A Study of Microsoft Kinect Calibration

Xing Zhou
Dept. of Computer Science
George Mason University
xzhou10@gmu.edu

September 11, 2012

Abstract

Microsoft Kinect has been widely used by researchers since its first release in 2010 due to its consumable price compared with other ranger sensors. Calibration is the first step if we are willing to use it. In this study, besides the calibration methods, a detailed description of Kinect's working principles as well as its calibration parameters are given. What's more, the process to align depth image with color image after calibration is described step by step. Finally the calibration result and a short discussion are presented.

Keywords: Kinect, Calibration, IR pattern

1 Introduction

Kinect is a motion sensing input device by Microsoft for the Xbox 360 video game console and Windows PCs[1]. Based around a webcam-style add-on peripheral for the Xbox 360 console, it enables users to control and interact with the Xbox 360 without the need to touch a game controller, through a natural user interface using gestures and spoken commands. Since it was launched in North America on November 4, 2010, it has become more and more popular in research communities especially in the areas of robotics and computer vision due to its output depth information as well as a consumable cost.

The Kinect device has two cameras and one laser-based IR projector as shown in Figure 1. Each lens is associated with a camera or a projector. Kinect sensor outputs video at a frame rate of 30 Hz. The RGB video stream uses 8-bit VGA resolution (640 × 480 pixels) with a Bayer color filter, while the monochrome depth sensing video stream is in VGA resolution (640 × 480 pixels) with 11-bit depth, which provides 2,048 levels of sensitivity.

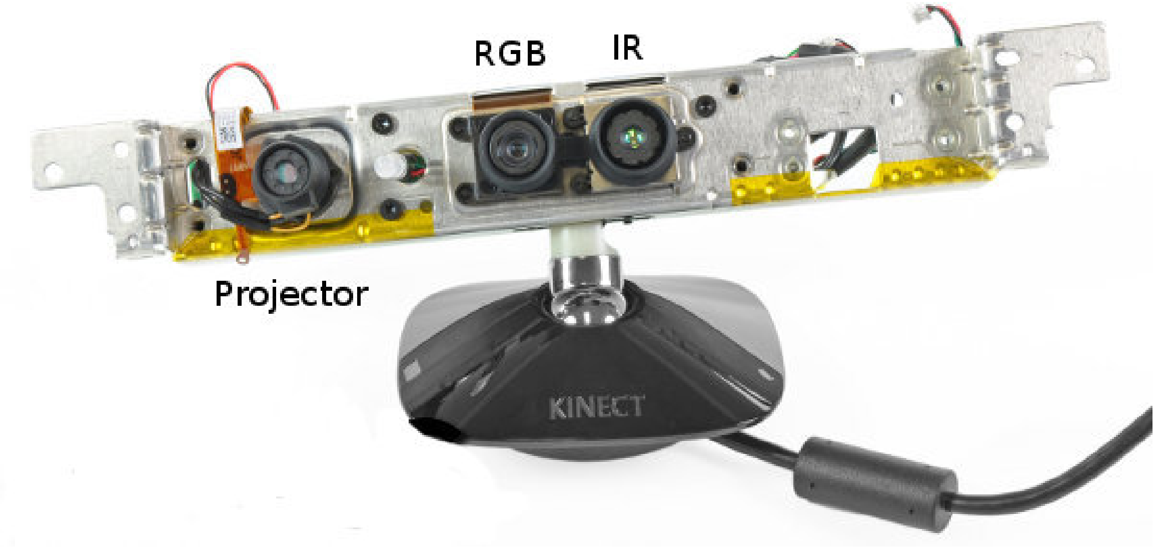

Figure 1: Microsoft Kinect Sensor: *http://www.ros.org/wiki/kinect_calibration/technical*

2 How it Works?

2.1 Principle behind Kinect[2][3]

Figure 2 is a schematic top view of a speckle imaging device. An illumination assembly *30* comprises a coherent light source *32*, typically a laser, and a diffuser *33*. (The term "light" refers to any sort of optical radiation, including infrared and ultraviolet, as well as visible light.) The beam of light emitted by source*32* passes through diffuser *33* at a spot *34* of a radius w_0, and thus generate a diverging beam *36*. Specifically, by triangulation in this arrangement, a Z-direction shift of a point on the object, δZ, will engender a concomitant transverse shift δX in the spot pattern observed in the image (Shown in Equation 1). Z-coordinates of the points on the object, as well as shifts in the Z-coordinates overtime, may thus be determined by measuring shifts in the X-coordinates of the spots in the images captured by assembly *32* relative to a reference image taken at a known distance Z. Y-direction shifts may be disregarded.

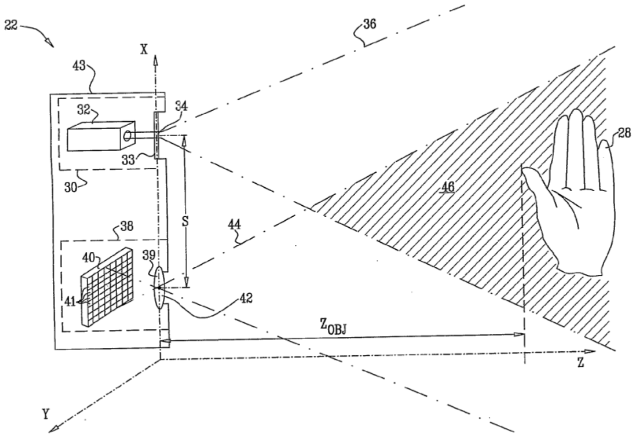

Figure 2: A schematic top view of a speckle imaging device

$$\delta X \approx \delta Z \frac{S}{Z_{obj}} \tag{1}$$

Before mapping an object, device *22* is calibrated by projecting the speckle pattern from assembly *30* onto an object of known spatial profile at a known distance from the device. Typically, a planar object extending across area *46* at a known distance Z_{obj} is used as a calibration target for this purpose. Image capture assembly *38* captures a reference image of the object, which is stored in a memory of processor *24*.

Thus, to generate the 3D map of object, image processor compares the group of spots in each area of the captured image to the reference image in order to find the most closely-matching group of spots in the reference image. The relative shift between the matching groups of spots in the image gives the Z-direction shift of the area of the captured image relative to the reference image. The shift in the spot pattern may be measured using image correlation or other image matching computation methods that are known in the art.

2.2 The way how Kinect works

The IR camera and the IR projector form a stereo pair with a baseline of approximately 7.5 cm. The IR projector sends out a fixed pattern of light and dark speckles.

Depth is calculated by triangulation against a known pattern from the projector. The pattern is memorized at a known depth. For a new image, we want to calculate the depth at each pixel. For each pixel in the IR image, a small correlation window (9×9 or 9×7) is used to compare the local pattern at that pixel with the memorized pattern at that pixel and 64 neighboring pixels in a horizontal window. The best match gives an offset from the known depth, in terms of pixels: this is called disparity. The Kinect device performs a further interpolation of the best match to get sub-pixel accuracy of 1/8 pixel. Given the known depth of the memorized plane, and the disparity, an estimated depth for each pixel can be calculated by triangulation.

3 Kinect Calibration Parameters[4]

Microsoft Kinect equips with two cameras (RGB color camera and IR camera) and one IR pattern projector. The goal of Kinect calibration is to get the following parameters (Table 1):

Table 1: Calibration Parameters and Descriptions

Parameters	Descriptions
$f_{RGB}, cx_{RGB}, cy_{RGB}$	Intrinsic Matrix of color camera
k_1, k_2, k_3, p_1, p_2	Distortion vector of color camera
f_{IR}, cx_{IR}, cy_{IR}	Intrinsic Matrix of IR camera
k_1, k_2, k_3, p_1, p_2	Distortion vector of IR camera
R, T	Extrinsic matrix between color and IR cameras [1]
b	Baseline between IR camera and IR projector
d_{off}	Depth offset

3.1 Disparity to depth relationship

For a normal stereo system, the cameras are calibrated so that the rectified images are parallel and have corresponding horizontal lines. In this case, the relationship between disparity and depth is given by:

$$Z = \frac{b \times f}{d} \tag{2}$$

where Z is the depth (in meters), b is the horizontal baseline between the cameras (in meters), f is the (common) focal length of the cameras (in pixels), and d is the disparity (in pixels). At zero disparity, the rays from each camera are parallel, and the depth is infinite. Larger values for the disparity mean shorter distances.

On the other hand, the Kinect returns a raw disparity that is not normalized in this way, that is, a zero Kinect disparity does not correspond to infinite distance. The Kinect disparity is related to a normalized disparity by the relation:

$$d = \frac{1}{8} \times (d_{off} - k_d) \tag{3}$$

where d is a normalized disparity, k_d is the Kinect disparity, and d_{off} is an offset value particular to a given Kinect device. The factor $\frac{1}{8}$ appears because the values of k_d are in $\frac{1}{8}$ pixel units.

3.2 Calculating baseline and disparity offset

3.2.1 Method Derivation

A monocular calibration of the IR camera finds the focal length, distortion parameters, and lens center of the camera. It also provides estimates for the 3D position of the chessboard target corners. From these, the measured projections of the corners in the IR image, and the corresponding raw disparity values, we do a least-square fit to Equation 4 to find b and d_{off}.

$$d = \frac{b \times f}{\frac{1}{8} \times (d_{off} - k_d)} \tag{4}$$

The value for b is always about $7.5cm$, which is consistent with the measured distance between the IR and projector lenses while d_{off} is typically around 1090.

3.2.2 ROS Result[5]

ROS argues that the Kinect's depth output appears to be linearly proportional to the inverse of the distance to the object (As shown in Figure 3). Depth calibration was determined experimentally, by measuring the reading of the center pixel in the depth image, and doing a regression on the data. The first attempts at calibration are

[1] $\mathbf{X}_d = R \times \mathbf{X}_{rgb} + T$

presented here and more accurate calibration has been proposed by some other researchers. For example, in the software RGBDemo developed by Nicolas Burrus he employs Levenberg Marquardt algorithm (LMA) to fit Equation 4.

3.3 IR camera to depth offset

The Kinect device can return the IR image, as well as a depth image created from the IR image (See Figure 4). There is a small, fixed offset between the two, which appears to be a consequence of the correlation window size. Looking at the raw disparity image below, there is a small black band, 8 pixels wide, on the right of the image.

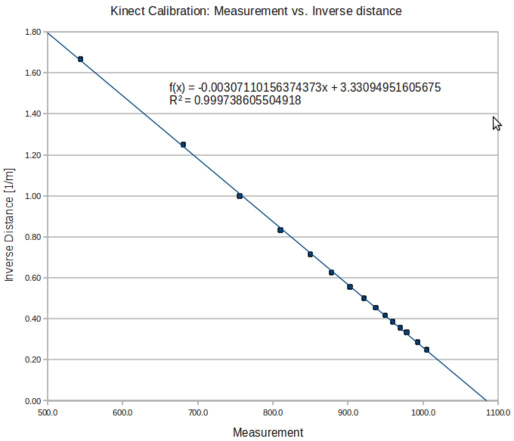

Figure 3: Measurement vs. Inverse distance

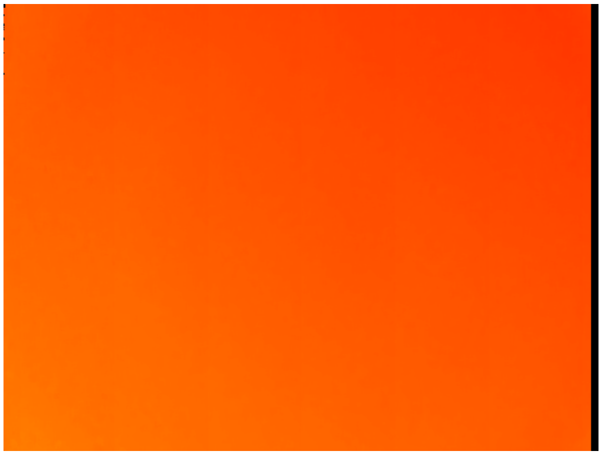

Figure 4: IR to Camera Offset

The null band is exactly what would be expected if the Kinect used a correlation window 9 pixels wide. To see this, assume a 9×9 correlation window; then, the first pixel that could have an associated depth would be at (5, 5) in the upper left corner. Similarly, at the right edge, the last pixel to get a depth would be at $N-(5,5)$, where N is the width of the image. Thus, there are a total of eight horizontal pixels at which depth cannot be calculated. The Kinect appears to send the raw disparity image starting at the first pixel calculated; hence, the offset in horizontal and vertical directions.

In the horizontal direction, the size of the correlation window is given by the null band size of 8 pixels. In the vertical direction, there is no such band. That's because the IR imager, an Aptina MT9M001, has resolution 1280×1024. If the Kinect uses 2×2 binning, then the reduced resolution is 640×512. The Kinect returns a 640×480 raw disparity image. There is no room on the imager in the horizontal direction to calculate more depth information, but there is in the vertical direction. Hence, we don't know directly from the disparity image the value of the vertical component of the correlation window. We have performed calibration tests between the disparity image and the IR image, using a target with a transparent background, and get a constant offset on the order of-4.8×-3.9. This offset is approximate, due to the difficulty of finding crisp edges on the disparity image target. It is consistent with a 9×9 or 9×7 correlation window. It does not vary with depth or Kinect device.

3.4 Lens distortion and focal length

For accurate work in many computer vision fields, such as visual SLAM, it is important to have cameras that are as close to the ideal pinhole projection model as possible. The two primary culprits in real cameras are lens distortion and de-centering.

Typical calibration procedures with a planar target, such as those in OpenCV, can effectively model lens distortion and de-centering. After calibration, the projection error of a camera is typically about 0.1 to 0.2 pixels. The projection error is a measure of the deviation of the camera from the ideal model: given known 3D points, how does the projection onto the image differ from that of the ideal pinhole camera? The projection error is given by the RMS error of all the calibration target points.

We calibrated the IR and RGB cameras of several Kinect devices. Typical projection errors are given in the following table (Table 2).

Table 2: Projection Error

	original	calibrated
IR	0.34	0.17
RGB	0.53	0.16

Table 3: Focal Length and FOV

	Focal Length(pixels)	FOV(degrees)
IR	580	57.8
RGB	525	62.7

3.4.1 Focal Length

The focal length of the RGB camera is somewhat smaller than the IR camera, giving a larger field of view. Focal lengths and field of view for a typical Kinect camera are given in this table. The RGB has a somewhat wider FOV than the IR camera.

3.5 IR to RGB camera calibration

The IR and RGB cameras are separated by a small baseline. Using chessboard target data and OpenCV's stereo calibration algorithm, we can determine the 6 DOF transform between them. To do this, we first calibrate the individual cameras, using a zero distortion model for the IR camera, and a distortion and de-centering model for the RGB camera. Then, with all internal camera parameters fixed (including focal length), we calibrate the external transform between the two (OpenCV calibrateStereo function). Typical translation values are:$(-0.0254,-0.00013,-0.00218)$.

The measured distance between IR and RGB lens centers is about $2.5cm$, which corresponds to the X distance above. The Y and Z offsets are very small.

In the three devices we tested, the rotation component of the transform was also very small. Typical offsets were about 0.5 degrees, which translates to a $1cm$ offset at $1.5m$.

3.6 Align depth image and RGB image

At this point all the individual steps necessary for a good calibration between depth and the RGB image are in place. Here are the steps to transform from the raw disparities of the depth image to the rectified RGB image.

Step 1: Transformation of raw depth values into meters

$$depth = 1.0/(raw_{depth} \times -0.0030711016 + 3.3309495161) \tag{5}$$

Step 2: Mapping depth pixels from depth image coordinates $[x_d, y_d]^T$ to depth camera coordinates $[X_d, Y_d, Z_d]^T$

$$X_d = (x_d - cx_d) \times depth(x_d, y_d)/fx_d \tag{6}$$

$$Y_d = (y_d - cy_d) \times depth(x_d, y_d)/fy_d \tag{7}$$

$$Z_d = depth(x_d, y_d) \tag{8}$$

Step 3: Transform point clouds from depth camera coordinates $\mathbf{X_d}$ to color camera coordinates $\mathbf{X_{RGB}}$

$$\mathbf{X_{RGB}} = R^{-1} \cdot \mathbf{X_d} - R^{-1} \cdot T \tag{9}$$

Step 4: Mapping point clouds from color camera coordinates $\mathbf{X_{RGB}}$ to color image coordinates $[x_{RGB}, y_{RGB}]^T$

$$x_{RGB} = (X_{RGB} \times fx_{RGB}/Z_{RGB}) + cx_{RGB} \tag{10}$$

$$y_{rgb} = (Y_{RGB} \times fy_{RGB}/Z_{RGB}) + cy_{RGB} \tag{11}$$

Note: After projecting to color image coordinates, x_{RGB} and y_{RGB} must be rounded to (1, 640) and (1, 480) respectively.

4 Kinect Calibration

4.1 Calibration Software

The software used here is **RGBDemo 7.0** [6] developed by Nicolas Burrus. The main difference between traditional color camera calibration and that of depth camera is that it is usually difficult to extract corners from chess board patterns. So, for color camera calibration, IR camera calibration as well as the stereo calibration of IR and color cameras the methods implemented in **OpenCV**[7] is borrowed. In addition, since only intensity image is needed in this stage the IR projector can be covered and getting close to the camera to obtain a better estimation of intrinsics and stereo parameters. However, for depth calibration (b, d_{off}), we will need some images with IR and depth, so we cannot cover the IR projector any more.

4.2 Calibration Results

4.2.1 Experiment settings

As discussed above, there are two different settings in this experiment. One is with the IR projector covered, the other leaves it uncovered.
In IR covered setting 26 sequences are captured while in IR un-covered setting 17 frames are acquired. One frame of each setting is given in Figure 5 and Figure 6.

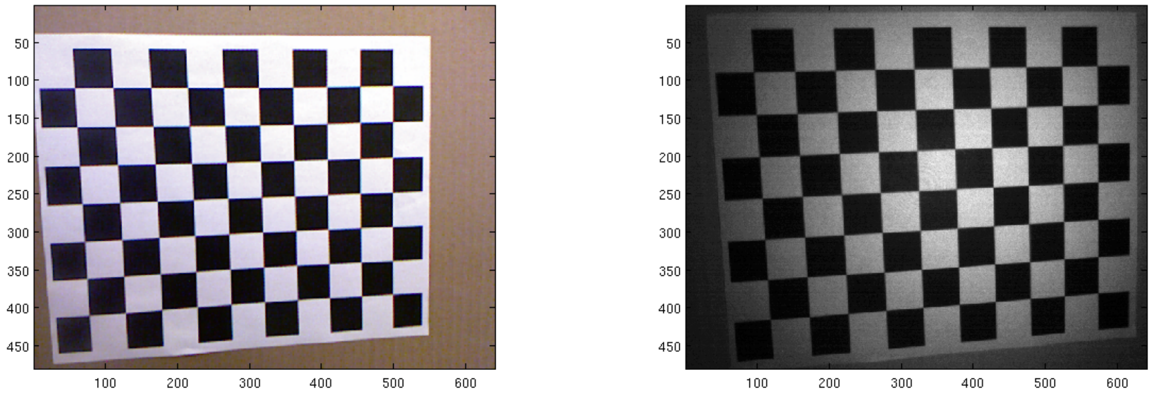

Figure 5: From left to right: color image and intensity image

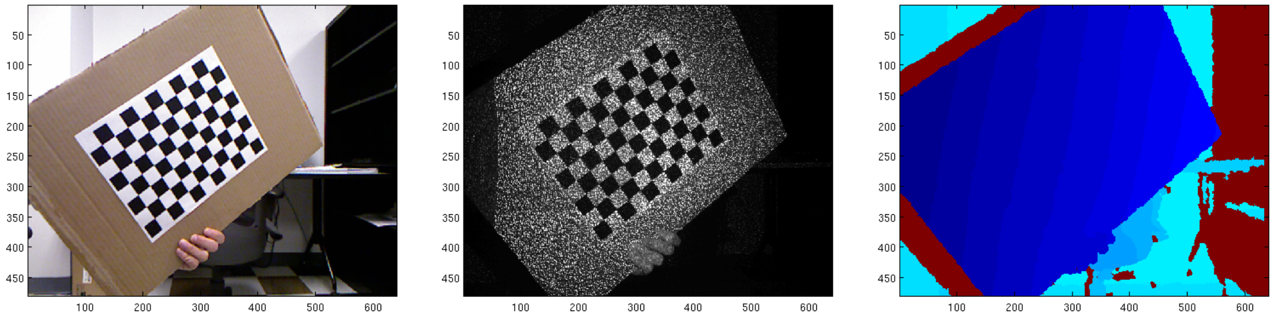

Figure 6: From left to right: color image, intensity image and depth image

4.2.2 Results and analysis

The calibration results are given in Table 4. We can notice that the projection error under IR covered setting $(0.68848 pixel)$ is less than that of IR un-covered setting $(1.19378 pixel)$. The reason is it is much more easier to detect chessboard corners under IR covered setting and the extracted corner is more accurate as well. A comparison of the extracted corners under different settings is shown in Figure 7 and Figure 8.

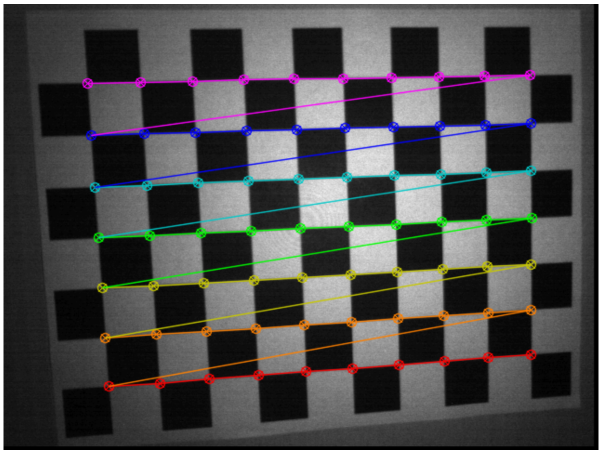

Figure 7: Corners in IR image under covered setting

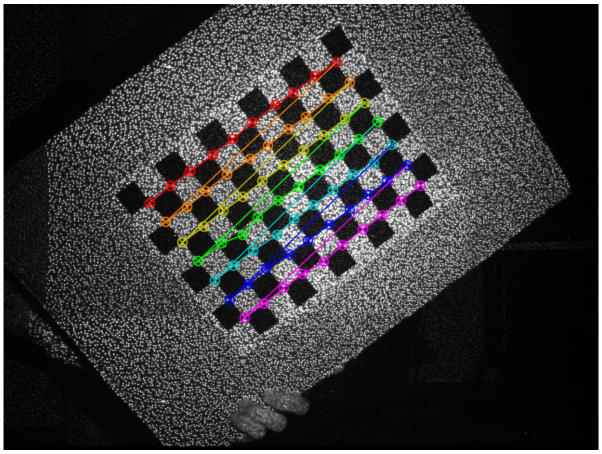

Figure 8: Corners in IR image under uncovered setting

Table 4: Calibration Result

Parameters	IR Projector Covered	IR Projector Uncovered
RGB Intrinsic	$\begin{bmatrix} 517.055 & 0 & 315.008 \\ 0 & 517.679 & 264.155 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 514.120 & 0 & 310.744 \\ 0 & 513.841 & 262.611 \\ 0 & 0 & 1 \end{bmatrix}$
RGB Distortion	$\begin{bmatrix} \text{2.2658e-1} & \text{-7.5265e-1} & \text{2.4148e-3} \\ & \text{-1.9091e-3} & \text{8.3151e-1} \end{bmatrix}$	$\begin{bmatrix} \text{2.0456e-1} & \text{-4.5719e-1} & \text{7.7826e-4} \\ & \text{-3.8524e-3} & \text{-5.5729e-1} \end{bmatrix}$
IR Intrinsic	$\begin{bmatrix} 580.606 & 0 & 314.758 \\ 0 & 580.885 & 252.187 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 596.270 & 0 & 321.490 \\ 0 & 597.689 & 250.363 \\ 0 & 0 & 1 \end{bmatrix}$
IR Distortion	$\begin{bmatrix} \text{-1.8760e-1} & 1.013 & \text{2.3033e-4} \\ & \text{-2.6935e-3} & \text{-1.8375} \end{bmatrix}$	$\begin{bmatrix} \text{-2.5917e-1} & 1.4193 & \text{-3.9565e-3} \\ & \text{-6.6566e-3} & \text{-2.5772e-1} \end{bmatrix}$
R		
$T(mm)$	$\begin{bmatrix} 25.06 & 0.65 & -2.1 \end{bmatrix}$	$\begin{bmatrix} 23.41 & -3.16 & 15.48 \end{bmatrix}$
$b(mm), d_{off}$	N/A	$\begin{bmatrix} 82.63 & 1090.39 \end{bmatrix}$
Projection Error$(pixel)$	**0.68848**	**1.19378**

In order to test the effects of calibration, we align the RGB color image with depth image together in a few different scenes according to the parameters obtained by calibration. For every scene, depth image is rectified by the steps described in 3.6 firstly, then edges in color image is detected with Canny edge detector. And finally the edge pixels are mapped from color image to depth image. The result before calibration and after calibration is given in Figure 9, Figure 10 and Figure 11.

5 Conclusions

Through this study, the principles and the parameters of Kinect both intrinsic and extrinsic are well understood. Although the inventors describe the measurement of depth as a triangulation process (Freeman etal,. 2010) in their patent, no further information has been disclosed. So, at the moment the ideas about depth measurement are all by guess. ROS illustrates that the inverse of depth is proportional to the disparity by experiments, and a derivation from original disparity to normalized disparity is given in Equation 3. But their description is not complete clear to us, which needs to be clarified in the future.

References

[1] http://en.wikipedia.org/wiki/Kinect. [Accessed June 1 2012].

[2] B. Freedman, A. Shpunt, M. Machline, and Y. Arieli, “Depth mapping using projected pattern,” May 2010.

[3] A. Shpunt and Z. Zalevsky, “Three-dimentional sensing using speckle patterns,” April 2009.

[4] http://www.ros.org/wiki/kinect_calibration/technical. [Accessed June 1 2012].

[5] http://www.ros.org/wiki/kinect_node/Calibration. [Accessed June 1 2012].

[6] http://labs.manctl.com/rgbdemo/index.php/Documentation/Calibration. [Accessed June 1 2012].

[7] http://opencv.willowgarage.com/documentation/python/camera_calibration_and_3d_reconstruction.html. [Accessed June 1 2012].

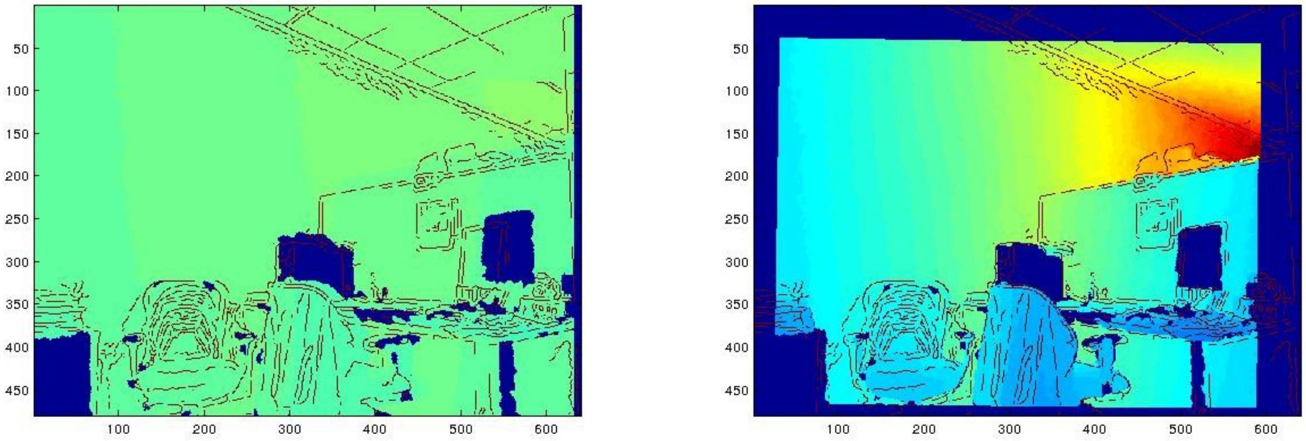

Figure 9: Left: Before calibration; Right: After calibration

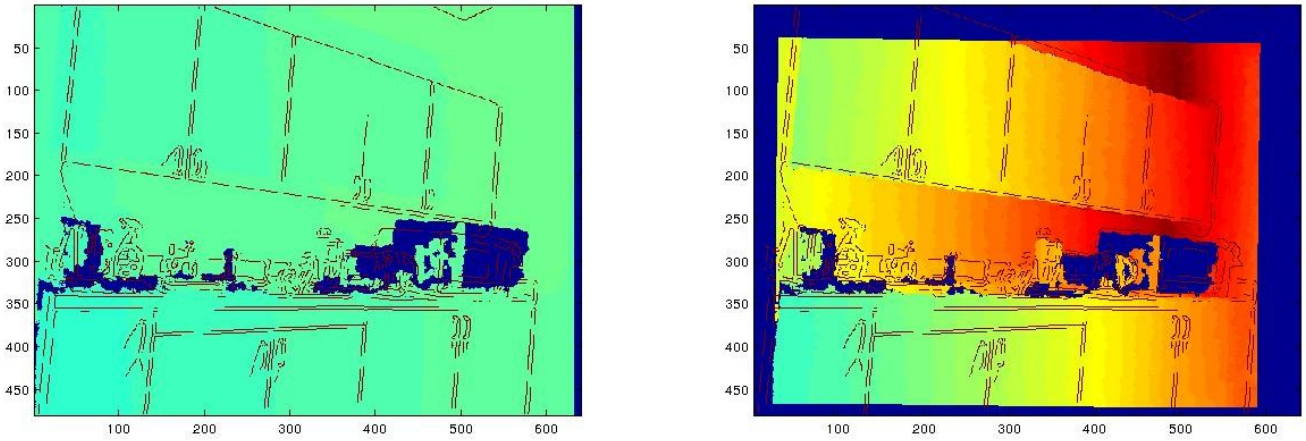

Figure 10: Left: Before calibration; Right: After calibration

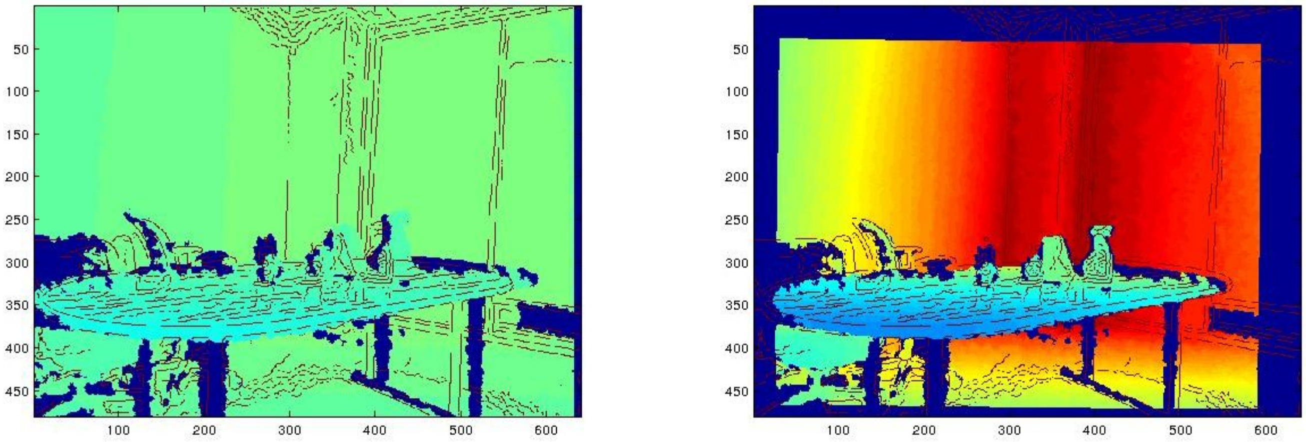

Figure 11: Left: Before calibration; Right: After calibration