.2841
Mean from both arms, 80 spm.	–0.0340	0.0600	0.4064	0.2865	0.0548	0.4238
Center of mass, 120 spm.	–0.0087	0.1082	0.6446	–0.0580	0.0480	0.6032
Head, 120 spm.	–0.0598	0.0911	0.5679	0.5383	0.0447	0.4105
Shoulder center, 120 spm.	–0.0602	0.0877	0.5133	0.3885	0.0447	0.4020
Back, 120 spm.	–0.0626	0.0877	0.5188	0.0621	0.0374	0.2548
Mean from both arms, 120 spm.	–0.0402	0.0883	0.5210	0.2835	0.0316	0.3661

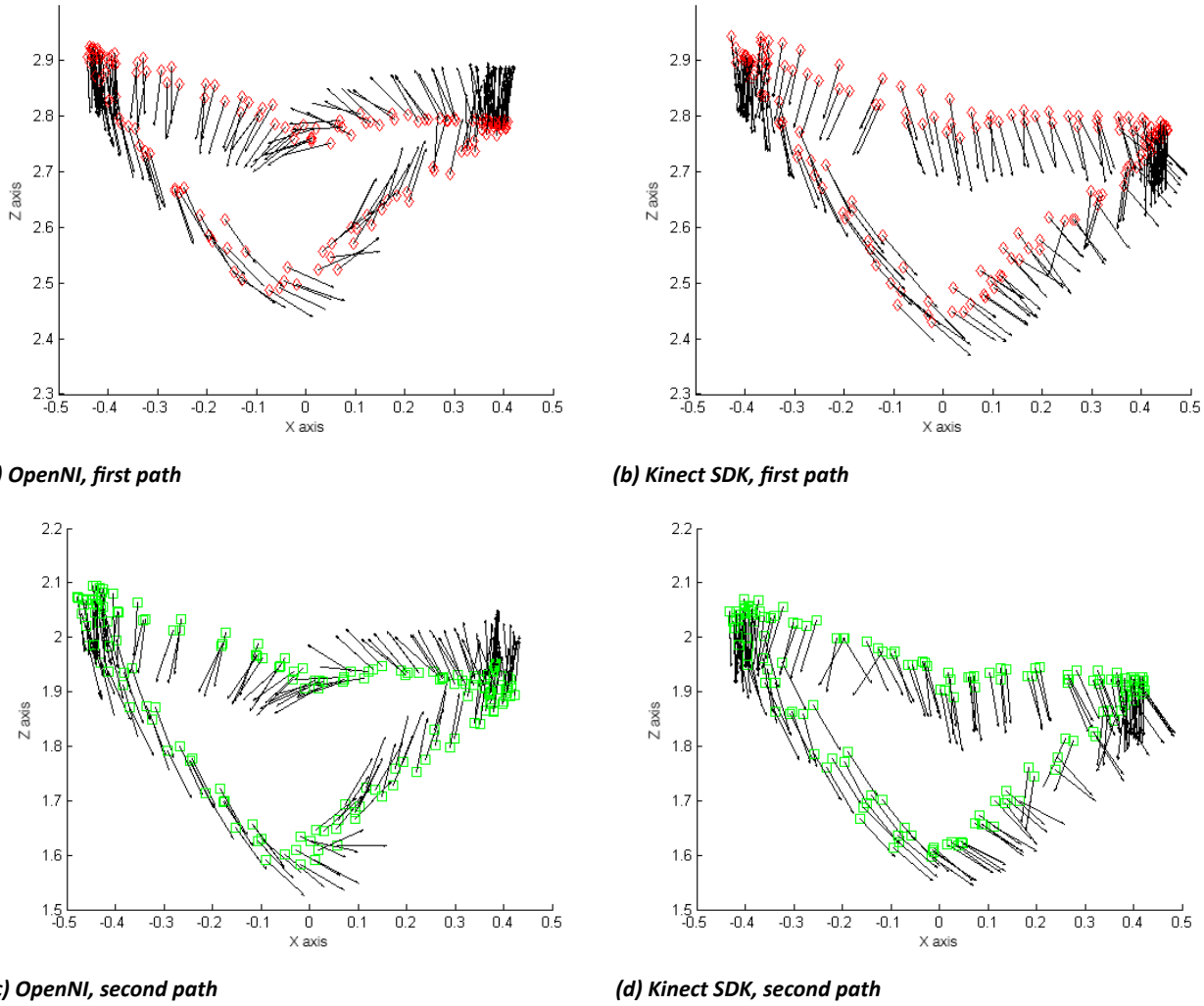
surement began with the launch of a new instance of the application written in urbiscript. This way the history of user detection did not influence the results. The moment when the user entered the Kinect's field of view was constant (since the user was constantly visible to the sensor and software tasked with acquiring data from the Kinect was being restarted) which also helped eliminate any possible errors. The results are shown in Table 2.

Analysis of the results shown in Table 2 reveals a few key differences between OpenNI and Kinect SDK. OpenNI can neither detect nor track a person's skeleton when he/she is standing still. To the authors' knowledge there can be two reasons that explain this behavior. Firstly, OpenNI does not use the rough information about the detected silhouettes provided by the Kinect sensor (last 3 bits of depth data). Secondly, OpenNI uses a sequence of frames in order to detect and track a person as opposed to the detection/tracking being based on a static image (as is the case with Kinect SDK). In addition to the observed inability to detect a stationary person, this hypothesis also finds confirmation in observations obtained during the usage of the software. Experience shows that with the fall in the number of processed frames per second (eg. due to excessive load on the computer) the efficiency of OpenNI drops drastically.

Another important difference is the effect of obstruction of the sensor's field of view on the ability to detect and track users. OpenNI library copes better with people seen from the neck down – it can detect their presence, which is impossible for Kinect SDK. However, the effectiveness of Kinect SDK is far higher in situations where only the upper parts of a user's body are visible. Such situations are more common, and information about the upper parts of the body are much more important in the context of human robot interaction. SDK was able to quickly detect (after processing an average of 20.4 – 24.8 frames) the person seen from both the knees up and the waist up. This result was obtained without switching the device into "seated mode". Utilizing OpenNI yields worse results - for the silhouette seen from the knees upwards, detection time was comparable to when the user was fully visible, but tracking was achieved on average 24.12s after detection and had a very large standard deviation. When the user was only visible from the waist up, OpenNI did not manage to detect a silhouette.

During the tests with a moving user, detection time and tracking start times are small for both libraries. Studies have shown that OpenNI is slightly faster - an average of 15 frames to detect human presence and an average of 27 frames to start tracking. Kinect SDK required an average of 20 frames for detection and 22 frames for tracking. It should be noted that the times of detection and tracking in OpenNI have a much higher standard deviation those observed for Kinect SDK. In the case of Microsoft software tracking always began 2 frames after detection. OpenNI exhibited no such regularities.

The last experiment dealing with detection time and tracking start times was meant to investigate the



(a) OpenNI, first path

(b) Kinect SDK, first path

(c) OpenNI, second path

(d) Kinect SDK, second path

Fig. 7. Orientation of user's body during tests (every third sample)

Tab. 2. Results of detection and tracking start times (presented as the number of frames and time in seconds). A dash in the corresponding field means that the software failed to detect/track the user. Dev. – standard deviation

Variant	Detection - frame		Detection – time		Tracking – frame		Tracking – time	
	Mean	Dev.	Mean	Dev.	Mean	Dev.	Mean	Dev.
OpenNI								
1	–	–	–	–	–	–	–	–
2	15.33	2.42	0.73	0.46	26.63	2.52	1.30	0.47
3	13.33	2.31	0.60	0.42	–	–	–	–
4	13.07	2.37	0.63	0.47	556.63	24.00	24.12	19.20
5	–	–	–	–	–	–	–	–
Kinect SDK								
1	20.97	1.35	1.03	0.55	22.97	1.35	1.11	0.54
2	19.93	0.92	0.92	0.20	21.93	0.92	0.99	0.20
3	–	–	–	–	–	–	–	–
4	20.40	1.21	0.99	0.51	22.40	1.21	1.06	0.51
5	24.80	2.49	1.13	0.62	26.80	2.49	1.22	0.62

effect of a larger number of people in the field of view of the sensor. As the Kinect SDK has the ability to track only two skeletons, the test was limited to two people. The test results are summarized in Table 3. Shorter average detection time was achieved using OpenNI and shorter average tracking start time using Kinect SDK.

3.4. Minimal Distance from a Wall

During numerous HRI studies the authors of this article observed negative effects when the user was

Tab. 3. Detection time and tracking start times (presented as the number of frames) when two people are present in the sensor's field of view
Dev. – standard deviation

	Detection frame - 1st person		Detection frame - 2nd person		Tracking frame - 1st person		Detection frame - 2nd person	
	Mean	Dev.	Mean	Dev.	Mean	Dev.	Mean	Dev.
OpenNI	16.10	2.72	20.83	2.85	27.87	2.82	34.73	3.68
Kinect SDK	23.27	2.80	29.93	3.89	25.27	2.80	31.93	3.89

close to a wall (or other flat surface). The proximity of such surfaces hinders the Kinect's to properly detect and track a user. Due to the features of OpenNI (inability to detect static users coupled with the dependence on history of detections), this test was performed only for the Kinect SDK library. The distance from a wall at which Kinect SDK managed to detect a human silhouette is (on average) 0.28m from the wall to the user's center of mass.

3.5. Users Passing Each Other

In order to maintain eye contact or keep up a conversation with the user, a social robot must be able to repetitively and robustly choose the speaker from among several people. It is extremely important that, after establishing first contact, the robot was able to correctly track a specific user. Without this ability undisturbed human-robot interaction can occur when there is only one user in the robot's field of view. We proposed a test, which aimed to examine whether both libraries properly ascribe identification numbers to users when they obscure each other. There were two variants of the experiment.

1) Variant with two people were walking at a pace of 80 steps per minute, following a path parallel to the X axis of the sensor. One of them was located 1.71 m from the device, the other one 2.35 m away. Their paths crossed in the middle (directly in from of the sensor).
2) Variant with three people, including one who is standing directly in front of the sensor, 2.35 m away from it and the other two walking at a pace of 80 steps per minute, following a square path symmetric with respect to the sensor's Z axis. Each side of the path was 1.29 m long and its proximal side was 1.71 m away from the Kinect. The walking people obstructed each other at the point where the first person was standing.

The results show a significant advantage of OpenNI library over Kinect SDK. A typical case of user detection for both packages is shown in Figure 8. Every user was given a specific identification number. Obviously, none of the packages managed to track an obscured person. However, in the case of OpenNI, once the user was again visible to the sensor (even when the user's side was turned towards the Kinect) tracking resumed almost immediately. More importantly, the person whose tracking was interrupted regained the exact same identification number that they had before being obscured. Kinect SDK always assigned a new ID number to the obscured user and detection was resumed only when the person turned towards the sensor.

3.6. Accuracy of Hand Position Measurement

During HRI experiments it is often necessary to accurately determine the position of the selected parts of the body (usually the hand or the head). This is especially important e.g. while recognizing the color of the object the user is holding, determining the direction indicated by a human or interpreting various other static gestures. To evaluate the accuracy of the Kinect, an experiment was devised in which the position of the right hand was measured. During the test, the sensor was placed 0.81 m above ground (distance between the floor and the center of Kinect's optical system). Right arm was still during the measurement, extended 1.35 m above ground, shifted by 0.86 m in the X axis of the sensor. The experiment was performed in the following variants:

- different distance from the sensor (in Z axis): near – 1.71m, far – 2.57 m,
- user standing still or moving left-right (his hand being still) with 0.43 m long steps at a pace of 100 steps per minute.

During the experiment, the effects of using filters smoothing the position of the joints in time was also examined. For OpenNI the level of filtering is defined as a number between 0 and 1, where 0 is no filtration and 1 is the total lack of joint movement (the default setting is 0.3). Kinect SDK has four selectable predefined filter configurations (where 0 – no filtration, 4 – the strongest smoothing).

The experimental results are presented in Table 4. It was observed, that the filter settings did not have a major impact on the resulting measurements. Further analysis of the results shows that in most cases OpenNI provides a more accurate measurement (2-3 times the accuracy of Kinect SDK). The standard deviation, in the case of the user being near the sensor, is about two times smaller for Kinect SDK. When we analyze the results from the variant where the user is farther away, results are comparable. Results delivered by OpenNI are significantly influenced by the user's motion but only when the user is at a distance from the sensor. In the case of Kinect SDK the movement of the user does not affect the accuracy of measurements (regardless of the distance).

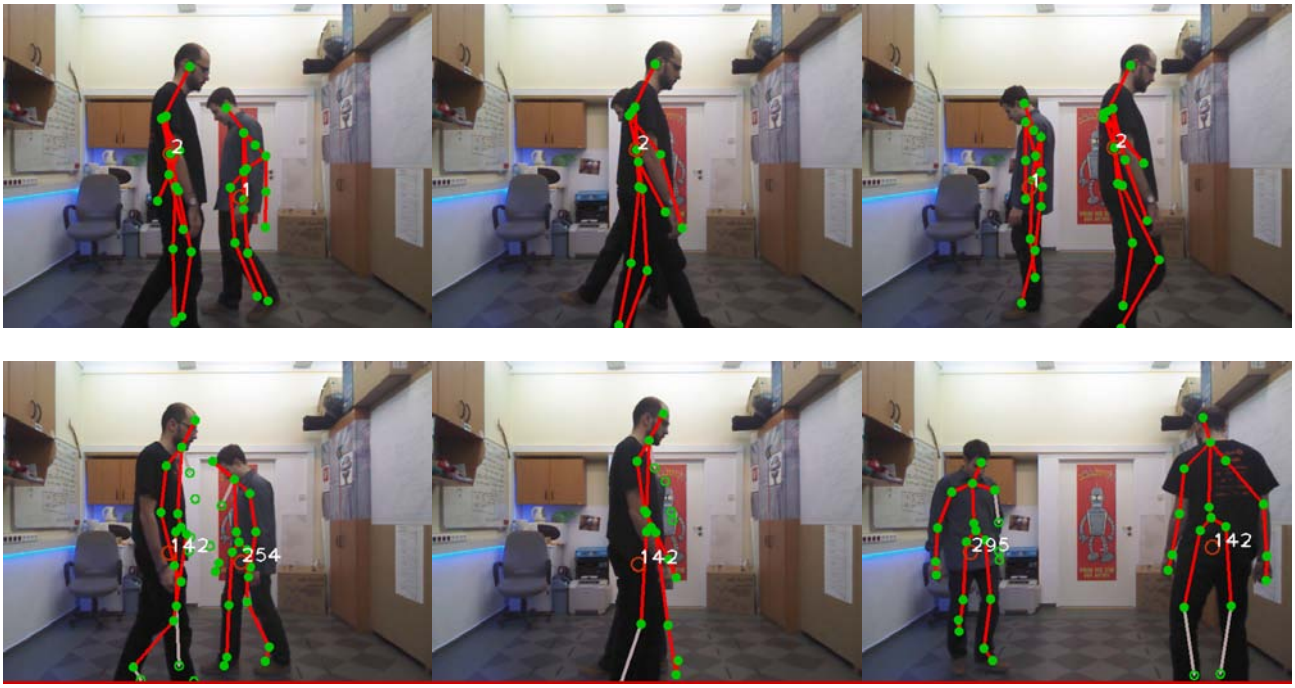


Fig. 8. Detection when users are passing each other (top: OpenNI, bottom: Kinect SDK)

Tab. 4. Results of hand position measurements (in meters)
Dev. – standard deviation

Filter setting	No movement				Movement			
	Near		Far		Near		Far	
	Error	Dev.	Error	Dev.	Error	Dev.	Error	Dev.
Open NI								
Filter = 0.0	0.0210	0.0426	0.2718	0.1224	0.0304	0.0558	0.0750	0.0191
Filter = 0.15	0.0305	0.0777	0.0420	0.0103	0.0299	0.0371	0.0649	0.0194
Filter = 0.3	0.0232	0.0410	0.0434	0.0105	0.0180	0.0329	0.0768	0.0222
Filter = 0.45	0.0220	0.0163	0.0369	0.0093	0.0391	0.0680	0.0879	0.0249
Filter = 0.6	0.0175	0.0269	0.0403	0.0073	0.0230	0.0391	0.0701	0.0232
Kinect SDK								
Filter = 0	0.0638	0.0171	0.1253	0.0106	0.0799	0.0223	0.1065	0.0292
Filter = 1	0.0606	0.0039	0.1183	0.0107	0.0822	0.0291	0.1096	0.0273
Filter = 2	0.0787	0.0159	0.1278	0.0096	0.0879	0.0265	0.1099	0.0260
Filter = 3	0.0880	0.0044	0.1293	0.0082	0.0664	0.0246	0.1024	0.0171

4. Conclusion

The experiments described in this article aimed to verify the applicability of the Kinect sensor in social robotics. Particular emphasis was placed on the key features and differences between Microsoft SDK and OpenNI/NITE libraries used to develop applications and algorithms oriented towards human perception.

Both libraries provide algorithms for detection and tracking of silhouettes and individual joints. Such algorithms are essential from the point of view of human-robot interaction. Kinect SDK is preferable with regard to the number of tracked joints (20 when compared to 15 joints available in the OpenNI). The advantage of OpenNI is its ability to track a virtually unlimited number of users (in Kinect SDK only two users can be tracked at the same time). A key asset of the software provided by Microsoft is a very large number of additional features - "near mode" and "Seated mode", 3D face detection, and support for the built-in microphone array. OpenNI is an open platform and can be easily expanded thanks to numerous libraries (currently there are about 50 on the OpenNI website). Each new library provides many application specific functions such as precise hand and face tracking, gesture recognition, scanning, and reconstruction of 3D models. However, to the authors' knowledge, there are no libraries compatible with OpenNI which allow access to the built-in microphone arrays. RGB-D image processing capabilities of both libraries are comparable.

To summarize the comparison of OpenNI/NiTE and Kinect SDK and experimental verification of their parameters, we can conclude that both provide a rich

new source of data about the environment, extremely useful when endowing a robot with user-oriented, social capabilities. Selecting the proper library for use with the Kinect sensor depends on the individual needs of the specific application. OpenNI allows tracking of an unlimited number of people, determining the correct orientation of the user, or precisely measuring the position of joints. Kinect SDK provides many additional features, fast detection and tracking even in cases of limited visibility and, above all, the ability to detect the users even if they are not moving.

ACKNOWLEDGEMENTS

The research described in this article has been funded by the Polish National Science Centre under grant 2012/05/N/ST7/01098.

AUTHORS

Michał Dziergwa – Chair of Cybernetics and Robotics, Wrocław University of Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, e-mail: michal.dziergwa@pwr.edu.pl.

Paweł Kaczmarek – Chair of Cybernetics and Robotics, Wrocław University of Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, e-mail: pawel.m.kaczmarek@pwr.edu.pl.

Jan Kędzierski* – Chair of Cybernetics and Robotics, Wrocław University of Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, e-mail: jan.kedzierski@pwr.edu.pl, www: http://lirec.iiar.pwr.edu.pl/~jkedzier.

*Corresponding author

REFERENCES

[1] R. A. Clark and et al., "Validity of the Microsoft Kinect for assessment of postural control", *Gait and Posture*, vol. 36, no. 3, 2012, 372–377, DOI:10.1016/j.gaitpost.2012.03.033.

[2] FLASH. "Homepage". http://flash.iiar.pwr.edu.pl, 2014.

[3] H. Gonzalez-Jorge, B. Riveiro, E. Vazquez-Fernandez, J. Martínez-Sánchez, and P. Arias, "Metrological evaluation of Microsoft Kinect and Asus Xtion sensors", *Measurement*, vol. 46, 2013, 1800–1806, DOI:10.1016/j.measurement.2013.01.011.

[4] P. Henry and et al., "RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments", *International Journal of Robotics Research*, vol. 31, no. 5, 2012, 647–663, DOI:10.1177/0278364911434148.

[5] T. Huynh, R. Min, and J. Dugelay, "An Efficient LBP-Based Descriptor for Facial Depth Images Applied to Gender Recognition Using RGB-D Face Data", *Lecture Notes in Computer Science*, vol. 7728, 2013, 133–145, DOI:10.1007/978-3-642-37410-4_12.

[6] J. Kędzierski. *System sterowania robota społecznego*. PhD thesis, Politechnika Wrocławska, Wrocław, 2014.

[7] K. Khoshelham, "Accuracy Analysis of Kinect Depth Data", *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XXXVIII-5/W12, 2011, 133–138, DOI:10.5194/isprsarchives-XXXVIII-5-W12-133-2011.

[8] K. Khoshelham and S. O. Elberink, "Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications", *Sensors*, vol. 12, 2012, 1437–1454, DOI:10.3390/s120201437.

[9] Microsoft. "Kinect SDK documentation". http://msdn.microsoft.com/en-us/library/hh855347.aspx, 2014.

[10] I. Oikonomidis, N. Kyriazis, and A. Argyros, "Efficient model-based 3D tracking of hand articulations using Kinect". In: *Proceedings of the British Machine Vision Conference*, 2011, 101.1–101.11, DOI:10.5244/C.25.101.

[11] OpenNI. "Project website". http://www.openni.org/, 2014.

[12] A. Ramey, V. Gonzalez-Pacheco, and M. A. Salichs, "Integration of a low-cost RGB-D sensor in a social robot for gesture recognition". In: *2011 6th ACM/IEEE International Conference on Human-Robot Interaction*, 2011, 229–230, DOI:10.1145/1957656.1957745.

[13] UKinect. "Module documentation". http://lirec.ict.pwr.wroc.pl/flash/?q=node/110, 2014.

[14] UKinectOpenNI2. "Module documentation". http://lirec.ict.pwr.wroc.pl/flash/?q=node/121, 2014.

[15] Urbi. "Project website". http://www.urbiforge.org, 2014.

[16] D. Vasquez and et al., "Human Aware Navigation for Assistive Robotics", *Experimental Robotics*, vol. 88, 2013, 449–462, DOI:10.1007/978-3-319-00065-7_31.