Piotr KOPNIAK
INSTYTUT INFORMATYKI, WEII, POLITECHNIKA LUBELSKA,
ul. Nadbystrzycka 36B, 20-618 Lublin

Rejestracja ruchu za pomocą urządzenia Microsoft Kinect

Dr inż. Piotr KOPNIAK

Jest adiunktem w Zakładzie Ochrony Informacji Instytutu Informatyki na Wydziale Elektrotechniki i Informatyki Politechniki Lubelskiej. Obecnie w pracy naukowej zajmuje się badaniami związanymi z przetwarzaniem obrazu i przechwytywaniem ruchu (ang. Motion Capure). Prowadzi zajęcia m.in. z programowania w języku Java oraz programowania urządzeń mobilnych.

e-mail: copy@pluton.pol.lublin.pl

Streszczenie

Artykuł dotyczy tematyki rejestracji ruchu postaci ludzkiej (ang. Motion Capture). Praca pokazuje jak do rejestracji ruchu można wykorzystać kontroler ruchu Microsoft Kinect. Praca zawiera wyniki badań weryfikujących jakość oprogramowania NITE służącego do wyznaczania rotacji członków ciała w stawach. Badania wykazały, że jakość rejestrowanego ruchu nie jest wysoka. Mimo tego uzyskane dane mogą być wykorzystane do analizy gestów lub rozpoznawania ruchu.

Słowa kluczowe: rejestracja ruchu, grafika 3D, kontrolery gier.

Motion capture with Microsoft Kinect sensor

Abstract

This paper refers to a subject of a human body motion capture. Common video based motion capture systems are very expensive. This paper shows that it is possible to use a cheaper solution for motion capture which is a Microsoft Kinect movement controller. There is presented the software for motion capture of a human skeleton. We may use Microsoft SDK programmer toolkit or freeware frameworks like CL NUI Platform, OpenKinect and OpenNI for this purpose. We used OpenNI and NITE libraries (diagram in Fig. 2) for our research. The results of measurements of human joint angles with use of Kinect and our OpenNI based application are given and discussed. These angles are used for recording the movement into motion capture BVH files. We used free application Brekel Kinect (Fig. 3) for this purpose. The recorded data can be used for a realistic animation development. In our case, the data was used as an input source for our own visualization application made in Java language. The quality of recorded motion is not very high due to the calculation delays and problems with appropriate joint localization by the NITE software. The Kinect controller is not suitable for very accurate motion capture, e.g. for the needs of biomedicine, but the captured data are of sufficient quality for research related to the analysis of gestures and remote control.

Keywords: motion capture, 3D graphics, game controllers.

1. Wstęp

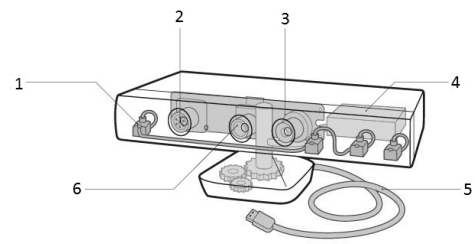
Techniki rejestracji ruchu (ang. Motion Capture lub w skrócie - mocap) wykorzystywane są najczęściej do tworzenia realistycznych animacji postaci stworzonych komputerowo na podstawie rzeczywistego ruchu postaci ludzkich i zwierząt. Możliwość dokładnego mierzenia przyspieszeń oraz pozycji poszczególnych członków ciała ludzkiego spowodowała także rozwój różnych zastosowań biomedycznych rejestracji ruchu i usprawniła trening sportowców.

Istniejące systemy mocap różnią się technologicznie. Wśród nich znajdują się systemy: mechaniczne - wykorzystujące zewnętrzny egzoszkielet z potencjometrami umieszczonymi w miejscach połączeń stawowych, systemy magnetyczne - analizujące zmiany w polu magnetycznym (wykorzystywane w samolotach bojowych), nowoczesne systemy wykorzystujące mocowane na ciele akcelerometry, np. system Xsens oraz systemy optyczne [1, 2]. Najczęściej mamy do czynienia z systemami optycznymi znacznikowymi – składającymi się z zestawu kamer rejestrujących ruch odpowiednio rozmieszczonych na ciele aktora znaczników. Przykładem takiego rozwiązania jest system Vicon. Nowszym rozwiązaniem są systemy bezznacznikowe, które analizują sylwetkę aktora dzięki danym z kilku kamer lub poprzez skanowanie 3D, a następnie wyliczają położenie szkieletu postaci. Przykładem tego typu systemu jest wprowadzony na rynek w 2010 roku czujnik ruchu Kinect do konsoli gier Microsoft Xbox 360. Jest to pierwsze tanie rozwiązanie dostępne dla szerokiego grona odbiorców. W związku z tym istotne jest zbadanie jego możliwości.

2. Kontroler Kinect

Kontroler Kinect jest urządzeniem zbudowanym przez firmę Microsoft na podstawie technologii firmy PrimeSense. Elementy składowe urządzenia przedstawiono na rysunku rys. 1 [3].

Kinect wykorzystuje procesor PrimeSense [4] generujący mapę głębokości obserwowanej przestrzeni. Współpracuje on z promiennikiem podczerwieni (element 2 na rys.1) i dwoma kamerami CMOS (elementy 3 i 6 na rys.1). Jedna z kamer jest kamerą RGB, druga, kamera głębokości jest monochromatycznym odbiornikiem światła podczerwonego. Urządzenie wyposażono także w silnik sterujący nachyleniem urządzenia (element 4). Kinect posiada także zestaw czterech mikrofonów z filtracją zakłóceń służących do rozpoznawania mowy (elementy 1 na rys. 1) [5].



Rys. 1. Elementy wewnętrzne kontrolera Kinect
Fig. 1. Inside elements of the Kinect controller

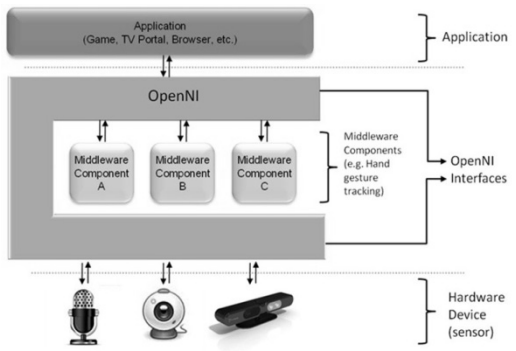
3. Oprogramowanie sterujące

Ze względu na potencjalnie dużą różnorodność zastosowań urządzenia Kinect szybko pojawiły się w sieci różne rozwiązania programistyczne umożliwiające podłączenie kontrolera do komputera PC. Należą do nich między innymi: CL NUI Platform [7], OpenKinect [8] oraz OpenNI [9] dostępne dla platform Windows, MacOSX, Linux i ARM. W odpowiedzi na zainteresowanie rynku firma Microsoft udostępniła w czerwcu 2011 roku SDK do niekomercyjnego wykorzystania kontrolera z systemem Windows, a w lutym 2012 SDK do zastosowań komercyjnych [10].

Najbardziej popularnym otwartym, wieloplatformowym szkieletem programistycznym dla różnych języków programowania jest produkt organizacji OpenNI [9]. Koncepcję OpenNI przedstawia diagram na rys. 2 [11]. OpenNI jest szkieletem programistycznym umożliwiającym komunikację z czujnikami wizji i dźwięku (ang. Hardware Device) oraz oprogramowaniem analizującym dane audio i video zapisywane przez te urządzenia (ang. Middleware Component). Przykładem takiego oprogramowania jest komponent pośredni zwracający lokalizacje postaci na scenie na podstawie analizy wejściowego obrazu wideo.

OpenNI zapewnia rozdzielenie sprzętu i oprogramowania stanowiąc wspólną bazę współdziałania dla niezależnych twórców modułów sprzętowych i modułów programistycznych. Obecnie wspierane moduły przez OpenNI to:

- **moduły sprzętowe:** czujnik 3D, kamera RGB, kamera podczerwieni oraz mikrofony,
- **moduły programowe:** analizator ciała ludzkiego (wyznaczający stawy, środek ciężkości, itd.), analizator lokalizacji dłoni, analizator gestów (porównujący rejestrowany ruch ze znanymi gestami), analizator sceny (identyfikujący podłogę i poszczególne obiekty sceny).



Rys. 2. Trójwarstwowa koncepcja szkieletu OpenNI
Fig. 2. Three-layered view of the OpenNI Concept

Programowe moduły obliczeniowe dla OpenNI firmy PrimeSense zawarte są w bibliotece NITE. Oferują one trzy podstawowe funkcjonalności: rozpoznawanie postaci ludzkich i ich otoczenia, śledzenie rąk oraz śledzenie wszystkich elementów szkieletu postaci [12].

Do prawidłowej obsługi kontrolera Kinect poprzez OpenNI wymagana jest instalacja kilku składników. Pierwszym składnikiem jest sam szkielet OpenNI [13], drugim jest sterownik sprzętowy dla urządzenia Kinect [14], a trzecim moduły programowe dla szkieletu OpenNI czyli biblioteka NITE [13].

4. Pomiar kątów ugięcia stawów

Systemy mocap zapisują zarejestrowane dane w różnych formatach plików. Najpopularniejsze z nich to: binarny format C3D wykorzystywany przez system Vicon, format FBX stosowany w produktach firmy Autodesk oraz tekstowy format firmy BioVision – BVH, stosowany w wielu rozwiązaniach mocap np. w systemie Xsens [15]. Format BVH został wybrany jako podstawowy format pliku z danymi o ruchu w naszych badaniach, które dotyczą automatycznego rozpoznawania ruchu i sekwencji gestów.

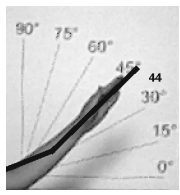
W plikach mocap zapisywane są wymiary szkieletu postaci oraz dane o ruchu jego kości, czyli o zmianach ich lokalizacji przestrzennej lub rotacji w stawach w funkcji czasu. Interesujące nas pliki BVH zawierają zapisy rotacji, w związku z tym przeprowadzono badania skuteczności pomiaru poprzez kontroler Kinect i jego oprogramowanie kątów obrotu kości w stawach.

Do badań wybrano staw łokciowy ręki, dla którego porównano kąty zmierzone przez kontroler i kąty rzeczywistego ugięcia wyznaczone za pomocą przygotowanej do badań planszy. Na planszy oznaczono liniami kąty 0°, 15°, 30°, 45°, 60° i 90° (rys. 3). Pomiarów dokonano 30 razy dla każdego kąta w płaszczyznach wyznaczonych przez osie X i Y, Y i Z oraz X i Z w lewoskrętnym układzie współrzędnych przestrzeni 3D. Oś widzenia kamery była zgodna z osią Z układu współrzędnych, a wektor kierunku patrzenia miał zwrot przeciwny do zwrotu osi. Wyniki średnie badań przedstawia tabela 1.

Otrzymane wyniki pokazują, że wartości zmierzone zbliżone do prawidłowych występują tylko w płaszczyźnie prostopadłej do kierunku patrzenia kamery. Zastosowane oprogramowaniem ma problem z prawidłowym wyliczeniem położenia stawów (widoczny na rys.3) i ruchu kości w kierunku zgodnym z kierunkiem patrzenia kamery, dlatego do czasu poprawy algorytmów NITE uzyskane dane należy traktować jako dane przybliżone. Nie nadają się one do szczegółowej analizy, np. biomedycznej, ale wystarczą do analizy gestów i zastosowań zdalnego sterowania.

Tab. 1. Skuteczność pomiarów kątów zgięcia stawu w płaszczyznach XY, YZ i XZ
Tab. 1. The efficiency of joint bend angle measurement in XY, YZ and XZ planes

XY	Plansza	0°	15°	30°	45°	60°	75°	90°
	Pomiar	2°	16°	29°	43°	57°	73°	86°
YZ	Plansza	0°	15°	30°	45°	60°	75°	90°
	Pomiar	-1°	3°	10°	19°	32°	53°	94°
XZ	Plansza	0°	15°	30°	45°	60°	75°	90°
	Pomiar	9°	45°	62°	70°	78°	89°	93°

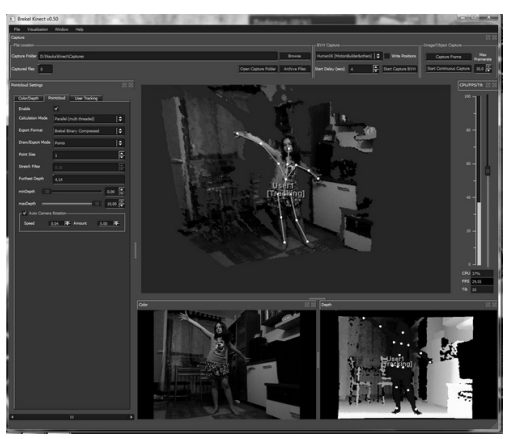


Rys. 3. Ręka z wyznaczonym szkieletem na tle planszy pomiarowej
Fig. 3. The arm with computed skeleton and measurement board in the background

5. Rejestracja ruchu – Brekel Kinect i pliki BVH

Pomimo małej dokładności dane z kontrolera Kinect będą wykorzystywane w naszych badaniach dotyczących rozpoznawania ruchu. Stworzony w języku Java moduł wczytujący i wizualizujący dane o ruchu korzysta z plików BVH. Plik BVH jest plikiem hierarchicznym, który, jak wspomniano wcześniej, zawiera dane o lokalizacji w przestrzeni kości szkieletu rejestrowanej postaci oraz tablicę wartości rzeczywistych odpowiadających rotacjom kości w stawach w funkcji czasu.

Szkielet OpenNI z NITE oferuje funkcje wyznaczania położenia szkieletu i rejestracji rotacji kości. W związku z tym można samodzielnie stworzyć program generujący pliki BVH. Można także wykorzystać gotową aplikację opartą o OpenNI - darmowy program Brekel Kinect [16]. Okno programu przedstawiono na rys. 4.



Rys. 4. Okno programu Brekel Kinect
Fig. 4. The main window of Brekel Kinect application

W oknie głównym programu wyświetlane są trzy obrazy. Główną część stanowi trójwymiarowy model widzianej przez kontroler Kinect sceny z widocznymi postaciami i ich szkieletami oraz nałożonymi teksturami. Drugi po lewej stronie na dole to obraz z kamery RGB, a trzeci po prawej to mapa głębokości z oznaczonymi postaciami i punktami, w których znajdują się ich stawy.

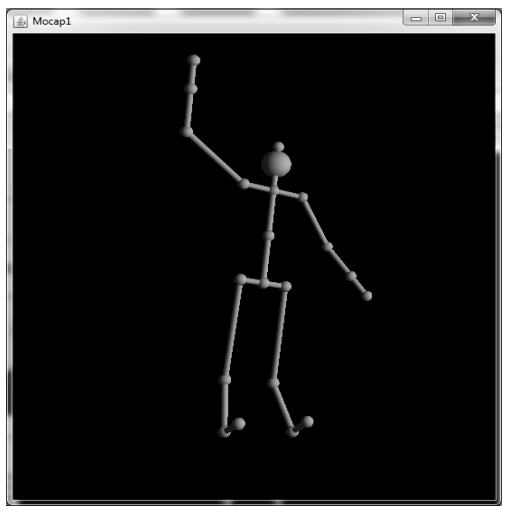
Program umożliwia dodatkową konfigurację generowanych plików BVH, poprzez dostosowanie nazw kości dla różnych aplikacji, a także włączenie funkcji dopisywania do pliku zawierającego dane rotacji danych o lokalizacji przestrzennej.

6. Wizualizacja zarejestrowanego ruchu

Zarejestrowane dane o ruchu postaci wykorzystano jako dane wejściowe dla autorskiej aplikacji badawczej opisanej szczegółowo w publikacji [15]. Aplikacja ma możliwość wizualizacji ruchu zapisanego w plikach BVH w celu dalszej jego analizy. Wczytywane pliki BVH przekształcane są wewnętrznie do własnego formatu plików XML, który stanowi podstawowy format zunifikowanej bazy danych ruchu. Aplikacja stworzona została w języku Java, a do generacji modelu szkieletu poruszającego się w przestrzeni wykorzystano darmowy silnik do tworzenia gier w języku Java – jMonkeyEngine [17].

Aplikacja wizualizacyjna jest pierwszym modułem tworzonego systemu gromadzenia danych o ruchu i jego automatycznego rozpoznawania. Baza danych o ruchu umożliwi jego automatyczne dopasowanie i rozpoznawanie na podstawie zgromadzonych wzorców, a także budowę kolejnych modułów do łączenia ruchów w dłuższe sekwencje, np. chód, podskok i bieg. Tworzony system umożliwi budowę realistycznych animacji na podstawie zgromadzonych danych bez konieczności przeprowadzania całego procesu przechwytywania ruchu dla innej sekwencji póz i gestów.

Okno aplikacji z animowanym modelem szkieletu zbudowanym dynamicznie na podstawie danych uzyskanych z kontrolera Kinect i zapisanych do pliku BVH przedstawia rysunek 5.



Rys. 5. Wizualizacja ruchu w aplikacji zbudowanej do badań
Fig. 5. Motion visualization in the application built for research

7. Weryfikacja badanego rozwiązania

Celem prac badawczych była weryfikacja przydatności kontrolera Kinect w badaniach związanych z rejestracja ruchu. Do badań wykorzystano kontroler Kinect, stworzoną w języku Java aplikację do pomiaru kątów ugięcia stawów szkieletu oraz aplikację Brekel Kinect zapisującą dane o ruchu do plików.

Wyniki pomiarów wykazały, że algorytmy wyznaczające położenie stawów szkieletu biblioteki NITE są mało dokładne, a otrzymane dane nie nadają się do dokładnej analizy ruchu, np. w zastosowaniach ortopedycznych czy sportowych. Dlatego podjęte zostaną prace w celu usprawnienia oprogramowania lub zastosowania kilku kontrolerów Kinect jednocześnie rejestrujących ruch postaci z różnych stron. Jest to możliwe przy wykorzystaniu sterownika CL NUI. Otrzymane wyniki mogą być w takim przypadku uśrednione podnosząc dokładność wyznaczania położenia szkieletu.

Aplikacja Brekel Kinect prawidłowo współpracowała z urządzeniem, prawidłowo rejestrowała ruch i zapisywała dane do plików BVH. Wygenerowane pliki bez problemu odczytał moduł importu i wizualizacji danych aplikacji badawczej.

Kontroler Kinect stworzony został z myślą o sterowaniu gier za pomocą gestów i nie rejestruje położenia stóp, dłoni ani głowy. Dane te do plików BVH dodawane są przez oprogramowanie. Dlatego w plikach BVH dłonie stanowią przedłużenie przedramion, stopy to dodane prostopadle kości do podudzi, a głowa jest usztywniona względem barków. W związku z tym urządzenia Kinect nie da się wykorzystać do rejestracji ruchu całego ciała, co oferują droższe rozwiązania mocap.

Wygenerowane animacje były bardzo dobrej jakości w stosunku do ceny wykorzystanego systemu rejestracji ruchu, jednak nie były doskonałe. Zdarzały się drżenia kości oraz nienaturalne obroty kości w stawach. Widoczne jest to na rysunku 5, gdzie kolana postaci ugięte są w niewłaściwym kierunku. Uzyskane z badanego systemu dane wymagają dodatkowych poprawek, ale jest to naturalny etap oczyszczania danych i jest konieczny także w przypadku droższych systemów mocap.

Podsumowując można powiedzieć, że kontroler Kinect może być z powodzeniem wykorzystany nie tylko jako kontroler gier, ale także jako źródło danych do uzyskania realistycznych animacji niewielkim kosztem. Uzyskane dane nie są doskonale, ale przy wykorzystaniu kilku zsynchronizowanych urządzeń Kinect rejestrujących obraz jednocześnie ich jakość może znacznie wzrosnąć, a dane będą mogły być użyte na potrzeby dokładniejszych prac badawczych, np. związanych z systemami analizy ruchu.

Prace badawcze nie objęły analizy metrologicznej kontrolera Kinect ponieważ szczegółowe wyniki tego typu analizy zawiera publikacja [18].

8. Literatura

[1] Kitagawa M., Windsor B.: MoCap for Artists. Workflow and Techniques for Motion Capture. Elsevier, 2008.
[2] Menache A.: Understanding Motion Capture for Computer Animation and Video Games. Morgan Kaufmann, 2000.
[3] Schemat budowy kontrolera Kinect, http://www.wired.com/magazine/wp-content/images/19-07/mf_kinect2_f.jpg
[4] PrimeSense Supplies 3-D-Sensing Technology to "Project Natal" for Xbox 360, http://www.microsoft.com/en-us/news/press/2010/mar10/03-31PrimeSensePR.aspx
[5] Opis budowy kontrolera Kinect, http://msdn.microsoft.com/pl-pl/library/kinect-sdk--wprowadzenie.aspx
[6] Opis zasady działania kontrolera Kinect, http://www.wired.com/gadgetlab/2010/11/tonights-release-xbox-kinect-how-does-it-work/all/1
[7] Otwarta biblioteka CL NUI Platform: http://codelaboratories.com/kb/nui
[8] Projekt OpenKinect: http://openkinect.org/wiki/Main_Page
[9] Szkielet OpenNI: http://www.openni.org/
[10] Strona domowa Kinect for Windows: http://www.microsoft.com/en-us/kinectforwindows/
[11] Struktura szkieletu OpenNI, http://openni.org/Documentation/images/picture3.jpg
[12] Biblioteka NITE: http://www.primesense.com/en/nite
[13] Strona pobierania modułów OpenNI, http://www.openni.org/Downloads/OpenNIModules.aspx
[14] Zmodyfikowany sterownik kontrolera Kinect, https://github.com/avin2/SensorKinect/
[15] Kopniak P.: Motion Capture Data Visualization Developed with the Utilization of jMonkeyEngine. Computer Graphics. Selected Issues. University of Szczecin, Szczecin 2010.
[16] Strona domowa projektu Brekel Kinect, http://www.brekel.com/?page_id=155
[17] Strona domowa silnika gier jMonkeyEngine, http://www.jmonkey-engine.com/
[18] Andersen M.R., Jensen T., Lisouski P., Mortensen A.K., Hansen M.K.: Kinect Depth Sensor Evaluation for Computer Vision Applications. Technical report ECE-TR-6. Aarhus University, 2012.

otrzymano / received: 30.08.2012
przyjęto do druku / accepted: 01.10.2012 *artykuł recenzowany / revised paper*