EGVE Symposium (2008)
B. Mohler and R. van Liere (Editors)

Camera Calibration of a Nintendo Wii Remote using PA-10 Robotic Arms

H. S. Grimes[†][1], R. S. Ferguson[1] and K. McMenemy[1]

[1]Intelligent Systems and Control Research Group, Queen's University Belfast, U.K.

Abstract

The Remote control unit for the Nintendo Wii computer games console features an infrared camera capable of detecting up to 4 infrared lights. As such, it is possible to calibrate this device using normal camera calibration techniques so that it can be used for tracking or human interfacing in a virtual environment. Camera calibration is an active area of research and there is a wealth of software available to perform the calibration e.g. ArToolKit and Matlab. Camera calibration typically requires at least five images which contain multiple grid points from which a matrix of camera parameters can be estimated. This paper proposes a method of calibrating the Wii Remote's IR camera by building up 24 calibration points from the 4 viewed infrared LEDs for a single viewpoint of the IR camera. This is done by moving the 4 LEDs in a known sequence using highly accurate PA-10 robotic arms. The camera calibration matrix parameters obtained from this method are presented.

Categories and Subject Descriptors (according to ACM CCS): I.3.4 [Computer Graphics]: Virtual device interfaces; I.3.7 [Computer Graphics]: Virtual reality; I.3.8 [Computer Graphics]: Applications; I.4.1 [Image Processing and Computer Vision]: Camera calibration; I.4.8 [Image Processing and Computer Vision]: Tracking;

1. Introduction

The Nintendo Wii Remote (Wiimote) is a piece of wireless technology which encompasses a three axis accelerometer, eleven useable buttons, vibration feedback, a speaker and an optical sensor that detects infrared (IR) sources. Since its launch in November 2006, the Wiimote has provided many people with a unique source of interaction with their games. Due to the Wiimote using bluetooth, a connection to the device from a Personal Computer (PC) was quickly deciphered. Applications of the Wiimote with a PC include being able to create a cost-effective whiteboard [Lee08].

Many applications can benefit from the Wiimote components, e.g. the accelerometers have been used in a virtual environment for rehabilitation of an individual's upper extremity after a stroke [BG07]. Connection to the Wiimote can be done via applications which emulate it as a human interface device (HID) controller [Ken08], as a background program [Wii07a] or as C/C++ libraries [Wii07b] which can be inserted into one's program.

After the Nintendo Wii's release, information regarding the IR image sensor on the Wiimote was limited. Since then it has been discovered that it is a widely available optical camera which detects and returns information on up to four IR sources, such as the size of a detected IR light source and its associated coordinates. The camera resolution is 1024 pixels by 768 pixels. The frame rate of the camera is not accurately known, but information from the IR sensor can be obtained every 10 ms.

This paper presents a calibration matrix for the Wiimote IR camera that can be used to provide positional information and aid in using the Wiimote as a tracking device. The method used to obtain the camera's calibration matrix, which uses a pair of industrial robots to allow for high accuracy, is also described. For the Wiimote to be used as a tracking device, it is first necessary to determine if small movements can be successfully detected by the Wiimote. These preliminary tests are outlined in section 2.

† Email: sgrimes01@qub.ac.uk

2. Camera sensitivity and accuracy trials

A number of sensitivity and accuracy trials were performed initially on the Wiimote to determine the feasibility of using it to track movement within a room for VR applications.

2.1. Camera sensitivity setup

Several tests were carried out using a robotic arm to determine the sensitivity and accuracy of the Wiimote. These involved monitoring the coordinates returned from a single IR LED while the Wiimote is moved a set amount. The tests moved the Wiimote horizontally and vertically by 5mm graduations and rotated the Wiimote by 2 degrees horizontally, as shown in figure 1. The Wiimote is attached to a robotic arm which is moved and rotated. More details on the robotic arm are given in section 3.2. It must be noted that the pixel difference for movement will change depending on the distance from the Wiimote to the LEDs. The horizontal and vertical movements were performed at a distance of approximately 129cm from the LEDs, and the rotational movement was performed at a distance of 180cm from the LEDs.

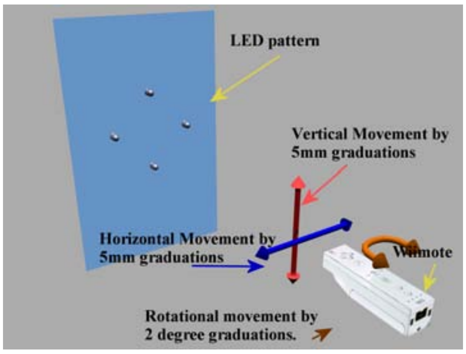

Figure 1: *Image showing the trials of the Wiimote for the sensitivity tests, highlighting the horizontal, vertical and rotational movements. Not shown is the Robotic arm attached to the Wiimote.*

2.2. Results and discussion

Results show that when the Wiimote was moved vertically by 5mm increments, it could accurately determine each 5mm step, as shown in figure 2. For each 5mm movement, the returned coordinate of an IR LED from the Wiimote changed on average by 4 pixels.

When the Wiimote moves horizontally by 5mm increments, the results show that the Wiimote can accurately determine each 5mm step, as shown in figure 3. For each 5mm movement, the returned coordinate of an IR LED from the Wiimote changed on average by 5 pixels.

With horizontal rotation of the Wiimote by 2 degree graduations, the results in figure 4 show that the Wiimote can detect each step with a large degree of accuracy. For each 2 degree movement, the coordinate of an IR LED detected changed on average by 48.33 pixels.

Results from the trials demonstrates that the Wiimote could be used for position and orientation sensing. Movement vertically or horizontally at a distance of 129cm provided a change of almost 1:1 pixel to mm ratio. Movement caused by minute shaking of a person due to muscle contractions or heartbeats would be visible in the results. For any camera, rotation will have a greater effect on the coordinate change returned of a tracked point, especially at a decreased distance between the camera and the object, which can be seen in the results. In practice, the software would have to be made to use more than a single LED pattern for tracking rotation, and the distance between the Wiimote and the IR LEDs increased to 2-5m for comfortable tracking. These trials were used to simply determine if the Wiimote could track small movements. Since this is possible, if exact locations in 3D space are to be returned using the Wiimote, then it is necessary to calibrate the IR camera. The procedures to do this are outlined in section 4 after a short review into the theory of camera calibration in section 3.

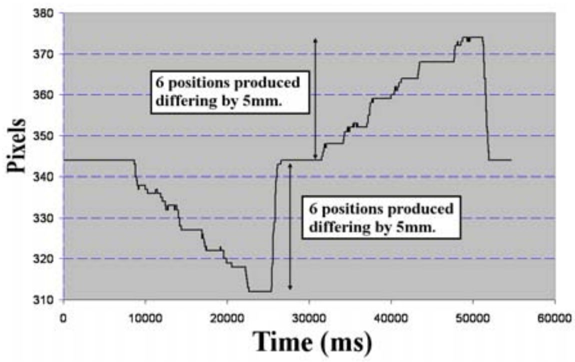

Figure 2: *Results for vertical movement of the Wiimote by 5mm increments in pixels.*

3. Camera calibration

Camera calibration is a required step for applications where knowledge of the transformations are required to accurately map a 3-dimensional (3D) coordinate to a point in the camera's 2-dimensional (2D) plane or when determining the position of the camera. Augmented Reality (AR) and computer vision make use of these transformations for inserting 3D objects seamlessly into a virtual scene. Many applications use these for highlighting and tracking objects in an image for specific purposes; an example is tracking humans in surveillance [VC00].

Camera calibration is a method for estimating the parameters of a camera. These include the intrinsic properties such as the distortions in the lens of a camera, the focal length, the

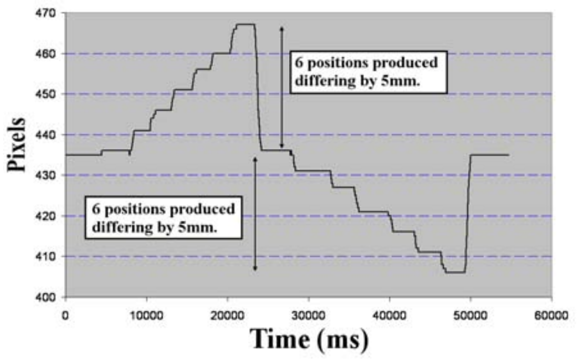

Figure 3: *Results for horizontal movement of the Wiimote by 5mm increments in pixels.*

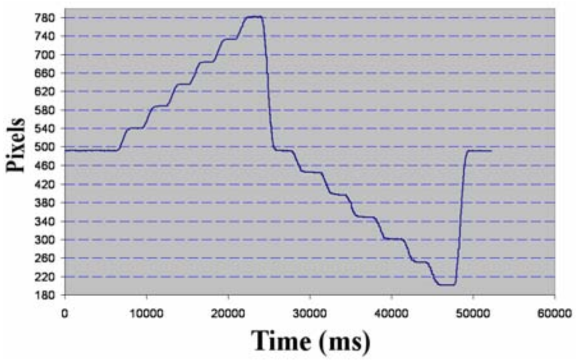

Figure 4: *Results for horizontal rotation of the Wiimote by 2 degree increments in pixels.*

skew and other internal properties of the camera. These parameters can be found using different methods. One method is using a 1-dimensional (1D) calibration object, which is observing 3 points of reference a fixed distance apart with one point fixed in position [Zha04]. Another is to take at least 3 images of a static scene from a moving camera and use corresponding points of interest in the images which do not move(considered self-calibration or 0-dimensional (0D) based calibration as no calibration object is used). Calibration using a 3D reference object is also possible by using an object which is accurately measured. Typically most calibrations are performed using a 2D calibration pattern that is viewed from different angles and orientations of the camera. This calibration setup is easy to create and so is the most commonly used method. Examples of commonly available calibration software based upon this method include ArToolKit, the Camera Calibration Toolbox for Matlab [KS07] and Z. Zhang's flexible Microsoft Easy Camera Calibration Tool [Zha99].

3.1. Wiimote IR sensor

The IR sensor on the Wiimote can be used as a cost-effective low-light marker tracking device in applications such as software which use ArToolKit. For this to occur more information about the sensor is required, such as its sensitivity, accuracy, distortion factors and the calibration matrix. The latter is difficult to retrieve on this particular camera as it can only return a maximum of four points of interest (IR sources as seen by the camera). Camera calibration requires as many point correspondences as possible to accurately create the equations needed to estimate the coefficients in the matrix. A minimum of eleven equations are needed to solve the matrix and since each point correspondence creates 2 equations, only five and a half point correspondences are needed (only one equation of the sixth correspondence is needed) [KM07] but usually more corresponding points are needed to retrieve an accurate matrix. For example, the ArToolKit calibration procedure requires at least 5 images of a pattern containing 24 points to obtain an accurate solution [Kat07].

If ArToolKit, or any other commonly available software, is to be used to determine the camera parameters of the Wimote, then it is necessary to increase the 4 points of interest obtained from the Wiimote. One method of creating the 24 positions needed for ArToolKit with the Wiimote is to extrapolate from the 4 real points as the corners of the pattern. This however can create inaccuracies as distortion factors are overlooked.

In calibration software which uses images taken of a scene from different views (2D calibration), it is often required that some or all of the points of interest are manually selected, which can also introduce human error. The method of calibration proposed here eliminates human error and correctly simulates a view of 24 points as needed by ArToolKit.

The calibration procedure for a Wiimote only needs to be performed once to get an accurate calibration matrix. Due to manufacturing standards each Wiimote uses the same camera part set to the same specification, so an accurate calibration matrix for one Wiimote should work with other Wimotes.

Robotic arms can be set up to create an accurate calibration grid to be used in calibration techniques such as ArToolKit or the Matlab Calibration Toolbox. This is described in section 4.1 for the ArToolKit one-step calibration procedure.

3.2. Robotic arms

In order to calibrate the Wiimote, a method of accurately moving the Wiimote and the LED pattern, which consists of 4 IR sources, to collect 24 points for a single image is needed. The calibration is performed using 2 robotic arms facing each other with the Wiimote mounted on one and the IR LEDs mounted on the other arm. Figure 5 shows this setup. The PA-10 robotic arms [Mit07] have 7 degrees-of-freedom and are controlled from a computer. The robots can be moved in two ways; either by specifying movement of the tool tip relative to the robot's base position or by setting the individual joint angles of the robot arm.

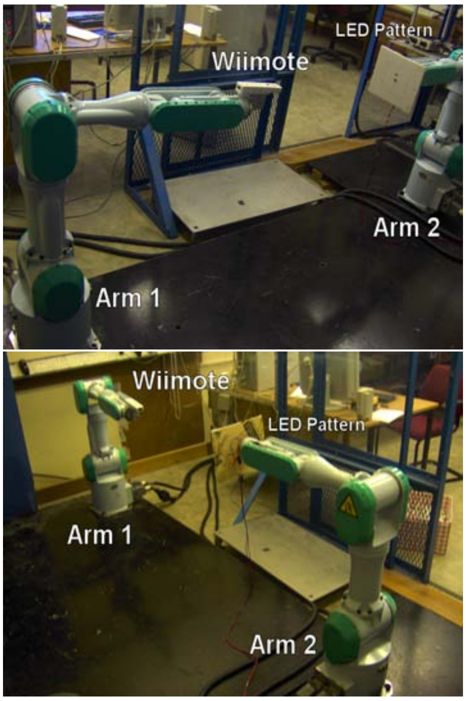

Figure 5: *Photographs of the robotic arms setup.*

The program used for the calibration and accuracy tests change the angle of each joint as well as moving the arm controlling the pattern by a set distance. The computer used to drive the robotic arms, ArToolKit's modified calibration software and the Wiimote interface libraries, is a 1.8GHz desktop with 256MB of RAM. This has a Bluetooth adaptor and runs Bluesoleil in order to connect the Wiimote as a HID.

4. ArToolKit

The authors proposed using the ArToolKit software to determine the calibration matrix. ArToolKit is a C library used for building AR applications [Kat07]. Many sample applications of ArToolKit are primarily based on overlaying virtual elements on the real world as seen through a camera by tracking markers. A typical ArToolKit application will track a visual marker by first taking an image from the camera, applying a distortion to correct the image according to the calibration and then search for the visual marker and use this information for a task. This can be an AR conferencing system [HK99] or even tracking multiple markers for motion capture [SLRB04].

The toolkit includes the source and applications to calibrate a camera using either a one-step method utilizing a pattern of 24 dots, or a two-step method elaborating on the first method and calculating the properties using parallel lines in a set pattern. The one-step calibration is described in section 4.1.

4.1. One-step calibration

The first method is a one-step calibration which utilizes a printout of 24 calibration points as seen in figure 6. An image is then taken using a camera of this calibration grid. The user manually highlights each point in the grid so that the calibration software can identify corresponding points across time, in the order shown in figure 7a. The user is prompted to take more than two photos of the points from different angles and to identify the dots in the pattern each time. It is recommended that this procedure is repeated ten times in order to collect calibration points from ten different viewpoints. Once all the images are taken, the toolkit calculates the distortion and projective matrix.

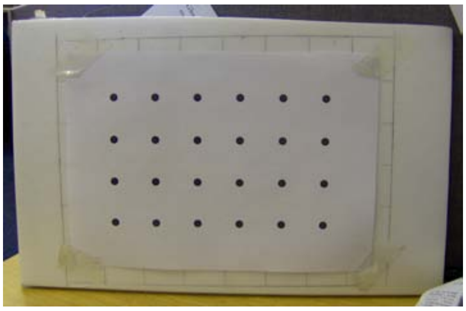

Figure 6: *Photograph of the 24 dot calibration pattern for ArToolKit.*

ArToolKit calculates the matrix by using the knowledge of the distance between each dot and the lines of best fit between the dots. The resulting matrix is stored in a suitable format that ArToolKit applications can read from. The matrix values obtained from the one-step calibration are shown in table 1 in the format displayed by equation 1 where α_x is the scale factor in the X axis, α_y is the scale factor in the Y axis, s_k is the skew and (x_0, y_0) is the coordinate of the principle point [Sha92]. The scaling factors are used to compute the focal length in the X and Y directions, for placing an AR image over a real image and for determining the position of the camera relative to the marker. They are in the format of pixels per unit millimeter and encompass the focal length of the camera. The skew factor is used to describe the skew between the horizontal and vertical axis of a camera image. A camera with a high skew factor would show an image in the shape of a parallelogram where a camera with no skew would show the image in the shape of a square or rectangle depending on the resolution and aspect ratio. Typically the camera projective matrix is a 3×3 matrix, but ArToolKit stores this in a 4×3 matrix with an augmented 4th column of 0.

$$\textit{Camera matrix } P = \begin{bmatrix} \alpha_x & s_k & x_0 & 0 \\ 0 & \alpha_y & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (1)$$

4.1.1. One-step calibration setup

To calibrate the Wiimote IR camera using the one-step method, robotic arms were setup up as in figure 5. Different positions of arm 1 allowed the Wiimote to detect a calibration pattern containing 4 LEDs from different viewpoints. Moving arm 2 created the pattern of 24 LEDs from 4 LEDs.

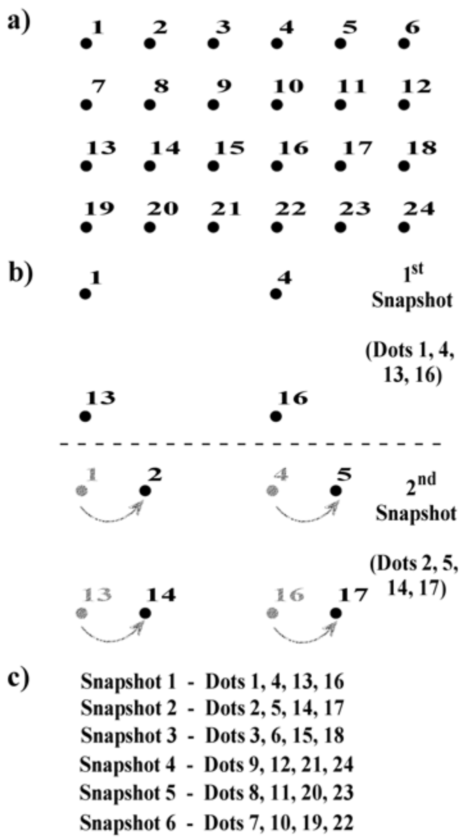

Figure 7: *a) 24 dots needed for calibration using ArToolKit. b) First 2 snapshots the Wiimote takes showing which dots are active. c) List of which dots is active for each snapshot.*

The Wiimote took a snapshot of the 4 LEDs and recorded their coordinates in the list of 24 calibration points. The first 4 calibration points recorded were dots 1, 4, 13 and 16. Arm 2 was then moved to the next position so that the next 4 dots the Wiimote detected were 2, 5, 14 and 17 as shown in Figure 7b and the coordinates recorded. The arm was moved in the pattern shown in figure 7c so that all 24 calibration points were recorded in a list, ready to be sent onto ArToolKit for calibration. Therefore there is no manual interaction in selecting any of the calibrations points.

Once completed, the position of the second robotic arm was reset to the first position of the dot pattern and arm 1 moved to a new viewpoint. This procedure was repeated for several viewpoints; gathering 24 calibration points from the movement of the LEDs for a specific viewpoint. The images the Wiimote detected are presented in figures 8 and 9, taken from the intermediate stages of the ArToolKits calibration.

The software which controls this procedure was written in C++ and run on the desktop described in section 3.2. This program encompasses the PA-10 libraries for controlling the PA-10 robotic arms, ArToolKit's calibration libraries, the expanded C++ library files to communicate with the Wiimote and the code necessary to build 24 calibration points using the robotic arms. The original ArToolKit's calibration functions were used for calculating the matrix parameters.

4.1.2. One-step calibration results and discussion

The setup for the Wiimote calibration ran successfully, and returned images from ArToolKit as shown in figures 8 and 9. The 24 calibration points can be seen with the lines of best fit which ArToolKit added to find the calibration matrix for the one-step method. Figure 8 shows the first viewpoint of calibration points detected and figure 9 show another viewpoint used during the calibration. A total of 10 viewpoints were taken to improve accuracy of the resulting calibration matrix for the 1-step method using ArToolKit and three different working Wiimotes were used to provide robustness.

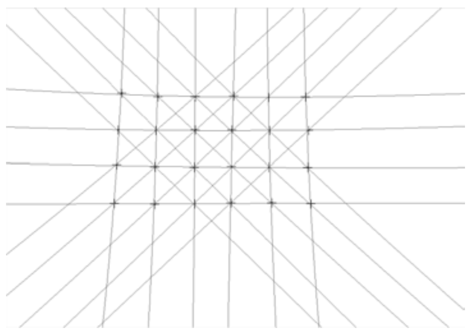

Figure 8: *Calibration screen 1.*

The matrix results are shown in table 1 for the one-step method. These results proved to be too unstable as the scaling factors varied wildly for each Wiimote and for each repetition of a single Wiimote. The same occurred with a normal camera's one-step calibration which was tested. Since the results did not show the expected similarity between Wiimote calibration matrices, it was decided to try the calibration using the two-step technique, which is known to be more ac-

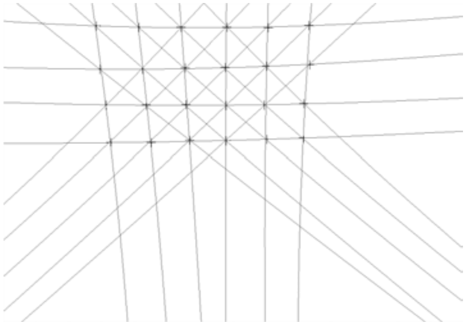

Figure 9: *Calibration screen 2.*

Wiimote	α_x	α_y	s_k	x_0	y_0
1	5408.1	2958.3	0	407	389
1	5468.0	2269.2	0	410	389
2	4868.3	2070.7	0	407	389
3	3924.4	1666.6	0	483	389

Table 1: *Matrix results using the ArToolKit one-step method*

curate. Calibration using this method is described in section 4.2.

4.2. ArToolkit's two-step calibration method

ArToolKit's two-step calibration uses a pattern of 16 lines, 9 vertical lines and 7 horizontal lines, a set distance apart. The user must move this pattern away from the camera by 100mm, find each line manually, and repeat this 5 times. This method is more accurate but less efficient with regards to time compared to the one-step method. A full description of the patterns used and what is involved in calibrating cameras using the ArToolKit can be found in [Kat07]. Calibrating using ArToolKit's two-step method will provide more accurate matrix results for any camera compared with the one-step method.

4.2.1. Two-step calibration setup

There is difficulty in using this method normally for the Wiimote as it requires manipulation of lines to fit the pattern. Using 2 dots to indicate a line and moving a robotic arm to create the positions of parallel lines will allow calibration. First a line is drawn on the screen using the calibration software between the 2 visible points from the Wiimote indicating a line from the pattern. The LEDs are moved to the next position necessary for the software using the robotic arm, and a new line is drawn as described in figure 10. After the 16 lines are found, the second robotic arm moves the Wiimote and the process is repeated.

This method also outputs a value for the skew factor, s_k. This value is often edited afterwards by developers to 0.0 if

Wiimote	α_x	α_y	s_k	x_0	y_0
1	1441.2	1437.9	6.5	558	359

Table 2: *Matrix results using the ArToolKit two-step method*

the camera is known to have no skew to correct the matrix for use.

The software for this method is based on the same procedures used for the one-step software except that only 4 feature points are displayed and the selection and calibration sections were taken directly from ArToolKit's second step calibration software.

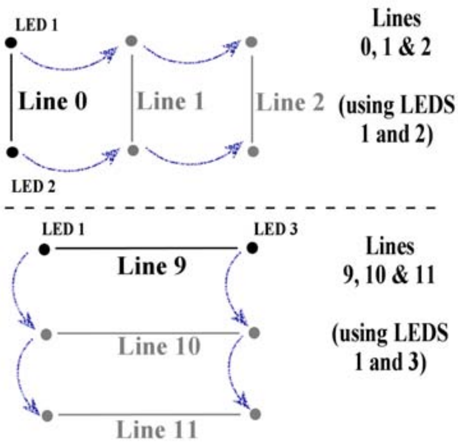

Figure 10: *The 16 lines needed by ArToolKit's two-step calibration are created by aligning the lines over where the LEDs are currently placed. LEDs 1 and 2 are used to create the vertical lines 0-8 and LEDs 1 and 3 are used to create the horizontal lines 9-15.*

Results of this calibration method obtained are shown in section 4.2.2.

4.2.2. Two-step calibration results and discussion

The results from the two-step ArToolKit matrix calibration yielded much lower scaling factors and different principal points than the one-step method. The coordinates of the principal point are (558, 359). Developers of AR applications tend to change the principal point from what the calibrations give to the center coordinates of the images returned. The scaling factors are shown to be (1441.2, 1437.9) for the Wiimote used.

The two-step method also gave a skew factor of 6.5 units. There was no value for skew output from the one-step calibrations. Developers tend to overwrite any skew value to 0.0 unless the camera is known to be skewed, so this result can be ignored for practical purposes.

Tests using ArToolKit's exview application, which returns the position of the camera relative to the marker or vice versa, with the Wiimote or a normal camera's two-step matrix gave the position accurately to within 2cm of its actual location from the marker (after taking out the skew and realigning the principal points to the center).

However there is only one matrix obtained for the two-step method, and the results aren't consistent with the results from the one-step method so a more rigorous, scientific technique is required to compare to the two-step calibration matrix.

5. Matlab camera calibration toolbox

The Matlab Calibration Toolbox is a standard calibration software built and executed in Matlab using sets of images taken from a camera. The toolbox extracts feature points from a visible pattern in an image and using these coordinates and the known attributes of the physical pattern, a calibration matrix is extracted from a series of images. The matrix output is in the same format shown in equation 1.

5.1. Matlab camera calibration toolbox setup

The toolbox requires sets of feature points extracted from images of an accurately known pattern. The sets of 24 calibration points used in the one-step calibration were exported to text files and imported into the toolbox for calibration of each Wiimote. This method uses the same functions as in the one-step method, except that the data isn't passed to ArToolKit's calibration functions but to an external text file. The only difference between calibration of a camera and a Wiimote in Matlab is that feature points first need to be extracted for a camera image whereas the Wiimote already outputs the feature points without software extraction from an image first. The Wiimote has a onboard PixArt chip which processes the camera images and forwards on the minimum information of the retrieved IR feature points to the system, so feature extraction using Matlab is unnecessary.

20 viewpoints were taken for the Matlab calibration method using the same pattern used in the one-step method described in section 4.1.1.

5.2. Matlab camera calibration results and discussion

The results from the Matlab Toolbox method, displayed in table 3, showed a strong similarity between most of the parameters of the Wiimotes. Deviations between the results of the scaling factors (α_x and α_y) were between 18.8 and 108.6 units across the three Wiimotes. This seems to indicate that the devices are quite different, but α_x and α_y are the products of the internal scaling factors and the focal length ($\alpha_x = \mathbf{f} \times \mathbf{k}_x$ [KM07]), and with the focal length estimated for the Wiimotes as 25.0mm in the Toolbox, then the internal scaling factors could only be different by maximum 4.344 pixels per unit millimeter.

Wiimote	α_x	α_y	$\mathbf{s}_k$	$\mathbf{x}_0$	$\mathbf{y}_0$
1	1290.2	1277.4	0	500	408
2	1309.0	1318.4	0	529	335
3	1374.6	1386.0	0	536	332

Table 3: *Matrix results using the Matlab Toolbox calibration method*

6. Conclusions

For applications to be able to use a Wiimote IR camera it is essential to calibrate it, this is difficult because the Wiimote can only track 4 IR points at any time. This paper presents a calibration method using PA-10 robots that delivered accurate calibration parameters for the Wiimote's IR camera that are appropriate for any device.

Calibration without using the PA-10 robots is possible but less accurate. To do this, the 4 LED pattern will have to be manually moved the set distance between the feature points of the 24 dot pattern. Snapshots of each movement with the Wiimote can be used to retrieve 24 feature points per viewpoint. This can be used in ArToolKit's one- or two-step calibration programs or an external program like the Matlab ToolBox.

The results from the ArToolKit one-step calibration method showed too much variance in the matrices retrieved and are unsuitable for use in applications. The two-step results seem more accurate but cannot be verified until additional Wiimotes are calibrated using this method. The parameters calculated by the Matlab Toolbox seem consistent with each other and comparable to the two-step method, proving the Matlab Toolbox to be the most robust of the calibration methods.

Future work include adapting the ArToolKit code to allow the Wiimote to be used for general tracking applications and developing procedures for position and orientation sensing for a head mounted display system. The benefit of using a Wiimote is that IR LED marker patterns are inconspicuous and the device provides a cost-effective wireless solution to accurate tracking and multiple user interaction in a CAVE.

References

[BG07] B. Galego L. S.: Leveraging online virtual worlds for upper extremity rehabilitation. *Bioengineering Conference, 2007. NEBC '07. IEEE 33rd Annual Northeast* (10-11 March 2007), 267–268.

[HK99] Hirokazu Kato M. B.: Marker tracking and hmd calibration for a video-based augmented reality conferencing system. *iwar 00* (1999), 85.

[Kat07] Kato H.:, 2007. (ArToolKit Software Library. Active: 18th January 2008.) http://www.hitl.washington.edu/artoolkit/.

[Ken08] KENNER C.:, 2008. (Glove Programmable Input Emulator Software Package 0.30. Active: 18th January 2008.) http://carl.kenner.googlepages.com/glovepie/.

[KM07] KAREN MCMENEMY S. F.: *A Hitchhikers Guide to Virtual Reality*. A. K. Peters, Ltd., Natick, MA, USA, 2007.

[KS07] K. STROBL W. SEPP S. F. C. P. K. A.: Callab: Camera calibration toolbox for matlab, 2007. (Institute of Robotics and Mechatronics. Active: 18th January 2008.) http://www.vision.caltech.edu/bouguetj/calib_doc/index.html.

[Lee08] LEE J. C.:, 2008. (Wii Projects. Active: 18th January 2008.) http://www.cs.cmu.edu/~johnny/projects/wii/.

[Mit07] MITSUBISHI:, 2007. (PA10 Robot. Active: 18th January 2008.) http://sanki-mhi.com/RJengHPe/products/pa10.html.

[Sha92] SHAH M.: *Fundamentals of Computer Vision*. December 1992. (Online book download. Active: 18th January 2008.) http://www.cs.ucf.edu/courses/cap6411/book.pdf.

[SLRB04] SEMENTILLE A. C., LOURENCO L. E., RODELLO I., BREGA J. R. F.: A motion capture system using passive markers. pp. 440 – 447.

[VC00] V. CHENG N. K.: A smart camera application: Dsp-based people detection and tracking. *Journal of Electronic Imaging 9*, 3 (2000), 336–346.

[Wii07a] 2007. (Wiinmote Software. Active: 18th January 2008.) http://onakasuita.org/wii/index-e.html.

[Wii07b] 2007. (Wiim C++ API libraries. Active: 18th January 2008.) http://digitalretrograde.com/projects/wiim/.

[Zha99] ZHANG Z.: Flexible camera calibration by viewing a plane from unknown orientations. *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on 1* (1999), 666–673 vol.1.

[Zha04] ZHANG Z.: Camera calibration with one-dimensional objects. *IEEE Trans. Pattern Anal. Mach. Intell. 26*, 7 (2004), 892–899.