Recovery of Forearm Occluded Trajectory in Kinect Using A Wrist-mounted Inertial Measurement Unit

Prayook Jatesiktat[1] and Wei Tech Ang[1]

***Abstract*— Kinect sensor is a successful device that lets 3D human motion capture be used in a general residential setting. This work aims to fulfill some missing capabilities in Kinect, which are forearm orientation estimation and forearm tracking in occlusion. By using a wrist-mounted Inertial Measurement Unit and Kinect's built-in skeleton tracking, we have developed a fusion procedure that improves the upper limb motion tracking without adding too many obtrusive devices to the user.**

I. Introduction

Kinect for Xbox One is an off-the-shelf time-of-flight depth camera sensor that comes with a built-in 3D skeleton tracking algorithm in Kinect Software Development Kit (SDK). The device is widely adopted in the home setting for gaming due to its ease of use in an affordable price. However, when the need for human motion capture at home starts to go beyond a simple gaming control (e.g. rehabilitation assessment), its capturing quality can be unsatisfactory for some types of movement [1], [2], [3], [4], [5], [6]. To improve this type of optical motion capture system, Inertial Measurement Unit (IMU) is chosen in a number of studies.

A few studies [7], [8] propose similar methods by attaching IMUs to every segment they need to track and use Madgwick's method [9] to fuse accelerometer, gyroscope, and/or magnetometer to get 3-DOF orientations in the world reference frame. They use absolute positions of torso and shoulder from Kinect. Segment orientations from Kinect are simply replaced by orientations from IMUs. The results show a higher motion capture rate [7] and better joint position accuracy [8]. Similarly, Raghuraman et al. [10] design an immersive tennis game using 2 Kinects in front and behind the player and one IMU on the user's wrist. However, the work does not report how the orientation from the wrist-mounted IMU is used in the control.

Helten et al. propose a hybrid method for real-time full-body tracking which targets at solving the occlusion problem and non-frontal pose of Kinect [11]. Instead of Kinect's SDK, they based their method on Baak et al.'s work [12] which synthesizes a surface mesh model that can be deformed according to an embedded kinematic skeleton with 42 degrees of freedom. A visual database lookup is indexed by first 5 geodesic extrema from Kinect's point cloud. An inertial database lookup is indexed by 3D vectors representing pointing directions of 2 forearms, 2 lower legs, and a user's head. Those lookups will be optimized locally and the better result will be selected. Note that the used information from 5 IMUs is not a complete 3-DOF orientation. Therefore, the segment roll information such as pronation/supination is missing.

Tian et al. [13] propose an unscented-Kalman-filter-based fusion method between Kinect's positions (i.e. upper arm and forearm) and two 9-axis IMUs on the user's upper arm and forearm. The state estimation algorithm covers position, velocity, and orientation quaternion. One of their targets is to use IMU data to fill in some Kinect outage, i.e. a short period that Kinect gives jumpy tracking trajectory. Still, their result could drift beyond 20 cm within 1 second outage.

In this work, two problems of Kinect are addressed: (1) its unstable forearm orientation estimation, and (2) its forearm occlusion problem. The former usually happens with the forearm pronation/supination due to the cylindrical shape of the forearm. Especially when the hand is closed, Kinect has no visual clue for the roll information. The later occurs either when the forearm is behind other body parts or when the forearm is pointing directly to the camera and gives a noisy depth image of forearm surface.

To solve these problems while maintaining the system's ease of use, only one IMU is attached to the user's wrist. An IMU includes a accelerometer, a gyroscope, and a magnetometer. A fusion of 3 sensors can provide an orientation of the IMU without visual information. On the other hand, the 3D position of the sensor cannot be retrieved without an aiding from global positioning sensor. For a classic strapdown inertial navigation algorithm, the noise in acceleration will be accumulated in the double integral process and become a large positional drift within a short time period [14]. Therefore, some visual information from Kinect is used to correct the drift in our trajectory recovery process.

The rest of the paper is organized as follows. Section II explains how the system is implemented. Section III describes an evaluation of our system against a marker-based motion capture system and the result is shown in section IV.

II. System Implementation

A. Hardware&Software Configuration

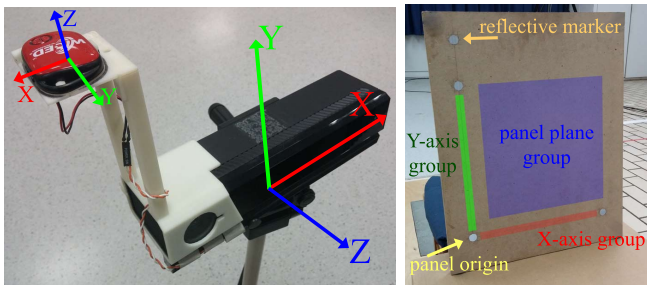
Kinect for Xbox One is chosen for our experiment. It includes a depth camera, an infrared camera, an RGB camera, and a microphone array. Kinect SDK 2.0 is used to retrieve both raw and processed data (i.e. skeleton) from the device.

For IMU, WICED Sense development kit is selected. It contains a 3-axis accelerometer, a 3-axis gyroscope, a 3-axis magnetometer, a buzzer, and a Bluetooth Low Energy module. Data are sampled at 80 Hz and transmitted to a Windows 10 machine via a Bluetooth 4.1 dongle.

[1]Prayook Jatesiktat and Wei Tech Ang are with the School of Mechanical and Aerospace Engineering, Nanyang Technological University, 50 Nanyang Ave, 639798, Singapore prayook001@e.ntu.edu.sg, wtang@ntu.edu.sg

978-1-5090-2809-2/17/$31.00 ©2017 IEEE

Two IMUs are used in this system. The first IMU (arm-IMU) is tied to the user's wrist. A round of black strap around the forearm at the IMU position is required for being detected by RGB camera. The second IMU (base-IMU) is fixed on top of the Kinect using a 3D-printed case (Fig.1a). Importantly, the IMU must be far enough from the Kinect to avoid a strong magnetic disturbance generated by Kinect. The buzzer cord is extended from IMU to place in front of the Kinect's microphone for a synchronization purpose.



(a) Kinect and base-IMU. (b) Calibration panel.

Fig. 1: (a) Kinect and base-IMU have a fixed relative orientation. (b) 4 reflective markers and 3 regions used to extract 6-DOF parameter of the panel.

B. Reference Frames

There are 3 main reference frames in this configuration

1) World Reference Frame $\{W\}$: The X, Y, and Z-axis are mapped to the local magnetic north, local magnetic west, and upward direction. The quaternion-based Kalman filter in section II-D gives each IMU's orientation in this frame.

2) Kinect Reference Frame $\{K\}$: This frame is defined by Kinect's depth camera and infrared camera which shares the same extrinsic parameters, i.e. position and orientation. Its X-, Y-, and Z-axis are mapped to the left, top, and front side of the device respectively. All the visual sensing information from Kinect SDK is in Kinect reference frame except RGB image which needs SDK's *CoordinateMapper* library to map their position to Kinect reference frame.

3) Eagle Reference Frame $\{E\}$: Eagle is an 8-camera marker-based motion capture system used as a gold standard. The frame is defined by the Eagle calibration process.

4) Other Reference Frames: For other reference frame notations, $\{B\}$ is the base-IMU frame, $\{A\}$ is the arm-IMU frame, $\{F\}$ is the forearm frame, and $\{P\}$ is the calibration panel frame.

C. Quaternion Notation

Quaternion is chosen as the main representation of orientation between reference frames. The notation ${}^{I}_{J}q$ means a unit quaternion that can convert a vector (V) in reference frame $\{I\}$ to reference frame $\{J\}$ by

$$ {}_{J}V = {}^{I}_{J}q \times {}_{I}V \times {}^{I}_{J}q^{-1} \quad or \quad {}_{J}V = rot({}^{I}_{J}q, {}_{I}V) $$

The operation $\times$ adapts according to its operands. It could be a quaternion multiplication, a matrix multiplication, or a cross product.

D. Quaternion-based Kalman Filter

In order to find ${}^{B}_{W}q$ and ${}^{A}_{W}q$, data from each individual 9-axis IMU are fused using a quaternion-based Kalman orientation filter proposed by Comotti et al. [15]. However, their vector observation step is modified by replacing the Gauss-Newton optimization method with an analytical solution. Our vector observation step assumes that the observed acceleration points in the same direction as Z-axis of the world reference frame, i.e. gravity direction. This assumption leads to all three world coordinate axes in the IMU reference frame by using only measured acceleration direction $(\hat{g})$ and measured magnetic direction $(\hat{m})$ in unit vector.

$$ V'_X = (\hat{g} \times \hat{m}) \times \hat{g}; \qquad V'_Y = \hat{g} \times \hat{m}; \qquad V'_Z = \hat{g} \quad (1) $$

These three orthogonal axes will be normalized and fed into van Waveren's method [16] to convert them to an observed quaternion $({}^{IMU}_{W}q)$. However, $-({}^{IMU}_{W}q)$ is always another solution. In order to get a smooth filter result, the solution that is closer to the current predicted orientation will be chosen as ${}^{IMU}_{W}q_{observed}$. That is

$$ {}^{IMU}_{W}q_{predicted} \cdot {}^{IMU}_{W}q_{observed} > 0 \quad (2) $$

E. IMU Calibration

The result from the orientation filter heavily depends on the IMU signal quality. This section describes a few significant calibrations needed for the described filter.

1) Magnetometer Calibration: Magnetic field intensity measured by magnetometer always has 2 kinds of interference: hard-iron and soft-iron interference. Hard-iron interference occurs from nearby magnetic sources such as permanent magnet and speaker. It causes a constant offset from the earth's magnetic field reading. Soft-iron interference occurs from nearby ferromagnetic materials, such as metal bars, screws, battery contacts. It causes a symmetrical distortion in the measurement.

Without such interference, thousands of magnetometer samples from a rotating IMU should form a point cloud of a perfect sphere. In reality, interference from the device itself will distort that sphere into a shifted-tilted ellipse. This distortion is static as long as those interference sources have no change in relative position and rotation to the magnetometer. It could be fixed by the following calibration.

An IMU will be randomly rotated until about 3,000 samples are collected. A 3D visualization of those samples is used to make sure that they distribute evenly on the ellipsoid. Then, Petrov's method [17] is used to fit an ellipsoid. This fitting provides 9 coefficients in a conic equation, the center, radii, and a rotation matrix. This information is used to create a transformation from raw data to calibrated data by.

$$ \begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} = H \times \begin{bmatrix} 1/\alpha & 0 & 0 \\ 0 & 1/\beta & 0 \\ 0 & 0 & 1/\gamma \end{bmatrix} \times H^T \times \left(\begin{bmatrix} m'_x \\ m'_y \\ m'_z \end{bmatrix} - \begin{bmatrix} o_x \\ o_y \\ o_z \end{bmatrix} \right) \quad (3) $$

m'_x, m'_y, and m'_z are the raw data. First, they are subtracted by the ellipsoid center (o_x, o_y, and o_z) to move its center

to the origin. H is a 3-by-3 rotation matrix that described the direction of 3 ellipsoid's radii. Parameter α, β, and γ are radii of 3 ellipsoid's axes according to the rotation matrix. This transformation scales the ellipsoid along 3 axes to have the same unit radius and provide a unit sphere as a result. m_x, m_y, and m_z are the calibrated result.

Although this problem can be fixed, there are a few more external factors that should be considered. First, soft-iron interference from some building structures will limit the usage to a local area where the magnetic field is fairly constant. Second, the device should be at least 1 foot away from other electronic devices or other ferromagnetic materials since they cannot be fixed in the calibration process. Third, the soft-iron interference is very sensitive. Even a rotation of a coin battery in its slot can change the shape of the ellipsoid.

2) Gyroscope Bias Removal: Gyroscope's bias is one of the most sensitive parameters in IMU as it varies by temperature. Our system has to keep tracking the bias during the operation. Every two consecutive samples from all 3 axes of the gyroscope are compared. If all three pairs have value difference within 3 deg/s (i.e. a measured range of static random noise), the value will be collected in the current bias average since the IMU is considerably stable. The average bias will change slowly and subtraction of this bias will work similarly to a low-pass filter.

3) Accelerometer Calibration: To get a very good accelerometer calibration, the sensor needs to measure the same gravitational acceleration from more than 30 different stable orientations. This could be done by a rotational stage or a robot arm. Due to the impracticality of the process for a normal user, the factory calibration setting that comes with the device is directly used.

F. Transformation Between Reference Frames

A method to find a transformation between two different sensing systems is to let 2 systems observe the same object. Then, the 2 different results will lead to a transformation.

1) Eagle-to-Kinect Transformation: A calibration panel with 4 reflective markers in L-shape (Fig.1b) is exposed to both Eagle and Kinect. The marker positions from Eagle are used to create the directions of the panel's X, Y, and Z-axis and its origin (${}_E P_o$). The 3 axes are turned to an orientation quaternion (${}^P_E q$) using van Waveren's method [16].

Unlike Eagle, Kinect cannot sense depth data on highly reflective surfaces and it needs some workarounds to get the panel's orientation and origin. First, four bright points on Kinect's infrared image are detected and localized in 2D. Using those four 2D positions, 3 non-reflective areas on the panel are estimated as shown in Fig.1b. Then, three groups of 3D points are collected, (1) X-axis group, (2) Y-axis group (3) panel plane group. Principal component analysis (PCA) is used in each group. The first principal component in group 1 and 2 will become panel's X-axis and panel's Y-axis respectively. The third principal component of group 3 will become panel's Z-axis. The 3 axes are converted to an orientation quaternion (${}^P_K q$) using van Waveren's method [16]. The virtual intersection point between X and Y-axis will become the panel's origin in Kinect reference frame (${}_K P_o$).

The orientation between both frames (${}^E_K q$) and the Eagle origin position in Kinect reference frame (${}_K E_o$) are

$$ {}^E_K q = {}^P_K q \times {}^P_E q^{-1} \tag{4} $$

$$ {}_K E_o = {}_K P_o - rot({}^E_K q, {}_E P_o) \tag{5} $$

Now, any position in Eagle reference frame (${}_E V$) can be transformed to a position in Kinect reference frame (${}_K V$) by

$$ {}_K V = {}_K E_o + rot({}^E_K q, {}_E V) \tag{6} $$

2) World-to-Kinect Transformation: In this case, only an orientation of the world reference frame in Kinect reference frame (${}^W_K q$) is needed. Our method follows Eq.7.

$$ {}^W_K q = {}^B_K q \times {}^B_W q^{-1} \tag{7} $$

${}^B_W q$ is an output from the base-IMU internal orientation filter which is always available. ${}^B_K q$ is a constant orientation quaternion of base-IMU in Kinect reference frame which needs to be found.

An IMU is attached on the calibration panel by visually aligning the IMU axes parallel to the panel axes. The method explained in II-F.1 is used to get ${}^P_K q$ from Kinect while the orientation filter running on the IMU gives the panel orientation in the world reference frame (${}^P_W q$). ${}^B_K q$ can be calculated from

$$ {}^B_K q = {}^P_K q \times {}^P_W q^{-1} \times {}^B_W q \tag{8} $$

G. Time Synchronization

There are 2 main steps in our time synchronization. The first step is to know time offset between base-IMU clock and Kinect clock using sound-based synchronization. The second step is to know time offset between the base-IMU clock and the arm-IMU clock via broadcast-based synchronization.

1) Sound-based Synchronization: Buzzer of base-IMU is extended and attached right in front of Kinect's microphone array. With this setting, 2 events can be triggered at the same time. The first event is when base-IMU commands the buzzer to stop its beep sound. The second is when Kinect stops hearing the sound. These 2 timestamps on different devices are used to find a time offset between the two systems.

2) Broadcast-Based Synchronization: Our synchronization exploits advertising capability of Bluetooth Low Energy (BLE) which has a broadcasting behavior. Our method works on an assumption that devices that receive the same advertisement packet always receive it at the same time. The procedure can be described as the following. The main computer will broadcast a single advertisement packet (non-connectable advertisement). Then, each IMU tries to scan for the packet. If it can receive, it will report the receiving timestamp to the PC. Because the BLE advertisement is unreliable, PC has to broadcast multiple rounds until all 2 IMUs report their timestamp. With the information, we can calculate the time differences among IMUs. Since the BLE clock has the resolution of 0.3125 ms, this method can provide a sub-millisecond accuracy. However, this method is not limited to 2 wireless nodes.

H. Rigid Forearm Model & Pre-session Measurement

A right forearm is modeled as a rigid body. In the forearm reference frame, Y-axis is in the elbow-to-wrist direction; X-axis points in the hand's dorsal side direction; and Z-axis to the thumb side. Its origin will be at the center of the black circular strap that ties IMU to the wrist. However, this model varies due to the way the user ties IMU to the wrist. Therefore, four parameters which are (1) radius from the forearm core to the strap surface (r), (2) strap-to-elbow distance (d_{se}) and (3) strap-to-wrist distance (d_{sw}), and (4) the IMU orientation relative to the forearm (${}^{A}_{F}q$) need to be measured at the beginning of each session.

The user has to raise his right forearm to about head level and faces his palm to the Kinect. Both upper arm and forearm will be on the coronal plane as shown in Fig.3b. Because the hand is open in a clear position, Kinect SDK 2.0 is able to provide the orientation of forearm (${}^{F}_{K}q'$) together with 3D-position of wrist and elbow. The arm-IMU orientation (${}^{A}_{W}q$) is also available from II-D. Therefore, a constant orientation offset between arm-IMU and forearm will be

$$ {}^{A}_{F}q = {}^{F}_{K}q'^{-1} \times {}^{W}_{K}q \times {}^{A}_{W}q \tag{9} $$

The method described in II-I is used to get 3D position of the black strap surface facing the camera. The shortest distance from the strap position to the forearm core (i.e. elbow-to-wrist line) is assigned to parameter r. Strap-to-elbow distance (d_{se}), strap-to-wrist distance (d_{sw}), and shoulder-to-elbow distance (d_{ul}) (i.e. upper arm length) are also measured simultaneously. After the measurement, forearm frame will have coordinate of elbow, wrist, and arm-IMU at $(0, -d_{se}, 0)$, $(0, +d_{sw}, 0)$, and $(+r, 0, 0)$ as seen in Fig.2.

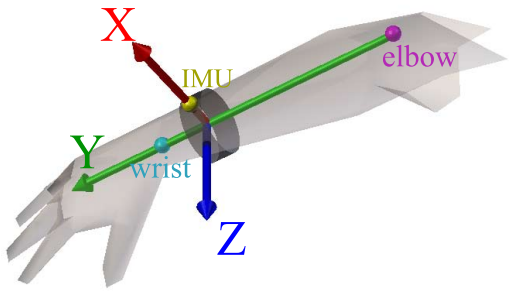


Fig. 2: Right forearm rigid model with a black strap.

I. IMU Position Estimation in a Non-occluded Moment

At each joint, there are 3 possible tracking statuses from Kinect SDK, *tracked*, *inferred*, and *not tracked*. While the wrist and the elbow are *tracked*, the following method is used to estimate the IMU position.

3D positions of the wrist and the elbow are projected to the Kinect's 2D RGB image. A sequence of pixels' illumination value between the 2 projected points will be extracted as grayscale. A moving average is used to search for the darkest part in the series which is expected to be the black strap. This index in the sequence will be mapped back to a pixel position on the RGB image and a pixel position on the depth image respectively using *CoordinateMapper* tool. Then, the 3D position of the seen part of the black strap can be retrieved.

However, that position is not arm-IMU's. This position is used to estimate the origin of the rigid forearm frame (${}_{K}F_{o}$) described in II-H by projecting the strap position onto the elbow-to-wrist line segment. Then, the sensor position is estimated by

$$ {}_{K}V_{imu} = {}_{K}F_{o} + rot({}^{F}_{K}q, {}_{F}[r, 0, 0]) \tag{10} $$

At this point, all 6-DOF information of forearm model is found in Kinect reference frame.

J. What is an occluded frame?

The occlusion scenario happens when the black strap is undetected or the visual position does not agree with the forearm rigid model. Usually, search for the black strap will be rejected when the state of the elbow or the wrist become *inferred* or *not tracked*. Occasionally, Kinect SDK might show *tracked* state while the given joint positions are wrong. This case will lead to a wrong black strap position which needs to be screened out. The following criteria are used to accept or reject the visual clue.

1) Forearm Y-axis Criteria: Elbow-to-wrist direction from visual data must be within 30 degree range from inertial data ($rot({}^{F}_{K}q, {}_{F}[0, 1, 0])$). Moreover, if the forearm is pointing nearly parallel to the camera ray, the visual forearm data will be rejected because the depth sensing is not accurate when the surface is not facing the camera.

2) Strap Position Criteria: The detected strap position must be within 5 cm from the visual elbow-to-wrist line segment and the strap should be closer to the wrist more than to the elbow. Otherwise, the forearm visual data of this frame will be rejected.

K. Input for IMU Position Recovery Process

For an occlusion gap, the following data are used to recover the missing trajectory of the arm-IMU.

1) Boundary Velocity and Position: Usually, the visual data during the transition in and out from occlusion are noisy due to the *depth inhomogeneity* error [18]. Therefore, the estimation of velocity and position right before and right after the occlusion cannot be done by a few samples. Available arm-IMU 3D positions over 500 ms before the occlusion are collected and fed into a linear least square fitting to estimate both initial position (p_0) and initial velocity (v_0). Similarly, a group of positions 500 ms after the occlusion are used to estimate the final position (p_n) and the final velocity (v_n).

2) IMU Orientation (${}^{A}_{K}q$) & Forearm Orientation (${}^{F}_{K}q$): They are available at 80Hz during the occlusion using ${}^{A}_{W}q$ from II-D, ${}^{W}_{K}q$ from Eq.7, and ${}^{A}_{F}q$ from Eq.9 with

$$ {}^{A}_{K}q = {}^{W}_{K}q \times {}^{A}_{W}q \;\; and \;\; {}^{F}_{K}q = {}^{A}_{K}q \times {}^{A}_{F}q^{-1} \tag{11} $$

3) Net Acceleration: Measured acceleration from IMU will be converted to Kinect reference frame and subtracted by the gravitational acceleration (9.81 m/s^2) by

$$ {}_{K}a_{net} = rot({}^{A}_{K}q, {}_{A}a_{measured}) - rot({}^{W}_{K}q, {}_{W}[0, 0, 9.81]) \tag{12} $$

4) Shoulder Position: Shoulder position normally moves slowly and is usually tracked. Therefore, a linear interpolation is used to fill in when the shoulder status is *not tracked*. This shoulder position together with the measured upper arm length (d_{ul}) will be used to correct forearm position.

L. IMU Position Recovery in an Occluded Moment

The process works as follows.

1) First Integration & Velocity Drift Correction: Start from initial velocity (v_0), ${}_K a_{net}$ is integrated directly overtime at each IMU timestamp until it reaches the ending of the occlusion with a velocity (${v_n}'$). Then, the velocity at the timestamp t_i will be added by $(v_n - {v_n}') \times (t_i - t_0)/(t_n - t_0)$ to make a smooth velocity transition toward v_n.

2) Second Integration & Position Drift Correction: Those corrected velocities are integrated overtime from initial position (p_0) until it reaches the ending of occlusion with a position (${p_n}'$). Then, the position at the timestamp t_i will be moved by $(p_n - {p_n}') \times (t_i - t_0)/(t_n - t_0)$ to bend the trajectory to p_n when it leaves the occlusion.

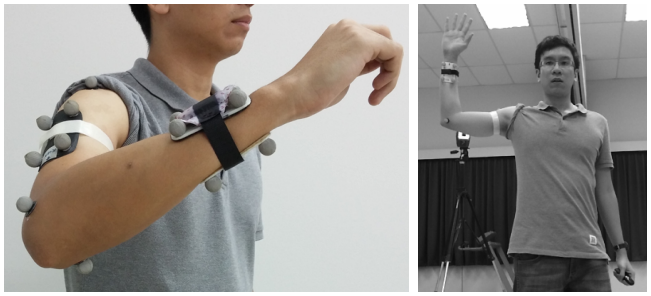
3) Correct Position using Shoulder Constraint: The position from step two will be translated as shortly as possible so that the estimated elbow position has a distance from the shoulder joint equal to the upper arm length (d_{ul}) measured in II-H. This can be seen as snapping a point radially to a sphere surface with radius d_{ul} while the sphere center (${}_K C$) is at

$$ {}_K C = {}_K S - rot({}^F_K q, {}_F[-r, -d_{se}, 0]) \tag{13}$$

${}_K S$ is the shoulder position in Kinect reference frame.

III. Evaluation

Eagle Digital RealTime System, an 8-camera marker-based motion capture system, is used as a ground truth. Fifteen markers are attached on the subject's right arm as shown in Fig.3a. To prepare for highly occluded movements, 8 markers are dedicated to the IMU position.



(a) Markers & IMU setup (b) Initial pose

Fig. 3: (a) The black strap should not be occluded by those reflective markers. (b) The initial pose in illumination channel from Kinect's RGB camera perspective

The subject stood at 2 meters away from Kinect and started his movement with the initial pose described in II-H for the system to measure r, d_{se}, d_{sw}, ${}^A_F q$, and d_{ul}. Then, he moved his forearm to the region behind his head and moved it back to the initial position. After that, he put his forearm in front of his belly, moved it to the region behind his back, and moved back to the initial position again.

Our focus is on the estimated arm-IMU position and forearm orientation from our fusion system and from Kinect SDK. An orientation difference from the gold standard (deg_{err}) can be shown as a degree of rotation by

$$ deg' = \arccos(\Re({}^F_K q_{measured}{}^{-1} \times {}^F_K q_{gold})) \times \frac{360}{\pi} \tag{14}$$

$$ deg_{err} = \min(deg', 360 - deg') \tag{15}$$

while $\Re(.)$ gives the real part of the input quaternion.

Since Kinect SDK cannot provide IMU position directly, IMU position estimation from Kinect SDK (${}_K D$) will be represented by an interpolation from SDK's elbow position (${}_K L$) and wrist position(${}_K T$).

$$ {}_K D = {}_K L + \frac{d_{se}}{d_{sw} + d_{se}} \times ({}_K T - {}_K L) \tag{16}$$

IV. Result and Discussion

A. Forearm Orientation

The forearm orientation errors from Kinect SDK and our system are shown in Fig.4. As expected, Kinect SDK 2.0 is not capable of forearm orientation estimation in both occlusion periods. While the forearm is in front of the belly in the middle of the recording, the orientation becomes highly unstable, especially for pronation/supination axis. Interestingly, as circled in Fig.4, the SDK is stuck at the 180-degree-flipped result at the end since it cannot differentiate the palm side from the dorsal side. On the other hand, our system show more consistent result with an error range of 6.8 - 17.4 degree. The error could come from multiple sources such as the measuring of ${}^A_F q$, skin tension artifact that can change marker position and rotate IMU attachment or external magnetic disturbances.

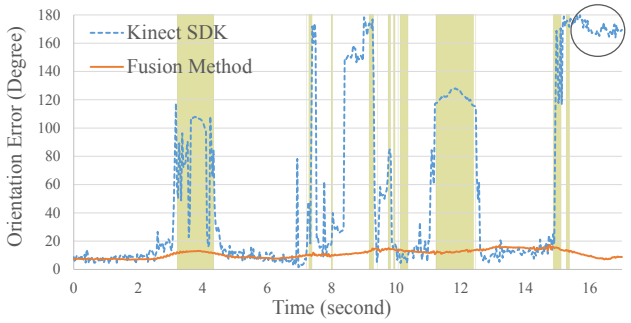


Fig. 4: Orientation errors. Occluded frames are highlighted.

B. IMU Position

The IMU position errors from Kinect SDK and our system are shown in Fig.5. Kinect SDK's elbow and wrist inferring cannot do well in occlusion especially for the first one (i.e. behind the head). On the other hand, our system could successfully recover IMU trajectory with a few centimeters of errors in the first occlusion. However, for the second occlusion (i.e. behind the back), our recovered trajectory

can follow the shape of the ground truth, but its size is smaller at the center of the occlusion which generates the maximum error to about 14 cm. Fig.6. shows the position of X component. Our analysis has found that the trajectory turns back before it should because the initial velocity (v_0) is smaller than what it should be.

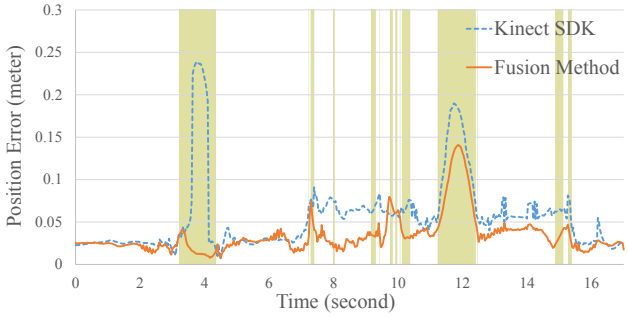


Fig. 5: Position error. The occluded frames are highlighted.

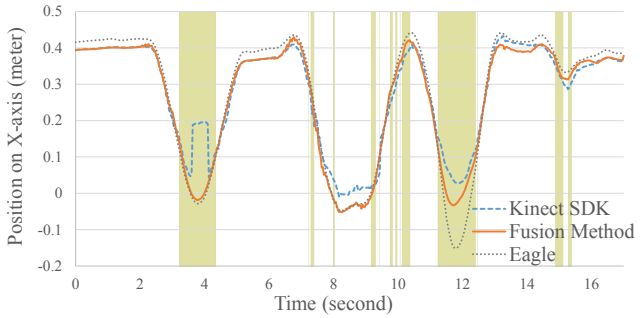


Fig. 6: X-axis position in Kinect reference frame. The occluded frames are highlighted.

TABLE I: Position and Orientation Errors

		Kinect SDK	Fusion Method	Error Reduction
Orientation Errors (degree)	RMS	84.25	11.34	86.54%
	Maximum	179.79	17.43	90.30%
Position Errors (meter)	RMS	0.07032	0.04071	42.11%
	Maximum	0.23921	0.14098	41.06%

V. CONCLUSIONS

TABLE I summarizes the errors comparison between Kinect SDK's skeleton tracking and our fusion technique. With only one additional IMU per arm, it complements the Kinect capability with a 3-DOF forearm orientation and the better tracking in occlusion. This non-obtrusive configuration has a potential to improve the quality of home rehabilitation assessment to the next level.

Since the algorithm is non-causal, the occluded trajectory cannot be recovered until the forearm leaves the occlusion. As a result, the algorithm is limited to non-real-time applications. However, for some specific application with pre-known movement, the system could be extended by an integration with statistical trajectory information to reduce the velocity and position drift in real-time during the occlusion.

REFERENCES

[1] A. Pfister, A. M. West, S. Bronner, and J. A. Noah, "Comparative abilities of microsoft kinect and vicon 3d motion capture for gait analysis," *Journal of medical engineering & technology*, vol. 38, no. 5, pp. 274–280, 2014.

[2] H. Mokhtarzadeh, A. Emmens, L. V. S. Peter, and O. Denny, "Feasibility evaluation of the microsoft kinect as a low cost alternative to motion capture systems," in *Australian & New Zealand Orthopaedic Research Society, 17th Annual meeting*, 2012.

[3] S. Obdrzalek, G. Kurillo, F. Ofli, R. Bajcsy, E. Seto, H. Jimison, and M. Pavel, "Accuracy and robustness of kinect pose estimation in the context of coaching of elderly population," in *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, aug 2012, pp. 1188–1193.

[4] B. Galna, G. Barry, D. Jackson, D. Mhiripiri, P. Olivier, and L. Rochester, "Accuracy of the microsoft kinect sensor for measuring movement in people with parkinson's disease," *Gait & Posture*, vol. 39, no. 4, pp. 1062–1068, apr 2014.

[5] B. F. Mentiplay, L. G. Perraton, K. J. Bower, Y.-H. Pua, R. McGaw, S. Heywood, and R. A. Clark, "Gait assessment using the microsoft xbox one kinect: Concurrent validity and inter-day reliability of spatiotemporal and kinematic variables," *Journal of Biomechanics*, 2015.

[6] X. Xu and R. W. McGorry, "The validity of the first and second generation microsoft kinect for identifying joint center locations during static postures," *Applied Ergonomics*, vol. 49, pp. 47–54, jul 2015.

[7] T. Todoroff, J. Leroy, and C. Picard-Limpens, "Orchestra: Wireless sensor system for augmented performances & fusion with kinect," *QPSR of the numediart research program*, vol. 4, no. 2, pp. 29–36, 2011.

[8] F. Destelle, A. Ahmadi, N. E. O'Connor, K. Moran, A. Chatzitofis, D. Zarpalas, and P. Daras, "Low-cost accurate skeleton tracking based on fusion of kinect and wearable inertial sensors," in *Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European*, sep 2014, pp. 371–375.

[9] S. O. Madgwick, "An efficient orientation filter for inertial and inertial/magnetic sensor arrays," 2010, unpublished. [Online]. Available: http://oodgeroo.ucsd.edu//~bob/estimation/Estimation_Winter_2015/Notes_and_docs_files/An_efficient_orientation_filter_for_inertial_and_inertialmagnetic_sensor_arrays.pdf

[10] S. Raghuraman, K. Venkatraman, Z. Wang, J. Wu, J. Clements, R. Lotfian, B. Prabhakaran, X. Guo, R. Jafari, and K. Nahrstedt, "Immersive multiplayer tennis with microsoft kinect and body sensor networks," in *Proceedings of the 20th ACM international conference on Multimedia*. 2396526: ACM, 2012, pp. 1481–1484.

[11] T. Helten, M. Muller, H. P. Seidel, and C. Theobalt, "Real-time body tracking with one depth camera and inertial sensors," in *Computer Vision (ICCV), 2013 IEEE International Conference on*, dec 2013, pp. 1105–1112.

[12] A. Baak, M. Mller, G. Bharaj, H.-P. Seidel, and C. Theobalt, *A Data-Driven Approach for Real-Time Full Body Pose Reconstruction from a Depth Camera*, ser. Consumer Depth Cameras for Computer Vision. Springer London, 2013, ch. 5, pp. 71–98.

[13] Y. Tian, X. Meng, D. Tao, D. Liu, and C. Feng, "Upper limb motion tracking with the integration of imu and kinect," *Neurocomputing*, vol. 159, pp. 207–218, jul 2015.

[14] O. J. Woodman, "An introduction to inertial navigation," University of Cambridge, Computer Laboratory, Tech. Rep. UCAMCL-TR-696, aug 2007. [Online]. Available: http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-696.pdf

[15] D. Comotti, M. Ermidoro, M. Galizzi, and A. Vitali, "Development of a wireless low-power multi-sensor network for motion tracking applications," in *Body Sensor Networks (BSN), 2013 IEEE International Conference on*, may 2013, pp. 1–6.

[16] J. van Waveren, "From quaternion to matrix and back," organization, feb 2005. [Online]. Available: http://fabiensanglard.net/doom3_documentation/37726-293748.pdf

[17] Y. Petrov. (2009) Ellipsoid fit. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/24693-ellipsoid-fit

[18] A. Kolb and F. Pece, *Range Imaging*, 1st ed., ser. Digital Representations of the Real World: How to Capture, Model, and Render Visual Reality. CRC Press, 2015, ch. 4, pp. 51–64.