Detecting 3D Position and Orientation of a Wii Remote Using Webcams

Jerry van den Heuvel and Jacco Bikker

NHTV - University of Applied Sciences, Breda, The Netherlands
mail@jerhill.com, bikker.j@nhtv.nl

Abstract. In this paper we present a method to detect the three dimensional position and orientation of a Wii Remote with one or more emissive spheres attached to it, providing an input device that has six degrees of freedom. Unlike other systems, our system can focus in different directions surrounding the user, with a high precision, and at a low cost. We describe the way object-, motion- and orientation tracking is done, as well as the applicability of the final product. We further describe how to improve the noisy data that is retrieved from the sensors of the Wii Remote, how to smooth detected positions, and how to extrapolate position and orientation.

Keywords: Wii-Remote, PlayStation-Move, Webcam, Triangulation, Object Tracking, Position, Orientation.

1 Introduction

Interactive entertainment systems where the player's actual body is used as primary input to interact with games, like Nintendo's Wii with the Wiimote, Sony's PlayStation3 with the Move and recently Microsoft's Xbox360 with the Kinect, received considerable attention [12]. All these systems require the player to face one direction: towards the screen where the game is displayed on. For environments where the player needs to be able to look in different directions, e.g. a setup where the player is surrounded by multiple screens, these systems do not suffice. The Wii Remote for example needs the sensor bar to determine its absolute position and orientation [32]. The PlayStation Move is only able to keep track of its exact position if it is visible to the Playstation Eye [26].

In this paper, we present a system that is able to determine the transformation, i.e. the orientation and position of a user input device, with six degrees of freedom, in a CAVE-like environment [29]. To reduce the cost of the system, we use a combination off-the-shelf hardware. A Nintendo Wiimote (extended with the MotionPlus sensor) is used to determine yaw, pitch and roll of the device. Two light orbs from the Playstation Move controller are used together with a number of low-cost cameras to increase the accuracy of the yaw and pitch, and to determine the position of the input device. Using four cameras, the player can face in any direction inside a cubical room.

We describe how the data from the various sensors and the cameras is combined in a filtered and stable final transformation that can be used to control Virtual Reality and Augmented Reality applications.

J.M. Allbeck and P. Faloutsos (Eds.): MIG 2011, LNCS 7060, pp. 448–458, 2011.

2 Previous Work

Determining the orientation and position of an input device is not new. In his seminal 1968 paper, Sutherland describes both mechanical and ultrasonic (carrier phase) head-tracking systems [28].

Later, Raab et al. proposed a three-axis input device based on quasi-static magnetic-dipole fields [23], which was originally invented by Kuipers [17]. More recently, this technique was refined by Paperno et al. [22], who use a rotating magnetic dipole to determine azimuth, elevation and distance from the source to a sensor.

In terms of low-cost tracking of an input device in the context of a CAVE environment, Sharlin et al. propose a wireless feet tracker that can be assembled for under $200 US, and achieves an accuracy of 10cm at 20Hz [27]. This approach does not detect precise movement, nor does it determine orientation.

Foxlin et al. describe a tracking system based on an inertial navigation system aided by ultrasonic time-of-flight range measurements to a constellation of wireless transponder beacons [8]. They report accuracies better than 2mm. This was further improved to 1mm accuracy for in-cockpit augmented reality by Foxlin et al. [9], and is commercially available from InterSense [14].

Commercial optical solutions are available from several companies, e.g. the FlashPoint system by Image Guided Technologies [13], the laserBIRD2 system by Ascension technology [1], and the CODA Motion Capture System by B&L Engineering [3].

Digital cameras are used in motion-capture systems such as the HiRes 3D Motion Capture System by the Motion Analysis Corporation [15; 21] to track a relatively large number of targets, albeit at a relatively low rate because of the need for 2-D image processing.

In the context of object tracking in 2D images, many approaches have been described. For our system, we use the approach of Derhgawen [6], who describes color based object tracking using thresholds to remove redundant colors from the image.

The use of Wii Remotes for a 3D user interface has been previously described by Wingrave et al. [32], who describe how theWii Remote can be used as a "spatially convenient device". They describe in detail how to determine the orientation and position of the Wii Remote and its applications.

3 In Theory

In this section, we describe how position and orientation is determined using the sensors in the Wii Remote and the Playstation Move system.

3.1 The Wii Remote and the Wii MotionPlus

The Wii Remote itself contains one internal motion sensor: an accelerometer, which reports acceleration in the device's x, y, and z directions, expressed in the unit of gravity g. The sensor bar, included with the Wii, is a box containing five IR-LEDs on each side (see Figure 1).

The Wii Remote detects up to four IR blobs [24], produced by any IR emitting source, using an infrared optical camera [5]. It then converts the tracked light blobs into 2D positions, along with their size in pixels, and sends this data to the Wii using a Bluetooth connection [32; 11].

The Wii MotionPlus is an extension for the Wii Remote and contains a gyroscope which detects the angular velocity about its *x*, *y* and *z* axis [32].

Combined, the data from these sensors is used to calculate the Wii Remote's precise position and orientation. Each individual Wii Remote collects its own data, which makes it possible to distinguish between different Wii Remotes.

Fig. 1. Wii's sensor bar

Determining Position

When the Wii Remote is pointed at the sensor bar, the Wii Remote picks up two points, $P_L = (x_L, y_L)$ and $P_R = (x_R, y_R)$ from the LED arrays, using the relative sizes of the points on the optical sensor's image. The distance d from the Wii Remote to the sensor bar can be found by calculating the distance corresponding to the left and right points on the optical sensor's image (see Equation 1, where $diam_{LED}$ is the diameter of the actual LED marker from the sensor bar, $diam_{L,R}$ is the diameters of the points in the optical sensor's image, θ is the optical sensor's viewing angle, m is the distance between the LEDs on the right and left side of the sensor bar and w_{img} is the width of the image taken from the optical sensor).

$$
\begin{aligned}
d &= \sqrt{d_L^2 + (m/2) - 2d_L(m/2)^2 cos(\phi)} \\
d_L &= \frac{w_L/2}{\tan(\theta/2)} \\
d_R &= \frac{w_R/2}{\tan(\theta/2)} \\
cos(\phi) &= \frac{d_L^2 m^2 - 2d_R^2}{2md_L} \\
w_L &= \frac{w_{img} \cdot diam_{LED}}{diam_L} \\
w_R &= \frac{w_{img} \cdot diam_{LED}}{diam_R}
\end{aligned}
\tag{1}
$$

Determining Orientation

The same points the Wii Remote used to detect its position ($PL = (x_L, y_L)$ and $PR = (x_R, y_R)$ - IR blobs produced by the sensor bar) can be used to calculate theWii Remote's rotation about its *y*-axis relative to its neutral orientation where *z*+ is facing upwards. This rotation (*roll)* is calculated with respect to the Wii Remote's *x*-axis (Equation 2).

$$\begin{aligned} roll &= \arccos\left(\vec{x} \cdot \frac{\vec{v}}{\|\vec{v}\|}\right) \\ \vec{x} &= (1,0) \\ \vec{v} &= P_L - P_R \end{aligned} \tag{2}$$

The accelerometer reports acceleration in the device's x, y, and z dimensions, expressed in the unit of gravity g. This reported acceleration vector is composed of the Wii Remote's actual acceleration a' and the Earth's gravity g, so obtaining the actual acceleration requires subtracting the gravity vector from the Wii Remote's reported acceleration a. Additionally, g is always oriented towards the Earth, so it is used to find a part of the Wii Remote's orientation (Equations 3 and 4).

$$pitch = \arccos\left({}^{a_{gz}}\!/_{a_{gy}}\right) \tag{3}$$

$$roll = \arccos\left({}^{a_{gz}}\!/_{a_{gx}}\right) \tag{4}$$

The accelerometers alone are not sufficient for calculating the yaw because the gravity vector aligns with the z-axis, meaning that it cannot be measured when theWii Remote rotates about its z-axis. Apart from that, the orientation can only be measured when there is no acceleration.

It is possible to combine the data retrieved from the gyroscopes of the Wii MotionPlus with the data retrieved from the accelerometer of the Wii Remote itself. This data however is noisy and needs to be filtered before it is useful.

In the context of VR/AR, the Kalman filter [16] is commonly used to reduce noise in the input signals [2; 18; 30; 31]. The Kalman filter is a set of mathematical equations that recursively estimate the state and error covariance of a signal, while minimizing the mean of the squared error covariance. We use the Kalman filter to reduce the noise of both sensors so the data of the gyroscopes can be merged with the data of the accelerometers. The absolute values of the accelerometer are used to get an estimate of the current orientation, and the relative values of the gyroscopes are used to improve this data and predict the orientation when there is acceleration.

3.2 PlayStation Move

The PlayStation Move motion controller contains three internal motion sensors: an accelerometer, a gyroscope and a terrestrial magnetic field sensor [11]. The terrestrial magnetic field sensor, also called magnetometer, resembles a compass and improves the detection of the device's motion and orientation. An external device called the PlayStation Eye, a 60 to 120Hz camera used to track the colored ball on top of the PlayStation Move, is needed to calculate the position of the device. The colored ball is also used to distinguish between different PlayStation Moves. Like theWii Remote, the PlayStation Move sends data via Bluetooth. At the time of writing, this data cannot be used on the PC.

Determining Position

The accelerometer and gyroscope of the PlayStation Move motion controller are used to get the relative position of the device. The PlayStation3, the PlayStation Eye and

the data received from the sensors are used to detect the precise movement and absolute position in 3D space of the PlayStation Move [26; 19]. The PlayStation Eye is used to track the glowing ball on top of the PlayStation Move where the position and the size of the ball, in camera space, are used to calculate the x, y and z position [20], with a precision of 1 millimeter in the *x*-axis and *y*-axis and a precision of 2 centimeters in the *z*-axis [19]. Detailed information about tracking algorithms and calculations has not been published by Sony, and so far, this information has not been reverse engineered.

Determining Orientation
As for the position, nothing is revealed yet. It is likely however that the data received from the motion sensors is used for this. The camera will not be able to tell the orientation of the glowing ball. Presumably, the magnetometer is used to determine the yaw of the device. The gyroscopes and accelerometer are more precise and accurate than the ones from the Wii Remote, and most probably used to calculate the pitch and roll.

3.3 Discussion

An important difference between the Wii Remote combined with MotionPlus and the PlayStation Move is the terrestrial magnetic field sensor which can be found in the PlayStation Move but is absent in the Wii Remote, with or without extension. This sensor is of great importance when determining the device's orientation, as it determines the absolute yaw of the device. Another difference is the fact that the Wii Remote is able to detect the position data itself while the PlayStation Move requires an external camera for this.

The Wii Remote currently is the best fit for our requirements: it is affordable and widely available, it is able to detect motion that can be used to deduct orientation and position and it provides convenient user input through buttons. Most important however is the fact that it is possible to connect the Wii Remote to any PC via Bluetooth. In theory, the PlayStation Move would be a better fit: it has the same capability and more accurate sensors to detect its motion and orientation. However, unfortunately there is no possibility to communicate with a PC, so it's not possible to control or receive data from this device.

To detect an object's position with six degrees of freedom, the above described methods are not enough. The Wii's method requires the user to point the Wii Remote at the sensor bar to be able to detect its position. The PlayStation Move on the other hand does not really need to be pointed at the PlayStation Eye, but the glowing ball needs to be visible to be able to track it. It thus still requires the user to keep the device in front of the camera. Combining both methods makes it possible to track an object from different angles. Using the PlayStation Move method to detect a colored object, in combination with theWii method of using triangulation to calculate the Wii Remote's position, creates new possibilities to determine the position of an object.

To detect the orientation of the object, the formulas provided earlier can be used (see Equations 2, 3 and 4). In case the calculated yaw will not be accurate enough it is considered to detect a second object's position that, together with the first object's position, can be used to determine the final yaw.

4 Implementation

In this section, we describe how our system determines the transform of the input device, which consists of a position and orientation. The position can be calculated using object tracking and triangulation. The orientation can be determined using the data from the sensors inside theWiimote and object tracking. Our system performs object tracking using color based detection. The delay that is caused by object detection is reduced by using prediction.

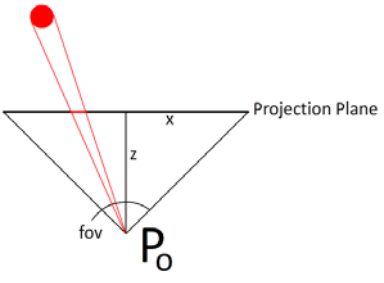

Fig. 2. Projection Explanation

4.1 Determining Position Using Triangulation

Using multiple cameras and triangulation allows us to detect the position of a single object.

To obtain a 3D position based on 2D images captured by the cameras, a minimum of two projected 2D positions is required. If only one camera is able to see the object, we can still approximate its position by calculating the distance from the sphere to the camera. Similar to the Playstation Move system, this distance can be determined using the size of the object. The actual size of the object relative to its size in the captured 2D image combined with the field of view allow us to estimate the distance of the object to the camera. However, when only a certain part of the object is visible, the smaller size on the captured image leads to a overestimation of the distance.

When the 2D position is found, a Z value is added to define the 3D position in camera space, where Z is the distance from a virtual point P_O to the plane the image is projected on (see Figure 2). The virtual point is calculated using the field of view of the webcam.

The 3D position is then used to construct a ray from the virtual point through this point. Once rays have been constructed and converted to world space, they are intersected, which yields a single position in world space, which is the position of the object.

4.2 Determining Orientation

Accelerometer

Values retrieved from the accelerometer are absolute acceleration values based on the gravitational force and can be converted to roll and pitch values ranging from -90 to 90 degrees (in the C++ libraries used to communicate with theWii Remote [25]).

These values contain considerable noise. To use these values optimally, they need to be in a range of 360 degrees instead of 180, and the noise needs to be filtered to be usable.

Gyroscopes
The gyroscopes detect a relative angular velocity over all three axes. The values retrieved from this sensors are slightly inaccurate and contain some noise as well. This problem is reduced by careful calibration. To calibrate the gyroscopes, the Wii Remote needs to be hold still while the motion data gets recorded. The recorded values can then be used to estimate noise in motion measurements, which can be subtracted from subsequent detected motion values to even out the imprecision.

Another potential problem is the time dependence of velocity measurements. If the measured data is used by independent devices, these devices need to agree on time. When the application is running on a PC, this is not trivial to do. Apart from this, we need to take latency into consideration. This latency is caused by the actual data transfer, and the time required to process the data.

Combining Sensor Data
To combine the data retrieved from the accelerometers with the data retrieved from the gyroscopes, we use the Kalman Filter [16].

The data of the accelerometers is used to create an acceleration vector. This vector holds the pitch and roll of the device. As this only works when the device is not accelerating, the data of the gyroscopes is used to create a second direction vector for orientation of the device. This vector is calculated using the estimated orientation of the previous frame and the currently measured speed. The two vectors are used to determine the final orientation vector for the current frame, by combining them using a fixed factor, based on the reliability of the gyroscopes.

Adding a Second Light Orb
To improve on the relative orientation data a second glowing ball (*light-orb*) with a different color is added, to retrieve an absolute orientation. Both light-orbs are tracked by the cameras, determining their positions. These positions are then used to calculate a directional vector. From this directional vector a pitch and yaw angle is derived. For the roll the absolute, filtered value retrieved from the accelerometer is used, corrected by the gyroscopes.

4.3 Combining Position and Orientation

Smoothing Position
The accuracy of the detection of objects in images captured by the cameras is affected by the noise in the captured camera images. The noise results in inaccurate size estimates for the blobs, and this in turn affects the 2D position as well as the final 3D position. The user perceives this as random movements, even when the object is not moving.

To reduce these random movements, damping is applied to the detected 3D position. For damping, we use Equation 5. In this equation, P is the estimated position, Pprev is the previous position, Plast is the last known position and d is the damping

factor. A small factor results in less damping, and an estimated position that is closer to the new position. A larger factor results in more damping, which causes the estimated position to be closer to the old position.

$$P = P_{prev} \times d + P_{last} \times (1 - d) \quad (5)$$

The damping factor is based on relative movement. If the movement is small, a lot of damping will be applied but when the movement is large, the damping factor will be small so only a little damping is applied. This results in smooth movement when the relative movement is small, while it is still possible to make fast movements without having a delay in the result, which would have been the case if a static damp factor was used.

By applying this smoothing to both light orbs, the orientation calculated using these is also smoother.

4.4 Object Tracking

To find an object in three dimensions using one or more cameras, the object must be detected in the captured 2D images. To be able to find an object, it needs to be recognizable and obvious, which explains the choice in the PlayStation Move system for an emissive colored ball. The ball shape is easily recongnized, as it is orientation independent. To make it detectable under most lighting conditions, the ball is made emissive and is able to change color. This way, the color of the ball can be chosen to contrast with the colors in the environment, which further improves detection [7].

Color-Based Object Detection

The easiest way to recognize an object in an image is by looking for its color, so it is best to have an object consisting of one solid color. Ideally, this color does not occur in the image, apart from the object that is to be detected. Different lighting should cause the object to change color. For this purpose, we created a custom version of the light orb, where we attach the glowing ball from the PlayStation Move to a Wiimote.

Camera Settings

Camera settings affect the precision of the color detection. Optimal settings reduce the need to filter. In Figure 3, it is clearly visible that the webcam filters out most of the unwanted environment.

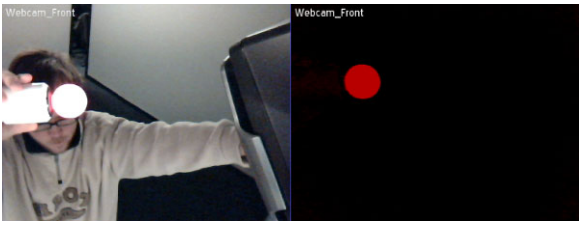

Fig. 3. Camera Settings – Left: Default, Right: Low Exposure

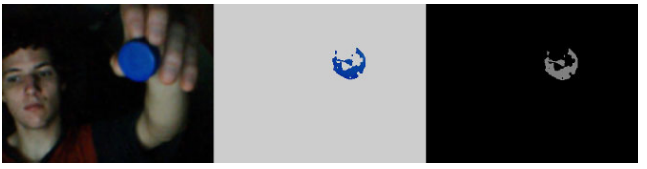

Fig. 4. Detecting a color in an image. Left: original; center: filtered color; right: color mask.

For color detection, we essentially filter out every other color in the image, until the only color that is left is the object color. This results in an image containing blobs where the colors matched (see Figure 4).

Nearby blobs are then combined. The result is used to determine the size and position of potential objects. Ideally, there should be only one object, which is generally the case if the color was picked well and the camera settings are correct. If we do however find more than a single object, we chose the largest blob.

Prediction

The rate at which the webcams provide images is independent from the update speed of the motion tracking system. When the system is updating at a higher high frame than the cameras, the detected position will lag behind which causes the motion to be delayed and possibly become choppy. In this case, we apply motion prediction to fill up the positions in the frames where there is no detected position available. Prediction is done by using the previous position and orientation combined with velocity to extrapolate position and orientation. Note that velocity is measured independently from the captured images, which allows us to use this quantity even if no images are available.

Velocity and angular velocity can both be retrieved in two ways, if we use the Wii Remote together with the light orbs.

The first option is to deduct them from the retrieved values of the Wii MotionPlus. Alternatively, we can calculate them using data from the previous frame. It is also possible to combine both approaches, which in practice gives the best result.

Our system uses prediction for all frames. This way, whenever a detection is available, the prediction and the detection are combined and averaged, and used as the final result for a frame. The averaging is needed to make the motion smoother, especially when the accuracy of the prediction was low. For frames in which there is no detection available, only the prediction is used.

5 Results

We performed our tests in a cubical area of $352 \times 352 \times 262cm$ ($l \times w \times h$). In every upper corner a webcam is placed that is focused at the center of the area in horizontal direction and at a height of $125cm$ from the ground in vertical direction. The range the orb can be detected in ranges from $20cm$ to $200cm$ in vertical direction (relative from the ground) and $-80cm$ to $80cm$ in horizontal direction (relative to the focus point of the cameras). The precision of the detection is averaged to $0.5cm$ in any direction. The visible delay is approximately $500ms$: position detection in our system is smooth, but when moving fast, the delay is clearly visible.

The orientation detection while using only one orb is also smooth, but the yaw tends to go out of sync when making wild movements. Adding a second orb solved this issue. After this modification, angular precision is 0:6*deg*.

The total cost of the system is €185,- (see Table 1 for detailed information), excluding the projection screens and beamers.

Table 1. Test setup. Apart from these hardware components, we used 3rd party (C++) libraries to communication with the Wii Remote, for image processing and for multithreading.

Amount	Description	Cost (€)	Total (€)
4	30Hz webcam	15	60
2	Light orb (glowing ball from Playstation Move, and some custom hardware)	46	92
1	Wii Remote + Motion Plus extension	25	25
1	Bluetooth dongle	8	8

6 Conclusion and Future Work

We presented a system that combines Wii Remotes with methods used by the PlayStation Move system to determine the orientation and position in any direction surrounding the player. Our system uses two light emissive objects and a minimum of two webcams. The webcams are used to find the 3D position of the emissive objects.

Using a second emissive object allows us to accurately determine the direction of the device and deduct an absolute yaw and pitch. The Wii Remote allows us to to calculate the roll and determine the final orientation.

To improve the object detection speed, the current webcams could be replaced by webcams having a higher frame rate. This will reduce the number of frames for which we need to rely on prediction, resulting in more accurate and smooth results. Using higher resolution webcams could improve precision as well. This will particularly improve the detection of small movements.

The position and orientation prediction is currently partly done by extrapolation. This can be improved by using smarter algorithms that recognize movements.

For better object tracking the HSV color (instead of RGB) scheme could be used. This will improve the object detection in more challenging lighting conditions.

Acknowledgments. Jimmy van den Heuvel created the light orb (glowing ball tool). We used the WiiYourself!_1.15 [25] library to be able to communicate with the Wii Remote. OpenCV2.1.0 [4] library was used for processing captured images. Nils Deslé created the (multithreaded) job system.

References

1. ASCENSION. laserBIRD2 system (2011), `http://www.ascensiontech.com`
2. Azuma, R.T.: Predictive tracking for augmented reality. PhD thesis, Chapel Hill, NC, USA. UMI Order No. GAX95-38370 (1995)
3. The CODA system, `http://www.bleng.com`
4. Bradski, G.: Opencv2.1.0 (2010)
5. Castaneda, K.: Nintendo and pixart team up (2006)

6. Derhgawen, A.: Real-time color based object tracking (2007)
7. Fenlon, W.: How playstation move works (2010)
8. Foxlin, E., Harrington, M., Pfeifer, G.: Constellation: A wide-range wireless motiontracking system for augmented reality and virtual set applications (1998)
9. Foxlin, E., Altshuler, Y., Naimark, L., Harrington, M.: Flighttracker: A novel optical/inertial tracker for cockpit enhanced vision. In: Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality, ISMAR 2004, pp. 212–221. IEEE Computer Society, Washington, DC (2004)
10. Fraga, E.S.: Symmetric multiprocessing algorithm for conceptual process design (2000)
11. Greenwald, W.: Nintendo wii remote vs playstation move: How do they work? (2010)
12. Hruschack, P.J.: Wiimote vs. move vs. kinect: Comparing control schemes in the three-way battle for motion control (2010)
13. IGT. Flashpoint 5500 system (2011), http://www.imageguided.com
14. INTERSENSE. Intersense corporation (2011), http://www.isense.com
15. Kadaba, M.P., Stine, R.: Real-time movement analysis techniques and concepts for the new millennium in sports medicine (2000), http://www.motionanalysis.com
16. Kalman, R.: A new approach to linear filtering and prediction problems. Transactions of the ASME-Journal of Basic Engineering 82(series D), 35–45 (1960)
17. Kuipers, J.: Object tracking and orientation determination means, system and process. In: U.S. Patent 3 868 565 (1975)
18. Luinge, H.J., Veltink, P.H., Baten, C.T.M.: Estimating orientation with gyroscopes and accelerometers. Technol. Health Care 7, 455–459 (1999)
19. Mikhailov, A.: Playstation move tech interview (2010)
20. Miller, P.: Playstation move: everything you ever wanted to know (2010), http://www.engadget.com/2010/03/11/playstation-moveeverything-you-ever-wanted-to-know
21. MotionAnalysis, Hires 3d motion capture system. MotionAnalysis Corporation (2011)
22. Paperno, E., Sasada, I., Leonovich, E.: A new method for magnetic position and orientation tracking. IEEE Transactions on Magnetics 37, 1938–1940 (2001)
23. Raab, F.H., Blood, E.B., Steiner, T.O., Jones, H.R.: Magnetic position and orientation tracking system. IEEE Transactions on Aerospace and Electronic Systems AES 15, 709–718 (1979)
24. Sathrum, L.R.: Using a nintendo Wii remote to help navigate a robot (2010)
25. Saugnier, N.: WiiYourself! v1.15 (2010)
26. SCE. Playstation®move motion controller for playstation 3. Tech. rep. (2010)
27. Sharlin, E., Figueroa, P., Green, M., Watson, B.: A wireless, inexpensive optical tracker for the CAVE(tm). In: Proceedings of the IEEE Virtual Reality 2000 Conference, VR 2000, pp. 271–278. IEEE Computer Society, Washington, DC (2000)
28. Sutherland, I.E.: A head-mounted three dimensional display. In: Proceedings of the 1968 Fall Joint Computer Conference, vol. 33, pp. 757–764 (1968)
29. Defanti, T.A., Sandin, D.: A 'room' with a 'view'. In: IEEE Spectrum, pp. 30–33 (1993)
30. Welch, G.F.: History: The use of the kalman filter for human motion tracking in virtual reality. Presence: Teleoper. Virtual Environ. 18, 72–91 (2009)
31. Williamson, B.M.: Realnav: Exploring natural user interfaces for locomotion in video games (2010)
32. Wingrave, C.A., Williamson, B., Varcholik, P., Rose, J., Miller, A., Charnonneau, E., Bott, J.: Spatially convenient devices for 3D user interfaces (2010)