Title

Minimally Intrusive Gaze Detection in Clinical Environments

Author

Wang, Yuchen

Publication Date

2015

UNIVERSITY OF CALIFORNIA, SAN DIEGO

Minimally Intrusive Gaze Detection in Clinical Environments

A Thesis submitted in partial satisfaction of the requirements
for the degree Master of Science

in

Bioengineering

by

Yuchen Wang

Committee in Charge:

Professor Vikash Gilja, Chair
Professor Gert Cauwenberghs
Professor Marcos Intaglietta

2015

The Thesis of Yuchen Wang is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California, San Diego

2015

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to express my special appreciation and thanks to my advisor Professor Dr. Vikash Gilja for his patience, motivation, and help along my way. His guidance is invaluable for my research and the writing of this thesis. I also want to especially thank all of my labmates for their encouragement and suggestions for my M.S. thesis.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Gert Cauwenberghs, Prof. Marcos Intaglietta, for their attendance at master defense, insightful comments, and questions.

ABSTRACT OF THE THESIS

Minimally Intrusive Gaze Detection in Clinical Environments

by

Yuchen Wang

University of California, San Diego, 2015

Professor Vikash Gilja, Chair

Motivated by the Electronic Health Record (HER) system's demand of capturing patients' multimodal activities and the wide application of gaze detection, we develop a minimally intrusive gaze detection system with Microsoft Kinect sensor and test its performance in a simulated clinical environment. Traditional methods require either a close distance between the camera and the user or a fixed head pose which may severely interrupt the clinical workflow and the interaction between the physician and the patient. Compared with the traditional methods, our system allows a wider range of detection, while achieving an accuracy around 70%.

INTRODUCTION

1. Free behavior monitoring system

In recent years, the Electronic Health Record (EHR) system have dramatically facilitated the monitoring of patient’s health and promoted preventive care. Compared with traditional paper-based records, electronic records highly reduces the medical errors and increases the efficiency of physician-patient interaction. The challenges for an accurate EHR system include the acquiring multimodal data such as medical history, behavioral record, etc. and the employing suitable algorithms to extract useful information for efficient treatment or disease-prediction. Our project focuses on the multimodal data capture, especially gaze information. As shown in [1], a majority of patients who suffer strokes from severe head injuries also end up suffering from arm impairment every year. By employing the automatic body movement capturing system, we can accurately evaluate the paretic arm mobility, which when compared with the traditional doctor experience-based method, yields essential information about activity limitations that patients experience during recovery [2]. The importance of multimodal data can be reflected not only in EHR system, but also in other fields like neuroscience. As human beings, we experience environmental stimulus, like images and sounds when the neuron in our nervous systems receive, transmit, and interpret signals. Our body will produce the corresponding responses through the communications between neurons by sending electrical signals. Modern neuroscientists have put many efforts into mapping function between stimuli and response. Most of the past work used controlled paradigms to reduce the number of stimulus and simplify the analysis, but no clear method to capture multimodal data in a free environment to better characterize the stimulus and increase the accuracy of the decoding has been proposed. To meet the requirements of a more accurate capturing of the multimodal data, we come up with this Free Behavior Monitoring (FBM) system to record patient's speech, body movements, gaze information, etc. in the natural setting, the FBM system can render a deeper insight into both healthcare problems and scientific problems [1-2]. In my project, I will focus on the gaze detection of the system. The motivation and importance of the gaze detection will be discussed in the following the section

2. Gaze detection

2.1 Application

A wide variety of fields including medical research [3-5], human-computer interaction [6-7], and neural science [8-11], use gaze detection techniques. In modern medicine, early identification of robust risk factors are essential for the early treatment of diseases like autism and Alzheimer disease. According to past research [4-6], patients with autism spectrum disorder (ASD) tends to fixate on the mouth region longer than the eyes during face viewing and prefer to focus on visual repetition, such as the spinning of a car wheel so gaze detection shows promise in characterizing the early features of ASD. by presenting toddlers with moving geometric patterns and tracking their eyes, Pierce and his collogues [4], find that if a toddler spent more than 69% of his or her time fixating on geometric patterns, the predictive accuracy of toddler having ASD can reach 100%. Alzheimer's disease (AD), frontotemporal dementia (FTD) and sematic dementia (SD) are another group of neurodegenerative disease that can cause severe changes in personal or social behavior, language ability, short-term memory loss, etc. By using gaze tracking technology to detect the mutual gaze between couples during a conversation during relationship conflicts, Sturm and his collogues [7] reveal that mutual gaze is preserved in AD couples, but diminished in FTD couples and enhanced in SD couple compared with the control group.

The gaze detection can help us better understand the relationship between the stimulus and neural responses. Recently, the studies [8, 9] on rhesus monkeys and human beings demonstrate that different visual stimulus will enhance or suppress the auditory cortex’s responses to the same voice stimuli. Ghazanfar et at recorded local field potential (LFP) activity in both core and lateral belt regions of the auditory cortex in two rhesus monkeys [8]. When the monkeys are viewing vocalizing conspecifics with “incongruent” conditions, for example, a coo face is paired with a grunt voice, the monkey’s LFP is notably enhanced and vice versa, is suppressed under “congruent” conditions. Similar studies have also been done on human beings [9]. In Golumbic et al’s study, they adopt 13 native English-speaking participants with normal hearing and record their magnetoencephalography (MEG) in an actively magnetically shielded room when they are exposed to eight kinds of stimulus (see Figure 1).

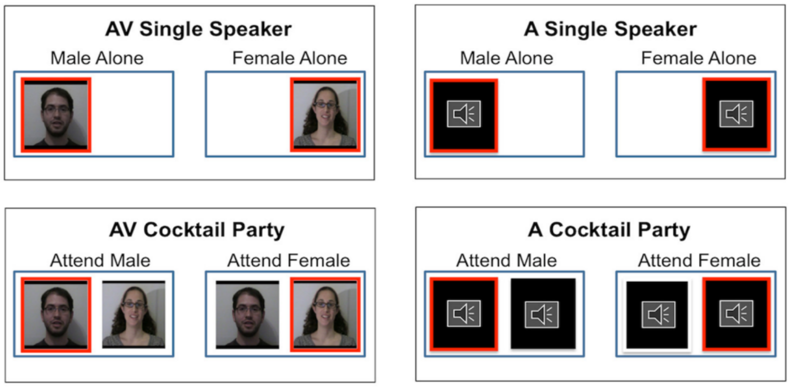

Figure 1: Experiment paradigm. Illustration of the four experimental conditions.

By analyzing the MEG data, they demonstrates that viewing a speaker's face enhances the ability of auditory cortex to track that speaker's speech. All of these studies show us that the visual input can modulate the neural output. In those studies, if the use of gaze detection techniques can help us find subject's precise gaze position, then we can better understand which specific object is influencing the neural responses.

Another application of gaze detection is in human-computer interface field, especially for the disabled [10-11]. Hutchinson and his collogues show that a non-intrusive eye-tracking system paired with the user's slight head movement allows the user to interact with the computer in running communication and manage other peripheral devices without physical contact [11]. The accurate gaze detection algorithm will benefit those people who cannot move their bodies due to accidents or diseases like amyotrophic lateral sclerosis (ALS).

2.2 Techniques

Most of the recent widely used methods are video-based gaze detection where one or multiple cameras remain focusing on the users' eyes and estimating the gaze vector based on acquired images. The algorithm usually include two parts: eye localization and gaze estimation [12].

2.2.1 Eye localization algorithm

For eye localization, we need firstly to detect the eyes' positions and then localize the pupil centers within the eyes for each frame of image [12]. Humans' open eyes have similar elliptical shapes, so the methods like elliptical shape fitting [13] and deformable eye model fitting [14] have been developed to interpret eye positions. One of the weakness of shape fitting models is that they are usually computationally demanding and require very high contrast images. In order to overcome those obstacles, researchers have been trying to detect the eyes by using the distinct features around eyes, like eye edges [15] and filter responses [16]. Employing active IR illuminations [17-19] can also increase the accuracy of eye location. In our research, the Microsoft Kinect is able to offer the eyes' spatial location in the camera space for each frame after mapping them into color space.

2.2.2 Gaze Estimation Algorithm

The approaches of gaze detection or estimation can be roughly categorized into geometrical method and machine learning method.

2.2.2.1 Geometrical Model-based Gaze Estimation

Figure 2 shows a typical simplified eye model commonly used in gaze detection research [20-22]. There are two main axes in this model: the first one is the optical axis which represents the geometric center line of the eye and the second one is visual axis which connects the fovea and the point of gaze [20]. The goal of this method is to estimate the direction of visual axis and integrate this vector with the three dimensional structure about the objects in the scene. By computing the intersection of the gaze vector with the nearest object in the scene, the point of regard (POG) is located.

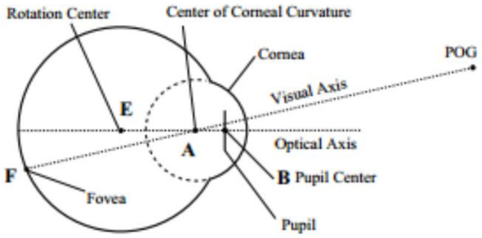

Figure 2: Eye model [20]

In order to estimate the direction of visual axis, several parameters need to be acquired such as the radius, center of eyeball (the rotation center in Figure2), direction of the optical axis, and the relative angle between optical and visual axis. Since the model is built in three dimensional world, one of the advantage of this method is that it takes the 3d motion of the head or eyeball, like rotation, into consideration and allows the user to move the head freely as long as the cameras are still able to detect the eyes. However, the requirement of estimating 3D points also results in problems like synchronization, point matching, occlusion and more data from the utilization of stereo and active cameras require processing [21, 22].

2.2.2.2 Geometrical Model-based Gaze Estimation

Machine learning - based methods usually includes a mapping from 2D eye image to the screen which the user is asked to gaze at (see Figure 3) [23].

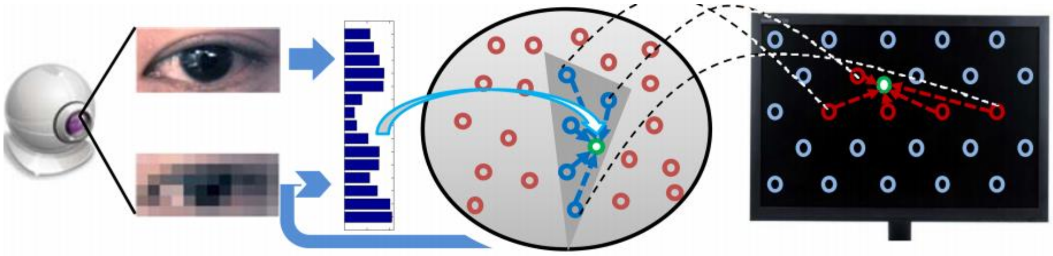

Figure 3: Flow of the machine learning-based method [23]

As shown in Figure 3, by employing a single RGB or IR camera and using eye localization algorithms, we can generate an image of a single or both eyes. After post-processing of these images such as extracting the features [23] and resampling, the images will be transformed into a multi-dimensional feature vector as shown in the light circle in the middle of Figure 3. Eventually, machine learning algorithms are performed to estimate gaze position on the screen by using the feature vector. During this process, different learning algorithms have their own advantages. The most common used linear regression model [23] is easy to implement and computationally efficient, but polynomial regression [24] can compensate notice nonlinearities when the image is noisy. Artificial neural network (ANN) and their variants are also used, but some parameters in the model like hidden layer number are very difficult to optimized, where in order to increase the robustness and efficiency of the model, researchers need to find more efficient features in the model.

The machine learning algorithm usually requires the fixed heads and the parameters in the model lack physical interpretations, but compared with geometrical model-based methods, they are easier to implement and able to give good results with fixed head.

2.2.3 Gaze Estimation Calibration

All of these methods still need calibration sessions. For geometrical model, we need the calibration to estimate cornea curvature, iris's size and other parameters decided by the specific assumption of eye structure. For machine-learning methods, we also need to determine the parameters of the mapping functions [12].

According to the literature [12-24], users are always required to keep a fixed close distance to the camera and the head motion must be kept in a very limited range in order to acquire a clear picture of the eye, specifically the pupil. This limitation in motion is opposite to the goal of the free behavior capture system, so we introduce a new method that combines the geometrical and machine learning method to resolve this problem while keeping the accuracy within an acceptable level. The needs of both the 3D structure and the RGB images of eyes led us to look for an accurate depth camera and a high resolution color camera such as the Microsoft Kinect Sensor.

3. Microsoft Kinect sensor

Kinect is categorized as a 3D depth camera containing a depth sensor, a RGB camera, and multi-array microphone (see Fgiure.4).

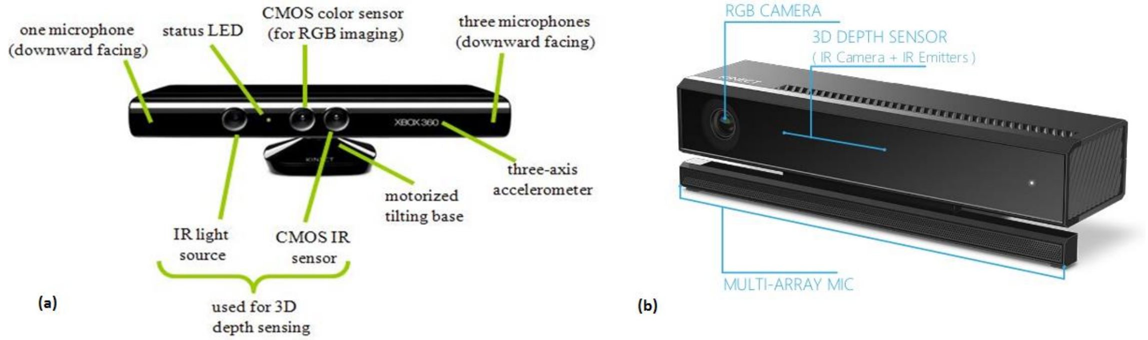

Figure 4: Microsoft Kinect sensor. (a) First version (b) Second version

Those sensing cameras provide the basis for full-body 3D motion capture, facial recognition and voice recognition capabilities [25].

3.1 Coordinate system

Kinects can stream color and depth data in a 30Hz framerate. Each data type has its own coordinate space and Kinect offers the functions to map data between the spaces. Each frame from the color sensor is made up of pixels, each of which contains red, green, blue components and is specified by(x_c, y_c) in a color image [26]. The IR light source and CMOS IR sensor work together to give a depth image of everything visible in the field of view of the CMOS IR sensor. The value of each pixel, at a particular(x_d, y_d) coordinate represents the Cartesian distance, in millimeters, from the camera center to the objects in that pixel [27]. The depth coordinate(x_d, y_d) represents a pixel in the depth image, but can be transformed into the camera coordinate (x, y, z) to represent a physical unit in 3D space. The original point (0, 0, 0) of the camera space coordinate is located at the center of the IR sensor, the x, y axes go to the left , up of the sensor respectively and the z axis extend in the direction that the Kinect is facing (see Figure 5)[27].

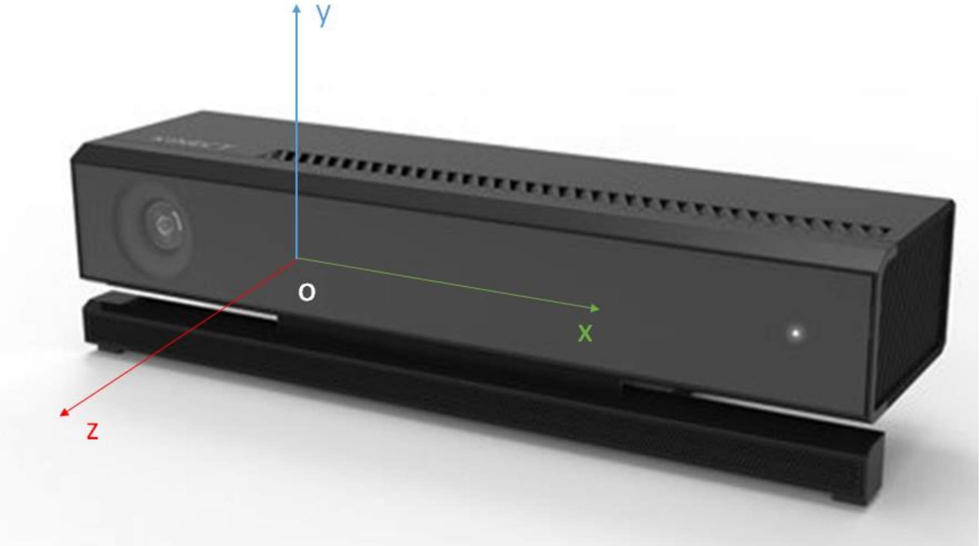

Figure 5: Coordinate system

3.2 Body tracking and facial recognition

In addition to streaming different kinds of images, Microsoft Kinect (abbreviated as Kinect) is also able to track up to six people's body motions and faces at high fidelity by utilizing depth images. In order to track the skeleton, the human body is segmented into different parts and represented by several joints, such as head, neck, and shoulders[25] (see Figure 6). By this intermediate representation, combing with machine learning algorithm and a large amount of training dataset[27], Kinect can provide the spatial location(x, y, z) of those joints in camera coordinate, accompanying each depth frame. By tracking the motion of those joints, we can estimate the motion of the human body. Since the Microsoft Kinect is designed for XBox 360, the underlying tracking algorithm and the training dataset assume the user are in the standing, not touching any objects state, which may cause some inaccuracies if the user is sitting on a bed or in a wheelchair [28].

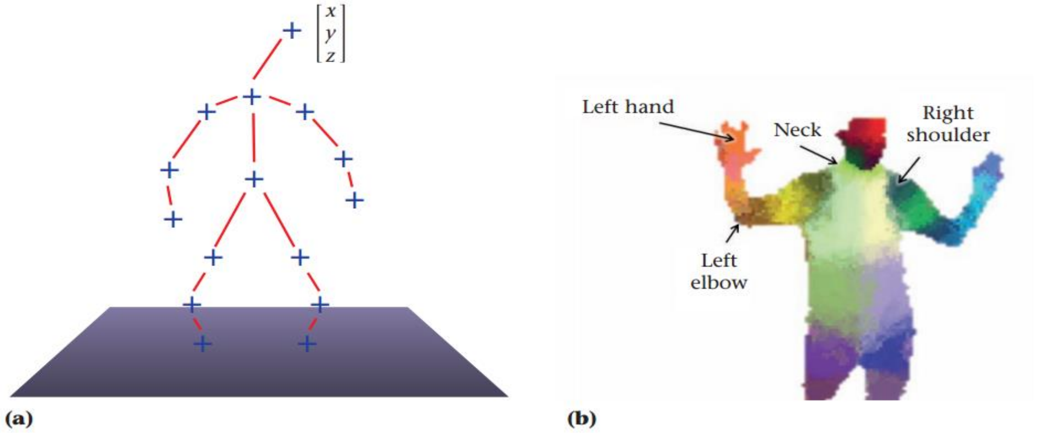

Figure 6: Body tracking (a) skeleton representation of human body (b) color map of body part

Face tracking engine is another Software Development Kit (SDK) offered by Microsoft which enables us to track human faces in real time. The facial information from Kinect can be divided into two groups: basic and High-definition facial information. The former one offer the abstract information of humans head like the orientation, head center position and the parameters called Animation Units (AU). Each of Animation Units (AU) represents the degree of the contraction or relaxation of a single or a group of muscles on the faces; for instance, AU1 shows the motion of frontalis ranging from zero to one. By following the rules of Emotional Facial Action Coding System (EFACS) [30], we can use these values to estimation the emotion of the user. For example, the summation of values of AU6 and 12 can be used to detect the degree of the happiness of users (See Figure 7).

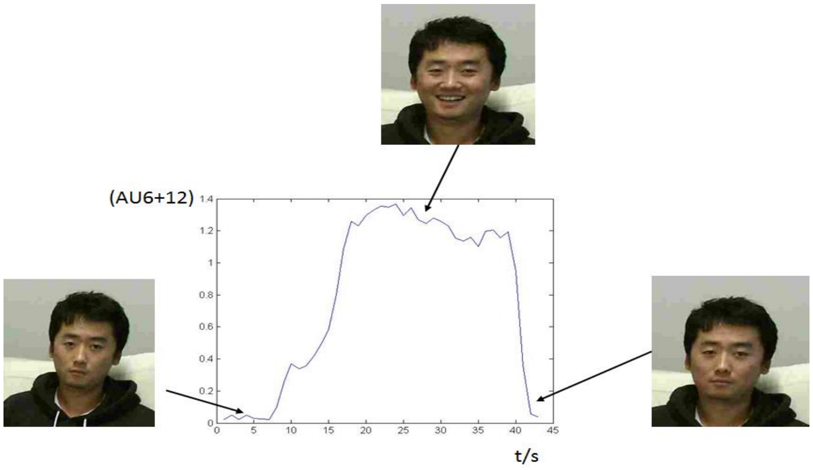

Figure 7: EFACS's representation of happiness.

As can been seen from Figure 4, with the user's face changes from neutral face to smiling, the sum of AU6 and AU12 increases from 0 to 1.4. And when the user returned back to the neutral face, the value also went back to 0. It illustrates that the Kinect has the potential to benefit further psychological and behavioral research.

The other group of facial data we can get from Kinect is called high-definition (HD) facial mask which consists pf 1347 points in camera space (see Figure 8). Each facial point has a clear physical meaning where the 19th point tracks the motion of user's nose tip [28]. The HD face tracking does not only detect the face from depth images, but also renders a face in 3D space.

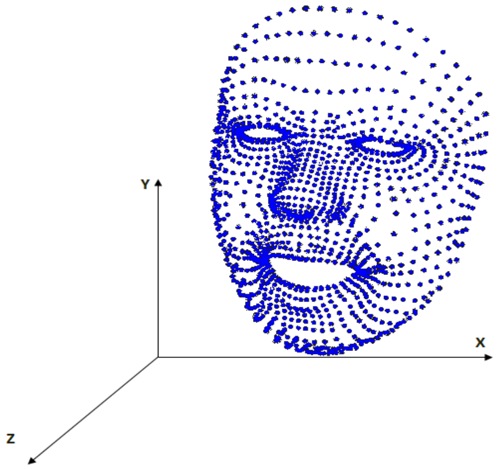

Figure 8: High definition Face Mask

By analyzing the temporal changing of these points in camera space, we can estimate the motion, like rotation or translation, of users' heads and by transforming these data from the camera space to depth space, eyes' detection become a lot easier which will benefit gaze detection.

The 3D depth accuracy of the Kinect camera has been evaluated quite extensively [29-32]. In [29], by quantitatively comparing the Kinect with a stereo-cameras and a 3D-Time-of-Flight (3D-TOF) camera, the author shows that Kinect's performance is better than TOF camera and close to that of the stereo-camera. Similar results has been documented in[30] whose result also reveals that Kinect's accuracy is very close to laser sensor when the distance is less than 3.5 meters. But when the distance between the object and Kinect increases, [31] shows that the random error in depth measurement ranges from several millimeters to 4 cm.

Performance of body tracking has also been investigated under different conditions [33], [34]. In [33], authors notice that the accuracy of the body tracking depends heavily on the type of poses. So they compare the performance of Kinect's body tracking algorithm with another more established maker-based motion capture system by adopting six exercises (see Figure 9) that are challenging for pose estimation algorithms.

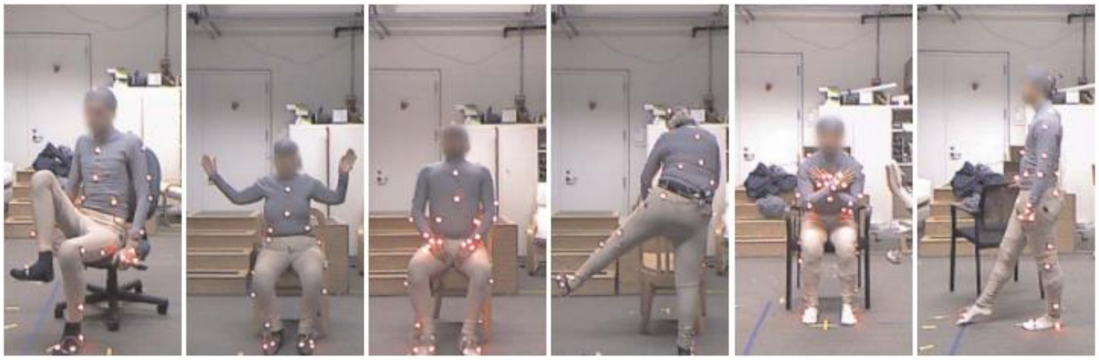

Figure 9: Six exercises [33]

The results show that, in a more controlled posture, like standing, the accuracy of Kinect's joint estimation is comparable to the other system, but, for other postures, the variability of the Kinect's pose estimation is about 10cm. Brook et al. [34] study the accuracy of Kinect's measuring movement of people with Parkinson's disease (PD). The movement includes quiet standing, multidirectional reaching, etc. in nine people with PD and 10 controls. By comparing with a Vicon 3D motion analysis system, Kinect is very accurate in measuring the timing of movement repetition, but varies a lot when measuring spatial characteristics: excellent for gross movement, but poor for fine movement like hand clasping. Unfortunately, not many similar studies on face tracking are documented.

METHOD

The diagram of our gaze detection algorithm is shown in Figure 10. In our study, every single Kinect deployed in our project is calibrated firstly, including the intrinsic calibration for depth, color cameras and the extrinsic calibration between the depth camera and the color camera. Another extrinsic calibration is executed so that we can unify all Kinects' depth cameras' coordinate systems in our study. All of the parameters from the calibration session will be stored for further use, like transforming data between different spaces. After that one Kinect will be deployed to stream out patient's high-definition face points which are used to localize eye positions in the color picture and estimate the geometrical structure of eyeballs. Eventually, the gaze vector will be estimated by the linear regression model with the above information. The details of each step will be described in the following sections.

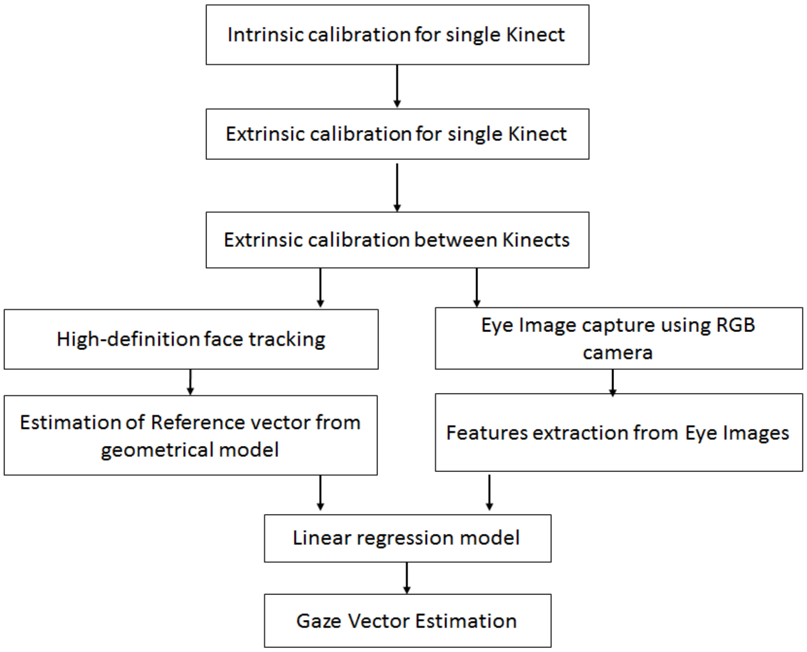
Intrinsic calibration for single Kinect
Extrinsic calibration for single Kinect
Extrinsic calibration between Kinects
High-definition face tracking
Eye Image capture using RGB camera
Estimation of Reference vector from geometrical model
Features extraction from Eye Images
Linear regression model
Gaze Vector Estimation

Figure 10: Method Diagram

1. Setup

The final goal of our project is to make the Free Behavioral Monitoring system feasible in clinical environment, so we try to simulate the clinical scenario in our project. The whole system and the test environment are shown in Figure 11.

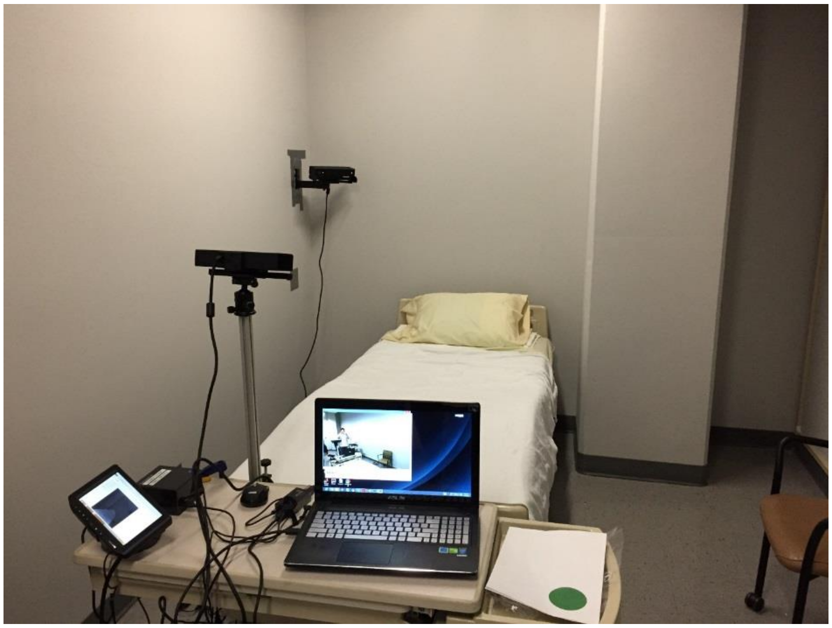

Figure 11: Free Behavioral Monitoring system and Test Environment

In our project two Kinects will be employed. One of them is mounted on the footboard of the patient's bed and facing the patient which can record the user's facial and body information. The other one is clamped around the headboard to capture the environmental information like the worldview of the patient, as shown in Figure 12. The data from these two Kinects will be sent to the recording PC and stored for further analysis.

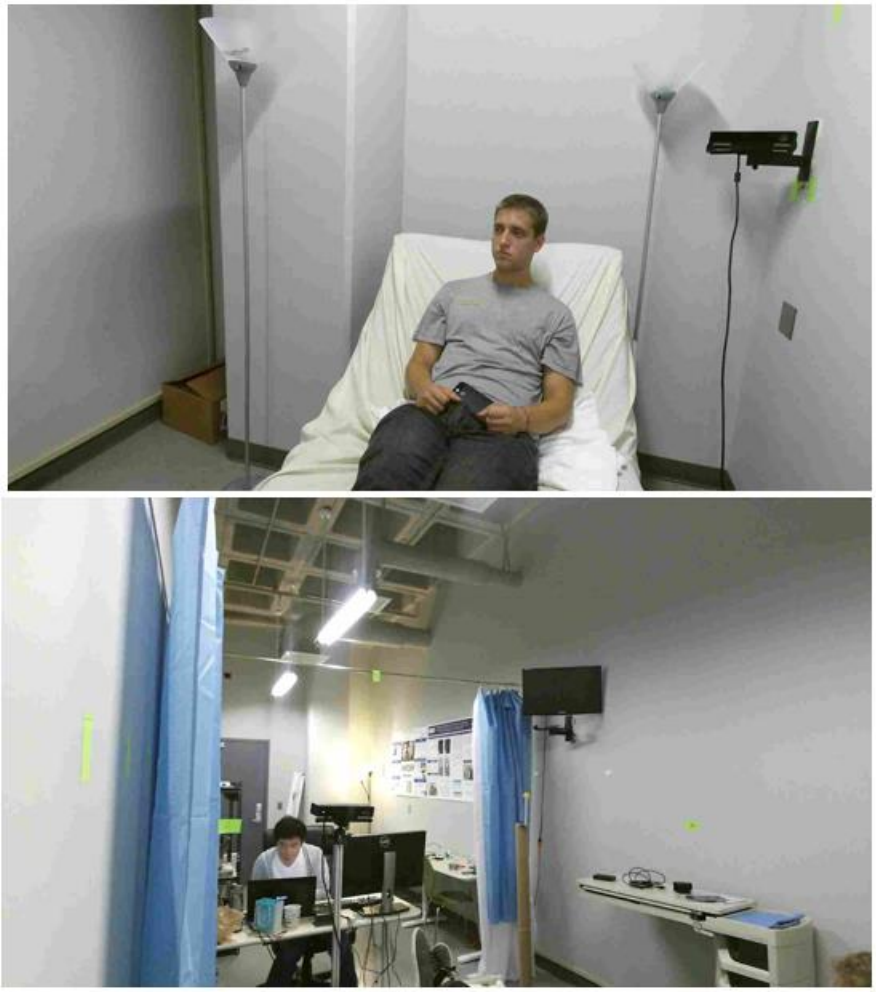

Figure 12: Views of Kinects.
(Top: from the Kinect on the headboard, Bottom: from headboard mounted Kinect)

2. Calibration

As discussed in the introduction section, Kinect has two cameras: color camera and IR camera. According to the manuscripts, all of the cameras have been calibrated after being fabricated to get the parameters like focal length, lens distortion coefficients, etc. which are unique for each Kinect and stored to nonvolatile memory on the sensor. Those parameters are extremely important for mapping the data between different spaces, but we find that sometimes the accuracy of those parameters could slip over time which could be due to physical collision, etc. In order to compensate for it, we will do our own calibration instead of simply trusting in built-in calibration results. The calibration can be categorized into two parts: intrinsic calibration and extrinsic calibration.

2.1 Intrinsic calibration

Intrinsic calibration is a process of estimating the parameters which characterizes the geometrical relationship between the scene and the camera. These parameters determined which pixel the three dimensional position (camera space coordinate for Kinect) of a point is projected to on the images. Intrinsic calibration has been studied extensively [36, 37] and the pinhole camera model is widely used to approximate this mapping from a 3D scene to a 2D image.

Pinhole camera model is a first order approximation of the real camera which does not take nonlinear phenomenon like geometric distortion into account and is based on the assumption that the camera aperture is a point. The principle of a pinhole camera has been shown in Figure 13. As shown in a), a 3D object is projected to the image plan through the camera aperture (pinhole) and the distance between the image plan and pinhole is defined as the focal length. After defining the axis for the image plane and the image center, defined as the intersection of the optical axis and the image plane, the equivalent mathematical representation is shown in b).

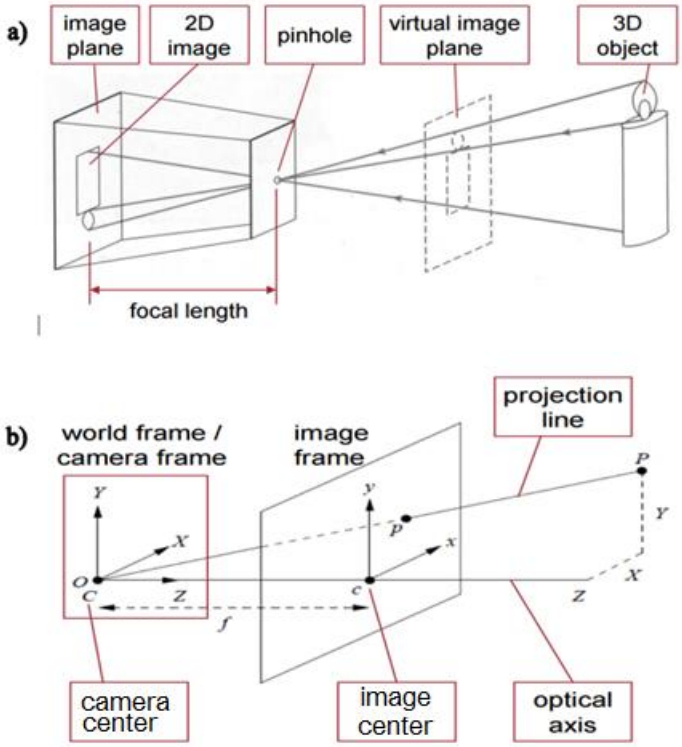

Figure 13: The Principle of Pinhole Camera. a) Physical model b) mathematical representation [36]

In b), it can be seen that a point in camera space (X, Y, Z) will be projected to a point on the image frame (x, y) based on following equations:

$$\begin{cases} \frac{x}{X} = \frac{f}{Z} \\ \frac{y}{Y} = \frac{f}{Z} \end{cases} \tag{1}$$

,where we notice that for negative X, the x will be negative which is obey to our common sense that the coordinates of a pixel is represented by a pair of positive integers in the computer. In order to make it positive, we usually move the center of the image coordinates to the left bottom corner of the image. Under this new coordinate system, if we define the coordinate of image center is (x_0, y_0), each pixel (x_n, y_n) can be related to three dimension point (X, Y, Z) by:

$$\begin{cases} x_n = \frac{f}{Z} X + x_0 \\ y_n = \frac{z}{Z} Y + y_0 \end{cases} \qquad (2)$$

As we discussed before, the pinhole model doesn't take the distortion into account, but for Kinect's cameras, especially depth camera, it has been shown that the radial distortion is very severe near the image edges because of the way IR cameras work. As documented in [37], imaging a three dimension point (X, Y, Z), the projected pixel coordinate from the pinhole model, will be expressed as:

$$\begin{cases} x_p = \frac{X}{Z} \\ y_p = \frac{Y}{Z} \end{cases} \qquad (3)$$

After considering the lens distortion and distortion coefficients$a_i (i = 1 \dots 5)$, the new pixel coordinate is defined as follows:

$$\begin{cases} x' = (1 + a_1 r^2 + a_2 r^4 + a_5 r^6) x_p + dx \\ y' = (1 + a_1 r^2 + a_2 r^4 + a_5 r^6) y_p + dy \end{cases} \qquad (4)$$

, where $r^2 = x_p^2 + y_p^2$ and dx, dy are defined as follows:

$$\begin{cases} dx = 2a_3 x_p y_p + a_4 (r^2 + 2x^2) \\ dy = 2a_4 x_p y_p + a_3 (r^2 + 2y^2) \end{cases} \qquad (5)$$

After moving the coordinate origin to the left bottom corner of the image, the pixel's coordinate stored in computer will be expressed as follows:

$$\begin{cases} x_n = f x' + x_0 \\ y_n = f y' + y_0 \end{cases} \qquad (6)$$

By using this model, according to documents [37], the nonlinear distortion problem can be solved. However, it also brings us more parameters to estimate (distortion coefficients). In our project, we use camera calibration toolbox [37] and a chessboard to finish the intrinsic calibration for each depth camera and color camera in two Kinects.

During the calibration period, we will place the chessboard in nine different positions relative to the camera (See Figure 14). By analyzing the images of chessboard captured by the camera, we can get the intrinsic coefficients for each camera. The calibration results of all of cameras will be shown in results section.

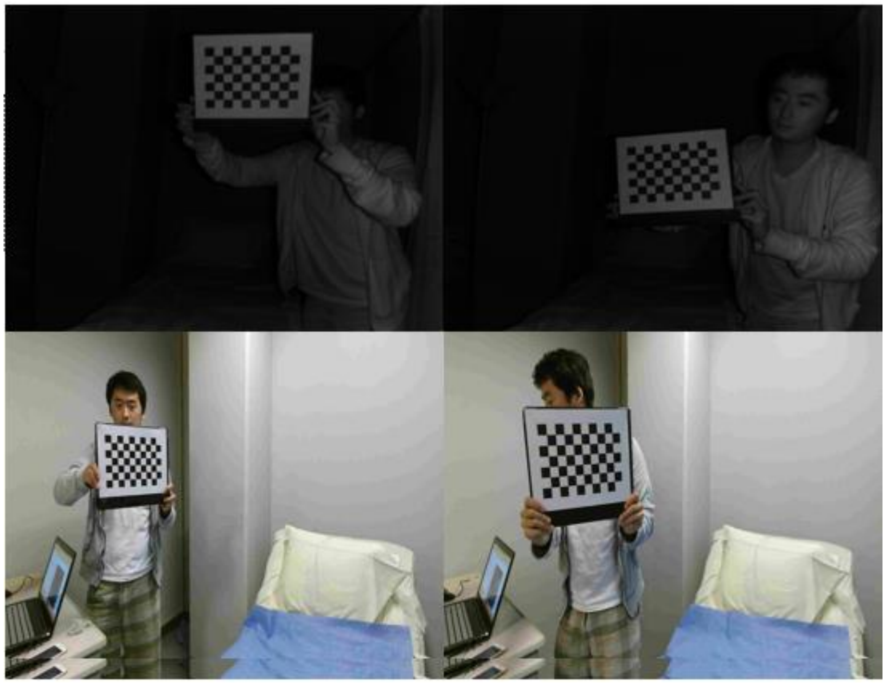

Figure 14: Intrinsic calibration paradigm (Top: IR Camera, Bottom: Color Camera)

2.2 Extrinsic calibration

Comparing with intrinsic calibration, extrinsic calibration focuses more on estimating the geometrical relationship between different cameras' coordinates, since each camera has its own coordinate. It is important, especially when we want to reconstruct the 3D structure of the scene by fusing different streams of data to get a better understanding of the environment. The extrinsic calibration consists of two parts. The first one is to unify the two sensors' coordinate systems in one Kinect. Another one will be unifying two Kinects' coordinates. Since each Kinect's coordinate system is just its depth camera's coordinate system, so the extrinsic calibration between two Kinects is equal to calibrate two IR cameras extrinsically.

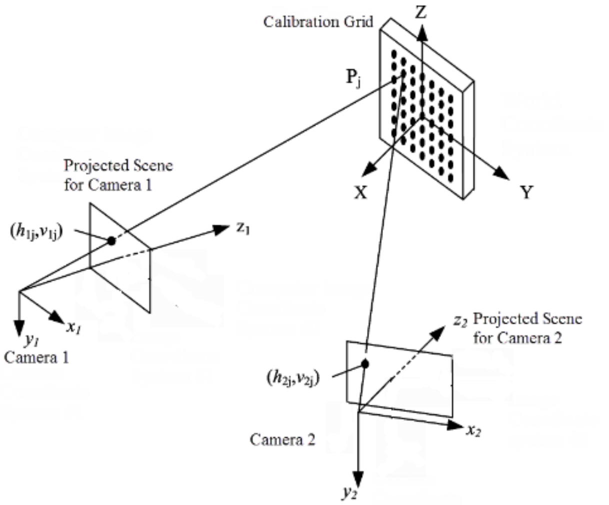

Figure 15: The principle of extrinsic calibration

The principle of extrinsic calibration is shown in Figure 15. It shows a typical stereo camera system and two cameras' coordinate systems are represented by (x_1, y_1, z_1) and (x_2, y_2, z_2) respectively. For these two coordinate systems, a point in 3D space will have two coordinates expressed as (x_{o1}, y_{o1}, z_{o1}) and(x_{o2}, y_{o2}, z_{o2}). Based on the basic algebra knowledge, we know that this two coordinate can be related as follows:

$$\begin{pmatrix} x_{o2} \\ y_{o2} \\ z_{o2} \end{pmatrix} = R \begin{pmatrix} x_{o1} \\ y_{o1} \\ z_{o1} \end{pmatrix} + T \qquad (7)$$

,where R is a 3×3 rotation matrix, describing the relative orientation between two cameras' coordinate systems, and T is a 3 ×1 translation matrix, describing the relative displacement between two systems. R has to satisfy the condition that $\det(R) = 1$. The final goal of extrinsic calibration is to estimate these matrixes through the calibration session. In our study, the calibration between the color camera and the IR camera in one single Kinect is performed by introducing a calibration chessboard at various poses. By aligning the chessboard patterns in two images from two cameras, we can solve for the parameters in the rotation and translation matrix. The analysis is also finished by camera calibration toolbox [37].

The use of chessboard to calibrate the color camera and the depth camera in one Kinect works because of the little relative orientation between the two cameras, so the same side of chessboard can be captured by both cameras easily and clearly. But for the two depth cameras in our two Kinects system, they are almost facing each other. The requirement that seeing the same side is very hard to achieve. In order to finish the calibration, the spherical target is being adopted. Because even though two Kinects can only see two opposite sides of the same target, they can estimate the target centers' position using the points they see. By matching the centers, the extrinsic calibration can be finished. The first step of the method is to detect and segment the sphere from the raw point cloud in 3 dimensional space. Then the radius and the center of the sphere is estimated by fitting sphere's points to the following equation:

$$(x_c, y_c, z_c, r) = Arg_{x_c,y_c,z_c,r} \min \sum (x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2 - r^2 \qquad (8)$$

Prior knowledge of the sphere's radius will increase the accuracy of estimation. During the calibration session, the sphere will be placed at least four different places. Once the 3D coordinate of the sphere center in jth position is extracted for both Kinects, notated as $(x_{c1}^j, y_{c1}^j, z_{c1}^j)$ and $(x_{c2}^j, y_{c2}^j, z_{c2}^j)$, the relative geometrical relationship between those two Kinects can be computed by using the following equation:

$$(R,T) = Arg_{R,T}\min \textstyle\sum_j || \begin{pmatrix} x_{c2}^j \\ y_{c2}^j \\ z_{c2}^j \end{pmatrix} - R \begin{pmatrix} x_{c1}^j \\ y_{c1}^j \\ z_{c1}^j \end{pmatrix} - T \,||^2 \qquad (8)$$

The results of both extrinsic calibrations will be shown in the result section and the accuracy will be discussed.

3. Head orientation

After finishing the calibration session, we can put different sources of data under same coordinate system and start the gaze detection. If we think about how we gaze at things for most of times, we can find that we will turn our head to the direction we want to see. In another words, the head orientation can be used to represent the gaze direction in some cases. Especially when we know nothing of the users' eye information, but still need to have a guess of the gaze direction. So the head orientation can be a substitute for the gaze direction when the user is detected to be blinking or when the tracking of eyes are lost.

As we discussed in the introduction, the basic face tracking API can stream out the face orientation data in terms of the quaternion (x, y, z, w). By using the equations as follows:

$$\begin{cases} Pitch = \text{arctan}(\frac{2(yz+wx)}{w^2-x^2-y^2+z^2}) \\ Yaw = \text{arcsin}(2(wy-xz)) \\ Roll = \text{ rctan}(\frac{2(wy-xz)}{w^2-x^2-y^2+z^2}) \end{cases} \qquad (9)$$

, the quaternion can be used to calculate the yaw, pitch, roll of the heads (see Figure 16).

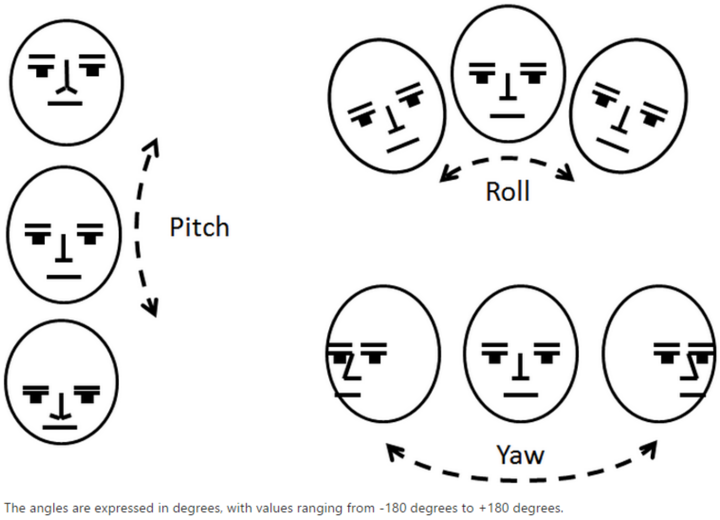
Pitch
Roll
Yaw
The angles are expressed in degrees, with values ranging from -180 degrees to +180 degrees.

Figure 16: Head Pose Angles

The yaw, pitch, roll are given in radian. Yaw, pitch, roll are one kind of representation for head orientation. But by using the equation (10), we can also calculate the head orientation vector. But when test its accuracy, it is found that these values are accurate only when the rotation angles are within 20 degrees.

$$\vec{n} = (-2(xz + yw), 2(yz - xw), 2(x^2 + y^2) - 1) \tag{10}$$

In order to improve the accuracy and the detectable range, we directly use high-definitional face points to estimate the head orientation vector. The way we do it is to use a portion of HD face points to construct a plane which is perpendicular to the direction of the head. The points are manually chosen and have been tested on different users to prove its stability. The final plane's direction will be regarded as the head orientation vector and used to approximate the gaze vector (see Figure 17).

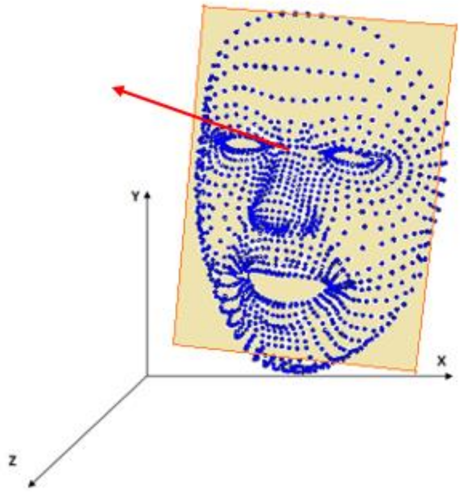

Figure 17: Head Orientation Estimation by constructing the plane (yellow plane). The red arrow is the normal vector of the plane vector.

4. Gaze vector estimation

Head Orientation can be regarded as the first approximation for the gaze detection. The more accurate algorithm will be performed by combing the geometrical and color information of the eyes.

Firstly, the 3D position of the points around the visible eye are used to estimate both eyes' geometrical structures, especially the eye's inner corner's and outer corner's positions. In the estimation, we assumes both eyes to be spheres and, according to the literatures, the distance between inner corner and outer corner can be regarded as the diameter of the eyeballs. After locating the eyeball centers' positions, we define the gaze vector as the vector goes from eyeball centers to the target as the gaze vector and another vector called reference vector which is a unit vector from the center of the eyeball to the center of the visible eye. The next step is to compute the relative angles between these two vectors. Based on the basic physical knowledge, the relative orientation between two arbitrary vectors can be described by three angles in 3D coordinate system. Assuming the gaze vector(x_g, y_g, z_g) and the reference vector(x_r, y_r, z_r), the mathematical expression of the rotation matrix is shown as follows

$$\begin{pmatrix} x_g \\ y_g \\ z_g \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{bmatrix} \begin{bmatrix} \cos(\beta) & 0 & -\sin(\beta) \\ 0 & 1 & 0 \\ \sin(\beta) & 0 & \cos(\beta) \end{bmatrix} \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x_r \\ y_r \\ z_r \end{pmatrix} \quad (11)$$

, where α, β, γ represent the rotation angles around x, y, z axis respectively. After having these three angles, we will start to look at their relationships with the image of the eyes. (See Figure 18). As can be seen from the figure, the relative position between the gaze vector and the reference vector should be consistent with the relative position between the iris and the visible eye center in the eyes images. Our ultimate goal is to use machine learning algorithm to learn this relationship and use it to estimate the gaze with the known reference vector and the eye image. But before that, we need to filter the images and extract some features to make the algorithms more robust.

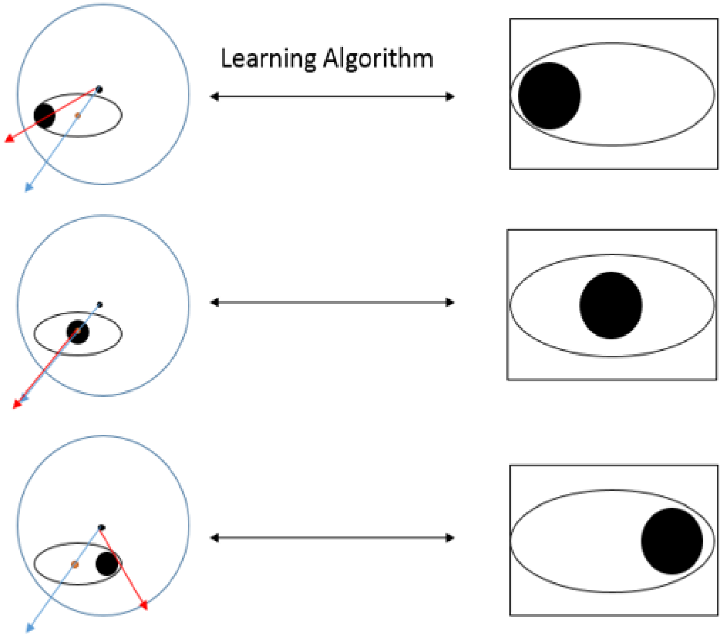

Figure 18: Relationship between geometrical structure and eye images.
(Red arrow: gaze vector, blue arrow: reference vector)

When the resolution of the eye images is very high, it will be very easy to detect the center of the pupil and its relative position to the center of the visible eye. In such cases, we can directly find the gaze vector by using the pure geometrical model. But limited by our Kinect’s camera’s resolution, the compression ratio of the images when they are stored in computer and the distance between the Kinect and the patient, it’s very infeasible (see Figure 19). As can be seen from the picture. it’s very hard to localize the pupil in our eye images So we try to extract some more general features from the eye images.

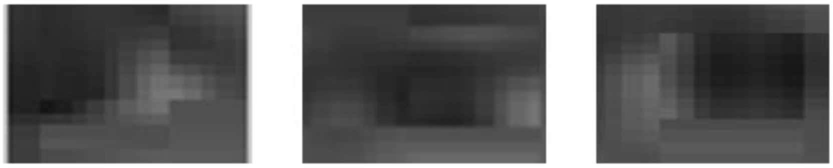

Figure 19: Eye images from our Kinects.

In our model, the 3D eye points will be projected to color image to localize the eyes. A rectangular box will be used to extract the eye images. By filtering and normalizing them, we can resample the images to construct a feature vectors (see Figure 20). The filter we use is the linear regression smoothing filter which smooth data by locally weights. After extracting the feature vectors. We will use the linear regression model to find the relationship between the relative angles and those feature vectors.

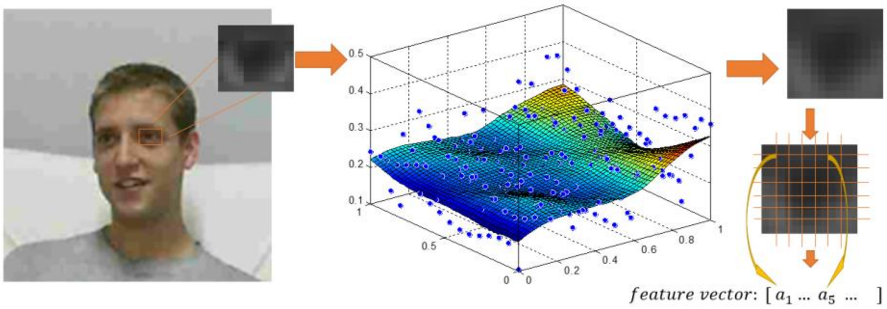

Figure 20: Construction of feature vector from eye images.

The eye images are extracted from the color images and transformed into gray scale. After being normalized and smoothed by linear regression filter, the square eye image will be resampled which is shown by orange grids: the value of each crossing point will be taken out to form a feature vector.

The linear regression model is approach for modeling the relationship between a scalar dependent variable y and one or more explanatory variables, denoted as $\vec{X}$. The model assumes that y can be decided or explained by $\vec{X}$ linearly which means y can be written as a linear function of $\vec{X}$. If we assume $\vec{X}$ is a R^n vector, whose i^{th} component are represented byx_i . The mathematical equation can be shown as follows:

$$y = \sum_{i=0}^{n} a_i\, x_i + \varepsilon \qquad (12)$$

, where ε usually represents the Gaussian noise. In machine learning filed, a training set (known the values of $\vec{X}$ and y) is used to estimate the coefficients a_i in the model and a test set which is independent of training set is employed to validate the model. In our study, the explanatory variables will be the feature vectors and the dependent variables will be three relative angles between gaze vector and reference vector.

In our project, the training dataset and test dataset are captured from five recording sessions. In each session, a ping-pong ball is placed in a different position and the user is asked to keep gazing at the ping-pang ball, but moving his head freely (See Figure 21). At the same time, we will record the color images and HD face points from the Kinect mounted on the footboard. The frame rate for color images is about 30fps and 15fps for the face tracking. By using the training data for which we have known the inputs' and the corresponding outputs' value, we can compute the parameters in the model. Eventually the test data will be used to show the accuracy and robustness of our method.

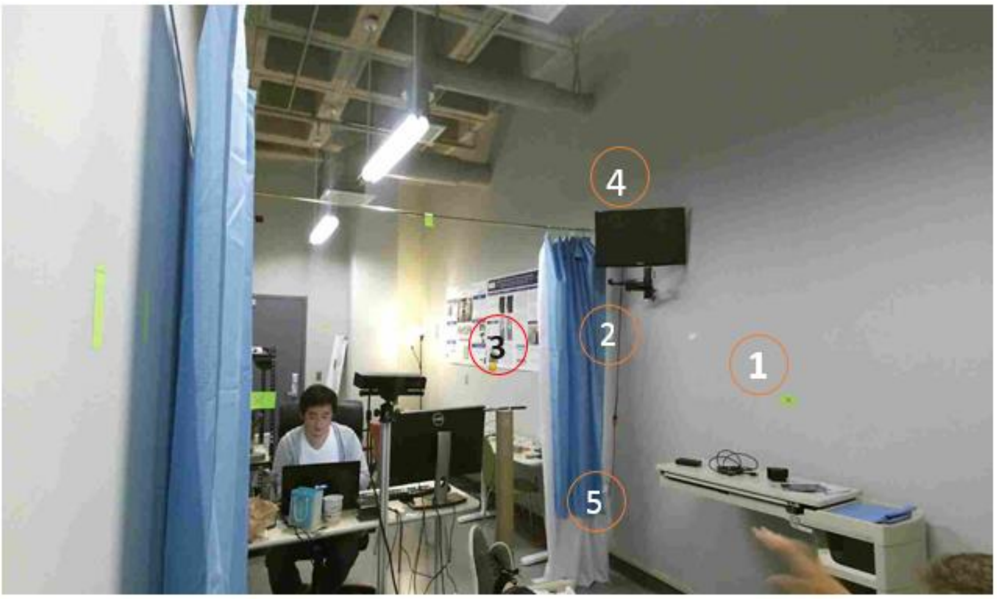

Figure 21: Target positions in different recording sessions.

RESULTS

1. Intrinsic parameters

The intrinsic calibration results for each cameras are shown in table 1.

Table 1: Intrinsic calibration results for each camera

	Focal length (mm)	**Image center (mm)**	**Distortion coefficients**
Kinect 1's IR camera	367.5	[263,209]	[0.056, -0.17, 0.0005,0.001, 0]
Kinect 2's IR camera	359.7	[260,202]	[0.06, -0.18, 0.0009,0.002, 0]
Kinect 1's RGB camera	1059.5	[948,537]	[0.016, -0.011, -0.0004,-0.0006,0]
Kinect 2's RGB camera	1055.7	[957,535]	[0.022, -0.016, -0.0006,-0.0008,0]

As shown in the table, the focal lengths and center coordinates are quite different from each other for different Kinect's depth cameras which shows the necessity of doing the intrinsic calibration for each camera. By using the above parameters and the pinhole model, we can project the points in the 3D camera space to the pixels on 2D color or depth images.

2. Extrinsic parameters

Table 2: Relative geometrical relationship between cameras in one Kinect

	Rotation Matrix	**Translation vector (mm)**
Kinect 1	$\begin{pmatrix} 0.9999 & 0.0047 & 0.0008 \\ -0.0047 & 0.9999 & 0.0018 \\ -0.0008 & -0.0018 & 0.9999 \end{pmatrix}$	$\begin{bmatrix} 52.8 \\ 0.0019 \\ -0.0023 \end{bmatrix}$
Kinect 2	$\begin{pmatrix} 0.9999 & 0.0006 & 0.0005 \\ -0.0006 & 0.9999 & 0.0035 \\ 0.0005 & -0.0035 & 0.9999 \end{pmatrix}$	$\begin{bmatrix} 52.8 \\ 0.0039 \\ -0.024 \end{bmatrix}$

The extrinsic calibration results for the cameras in one Kinect are shown in table 2. The parameters give us relative geometrical relationship between the color camera and IR camera in one Kinect.

The above rotation matrix and translation vector allow us to finish the mapping between the 2D color space and 2D depth space for each Kinect separately. One more thing we need to notice is that, according to the translation vector, the distance between the color camera and IR camera in the same Kinect is about 52.8mm and it is consistent with the measured physical distance (51mm). This gives us more confidence in trusting the extrinsic calibration results.

Table 3 shows the results of the extrinsic calibration between two Kinects. By using rotation matrix and translation vector, we can transform the spherical targets' points from one Kinect's coordinate to the points in the other Kinect's coordinates. The 3D reconstruction of the spherical targets from both Kinects are shown in Figure 22.

Table 3: Relative geometrical relationship between Kinects.

Rotation Matrix	Translation vector (mm)
$\begin{pmatrix} -0.7739 & 0.1174 & -0.6742 \\ -0.1630 & 0.9852 & 0.0036 \\ -0.6436 & -0.1529 & -0.7367 \end{pmatrix}$	$\begin{bmatrix} 1061.3 \\ 338.1 \\ 1623.5 \end{bmatrix}$

As shown in the figure, especially the area within the orange circle, the points from the Kinect mounted on the headboard (red) and that from the Kinect mounted on the footboard clearly show the two sides of the same spherical target. The distance between the target's centers estimated from two Kinects have also been calculated and are shown in Table 4. For different places, the misalignment ranges from 2.8mm to 13mm.

Compared with the geometrical size of the objects in the environment, the error is relative smaller.

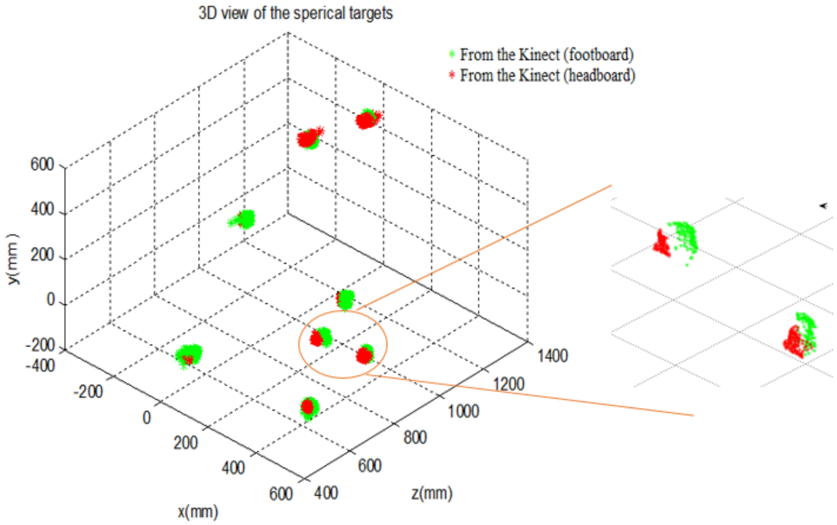

Figure 22: 3D reconstruction of spherical targets from both Kinects

Table 4: Relative distances between ball centers

Position	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Distance (mm)	10.2334	12.9735	11.7640	9.9914	6.4880	9.6813	14.1591	2.8277

3. Head orientation vectors

The top and left side view of the 3D face with estimated head orientation vectors attached have been shown in Figure 23 and 24. In the figures, the green line indicates the vector estimated by our method and the red line is computed from quaternion given by Kinect. Since the head orientation vector is just a rough approximation for gaze vector and used only when the eyes are hard to detect, there is no qualitative methods applied to test its accuracy. But from the figures, especially Figure 24, it does show that the vector estimated by our method is more accurate than that from Kinect.

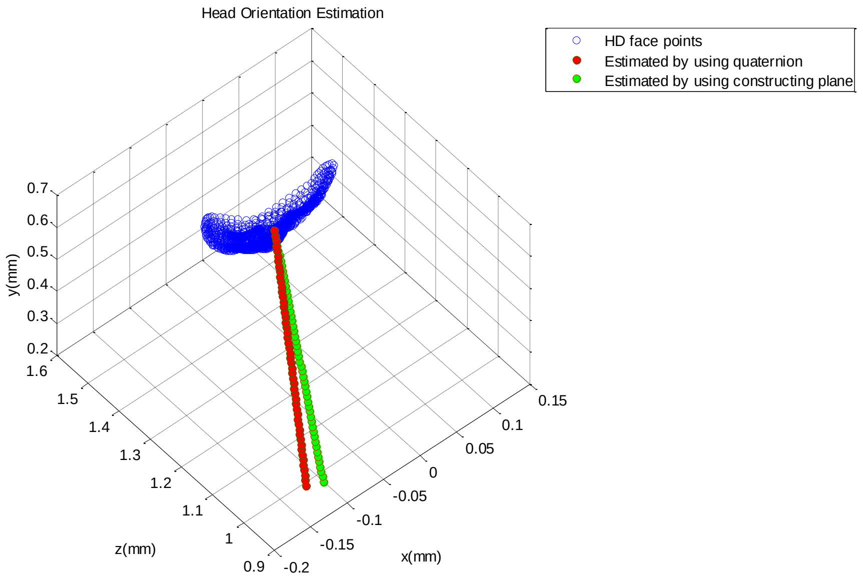

Figure 23: Comparison between the estimated head orientation vectors from two methods
(Top View)

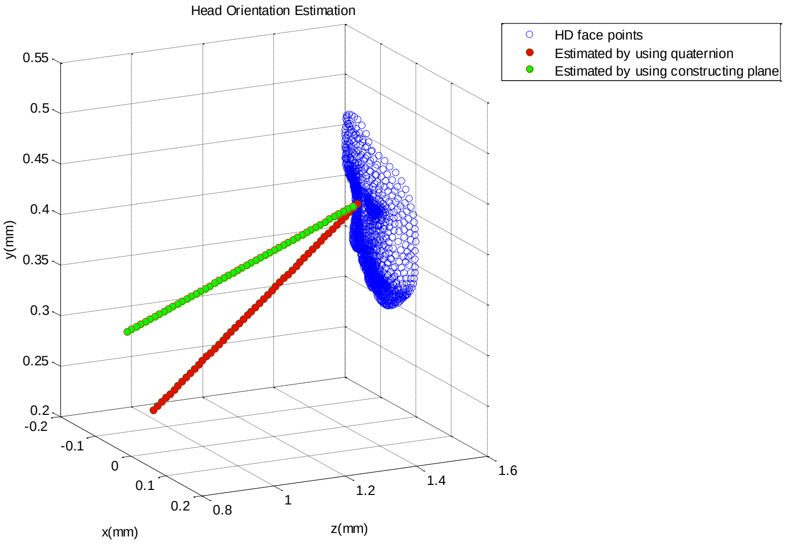

Figure 24: Comparison between the estimated head orientation vectors from two methods
(Left Side View)

4. Stability of eyeball center and visible eye center

The estimation of eyeball centers and visible eye centers play a very important role in our project, since it directly decides the accuracy of the reference vector and gaze vector. So it’s necessary to detect its stability. Both centers’ 3D position can change with head motion, but their relative points to the other points on the face should be a constant across the time. So in Figure 25, we will test how the relative distances between eyeball centers, visible eye centers and nose tip change with time.

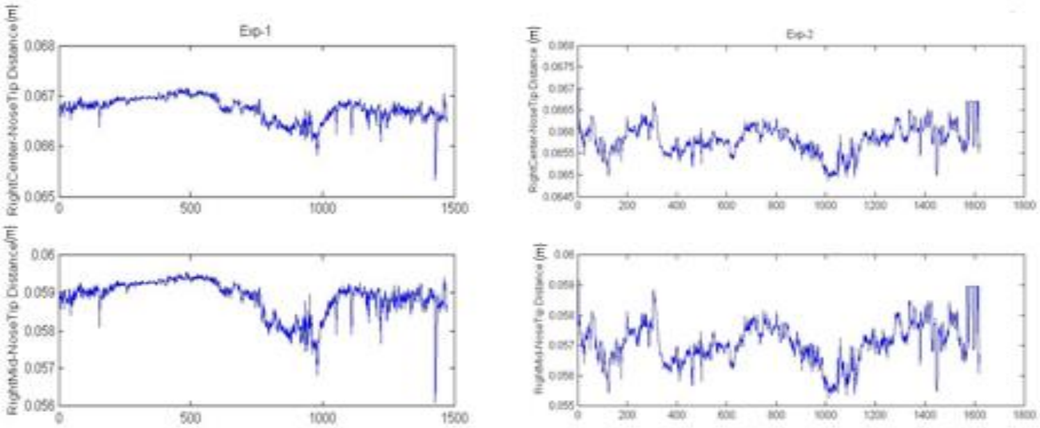

Figure 25: Stability analysis

As shown in the pictures on the top left, the mean distance between right eyeball center and nose tip is around 0.0665m in the first session and the standard deviation is less than 5mm.

By comparing the top two subplots, it also shows that the relative distance in different experiments are pretty consistent (around 0.066 m). Both results indicate that the estimation of eyeball centers are very stable and the same conclusion can be drawn for the relative distance between visible eye centers and the nose tip according to the bottom plots in Figure 25.

5. Angle distribution

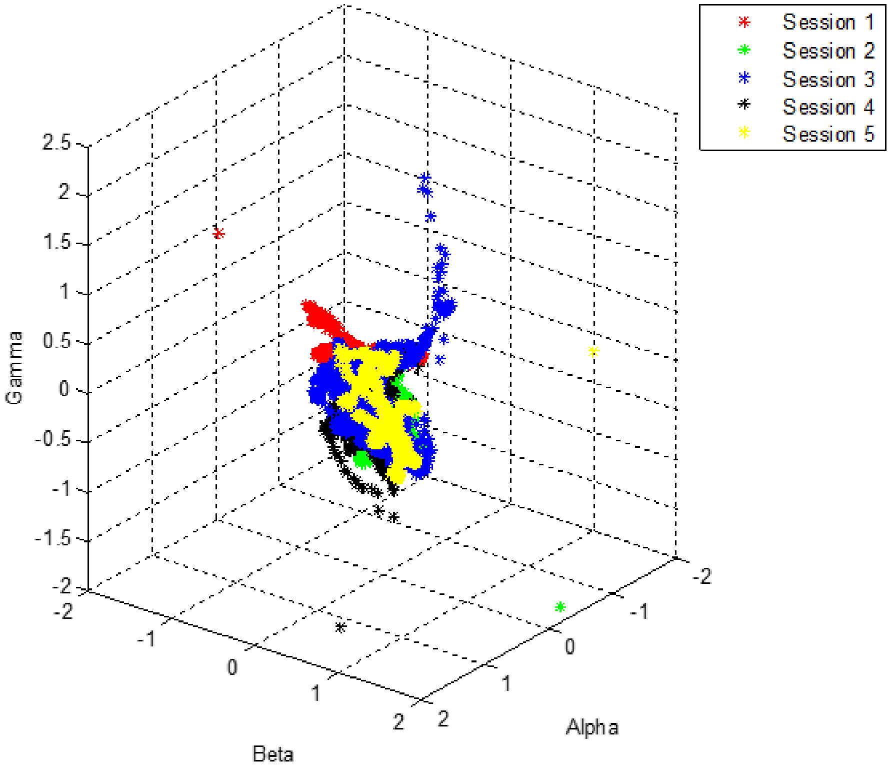

Figure 26: Angle distribution

The gaze vectors and corresponding reference vectors are extracted from each frame in this dataset and the relative angles are computed. The angle distribution are shown in Figure 26 and different colors indicates the data from different recording sessions. As can been seen from the picture, all of the relative angles cluster together and the reason is that, during each recording session, the user can gaze at the target in different ways by moving the head and pupils. So each recording session have almost covered all of the possible relative angles.

6. Predicted gaze vectors:

After decomposing the rotation matrix and acquiring the three relative angles between reference vectors and gaze vectors. We can start the gaze vector estimation and test its accuracy.

Table 5: Confusion matrix for the type 1 Gaze Prediction

	TARGET 1	TARGET 2	TARGET 3	TARGET 4	TARGET 5
SESSION 1	0.7574	0.0163	0.0860	0.1241	0.0163
SESSION 2	0.2136	0.5502	0.0712	0.1438	0.0213
SESSION 3	0.1810	0.0497	0.6572	0.0746	0.0375
SESSION 4	0.3343	0.0698	0.2193	0.1490	0.2276
SESSION 5	0.1714	0.2220	0.1412	0.0081	0.4573

The prediction is divided into two categories. Firstly, we will use the data from the four sessions as training dataset to calculate the parameters in the linear regression model and test the model by the data from the remaining session. For each frame in the remaining session, the feature vector from the eye image will be put into model to estimate the relative angles and the angles will be used to rotate the reference vector to get the gaze vector. Estimated gaze vectors (from eyeball center to five targets) will be used to find the closet target. The accuracy is shown in table 4, the i^{th} row indicates the results for the prediction that using i^{th} session as testing dataset. For example, for the first row, the training datasets are from session 2-5 and the 1st dataset is used to test the performance, the results that, for around 75% of the whole time, the model can succeed predicting that the patient is looking at the first target. The goal of this prediction is to test the algorithm's performance in the condition that is not included in the training session.

The second prediction process uses 20% of the dataset from fives experiments to train the model and test the accuracy by using the remaining 80% dataset. The way of estimating the target of gaze is the same as the one used in the above. The confusion matrix are shown in Table 5. As we can see from the table, for target 1, 2, 3, the accuracy of detection can achieve above 75%. For the fourth and fifth targets (see Figure 21), the low accuracy is caused the poor eye images captured. When the patient is looking up or down, it's much harder to see the complete eyes. But comparing table 5 and 6, it shows that, if we try to cover as much possible targets as possible in our training session, the prediction accuracy can be increased by around 10%.

Table 6: Confusion matrix for the type 2 Gaze Prediction

	TARGET 1	TARGET 2	TARGET 3	TARGET 4	TARGET 5
SESSION 1	0.8632	0.0198	0.0615	0.0361	0.0193
SESSION 2	0.1012	0.8202	0.0532	0.0097	0.0157
SESSION 3	0.1981	0.0045	0.7571	0.0085	0.0318
SESSION 4	0.1313	0.0413	0.1504	0.6691	0.0079
SESSION 5	0.1435	0.1772	0.1168	0.0012	0.5613

DISCUSSION

Recently, some intrusive commercial hardware have been used to track gaze points, like Pupil, Tobii, but, in the clinical environment, mounting an eye- tracking system on patients' heads or mounting a camera close to patients may influence the clinic flow, thus making them infeasible. Our work is trying to develop a system which minimizes the intrusiveness and can be employed in clinical environment, even though our system may not be as accurate as the current commercial system. Compared with the traditional methods of gaze detecting, our method allows a wider detection range and loose the limitation on users' movements. Traditional methods usually requires a distance around 0.5m between the user and the camera to acquire a high resolution eye images, but the distance can be around 2m in our project, while achieving a accuracy around 70%.

In the clinic environment, one of the tradeoff we need to consider is the memory used to store the data and the compression ratio of the images. Currently, we store the data as JPEG format, in order to save more spaces, we compress the picture by a ratio around 15%. When the ratio goes higher, the images can become less and less blurred. By employing the images with higher compression ratio and finding more robust features, we believe it can further improve the accuracy of our method.

One more way to improve the performance is to filter out the 'bad' training data. Now the raw data is been used to training the linear regression model, but, if we can filter out the frames in which the patient is blinking or closing his eyes, we believe it will also help improve current method's performance.

Now all the results we get are all from the offline analysis which means we collect the data firstly and then use Matlab to process it. But, for the purpose of the clinical use, we will need to test the algorithm on real-time system to compute its latency and flexibility in the future.

REFERENCES

1. Reinkensmeyer DJ, Kahn LE, Averbuch M, McKenna-Cole AN, Schmit BD, Rymer WZ. Understanding and treating arm movement impairment after chronic brain injury: progress with the ARM Guide. *J Rehabil Res* Dev. 2000;37:653-662.

2. Gebruers N, Vanroy C, Truijen S, Engelborghs S, De Deyn PP. Monitoring of physical activity after stroke: a systematic review of accelerometrybased measures. *Arch Phys Med Rehab* 2010; 91: 288–97.

3. Pierce K, Conant D, Hazin R, Stoner R, Desmond J. Preference for Geometric Patterns Early in Life as a Risk Factor for Autism. *Arch Gen Psychiatry*. 2011;68(1).

4. Osterling JA, Dawson G, Munson JA. Early recognition of 1-year-old infants with autism spectrum disorder versus mental retardation. *Dev Psychopathol*. 2002; 14(2):239-251.

5. Baranek GT. Autism during infancy: a retrospective video analysis of sensorymotor and social behaviors at 9-12 months of age. *J Autism Dev Disord*. 1999;29(3):213-224.

6. Mirenda P, Smith IM, Vaillancourt T, et al. Validating the Repetitive Behavior ScaleRevised in young children with autism spectrum disorder. *J Autism Dev Disord*. 2010;40:1521–1530.

7. Sturm, V. E., McCarthy, M. E., Yun, I., Madan, A., Yuan, J. W., Holley, S. R., Levenson, R. W. Mutual gaze in Alzheimer’s disease, frontotemporal and semantic dementia couples. *Social Cognitive and Affective Neuroscience*, 6, 2000: 359 –367.

8. A.A. Ghazanfar, *et al.* Multisensory integration of dynamic faces and voices in rhesus monkey auditory cortex. *J. Neurosci*, 25 (2005): 5004–5012

9. Zion Golumbic, E.M., Cogan, G.B., Schroeder, C.E., and Poeppel, D. Visual input enhances selective speech envelope tracking in auditory cortex at a ‘‘cocktail party’’. *J. Neurosci*. 2003; 33, 1417–1426.

10. K. N. Kim, and R. S. Ramakrishna, Vision-based eye-gaze tracking for human computer interface, *IEEE International Conference on Systems, Man and Cybernetics*, 1999: 324-329.

11. Hutchinson, T. E., White, K. P., Martin, W. N., Reichert, K. C., & Frey, L. A. Human–computer interaction using eye-gaze input. *IEEE Transactions on Systems, Man, and Cybernetics*, 1989; 19: 1527–1534.

12. Hansen, D. and Ji, Q. In the eye of the beholder: A survey of models for eyes and gaze. *IEEE Trans. on PAMI*, 32(3) 2010 :478–500

13. D.W. Hansen and A.E.C. Pece. Eye Tracking in the Wild, *Computer Vision and Image Understanding*, vol. 98, no. 1, pp. 182- 210, Apr. 2005.

14. G.C. Feng and P.C. Yuen, Variance Projection Function and Its Application to Eye Detection for Human Face Recognition, *Int'l J. Computer Vision*, vol. 19, pp. 899-906, 1998.

15. R. Herpers, M. Michaelis, K. Lichtenauer, and G. Sommer, "Edge and Keypoint Detection in Facial Regions," Proc. Int'l Conf. *Automatic Face and Gesture-Recognition*, 1996: 212-217.

16. S. Sirohey, A. Rosenfeld, and Z. Duric, A Method of Detecting and Tracking Irises and Eyelids in Video, *Pattern Recognition*, vol. 35, no. 6, pp. 1389-1401, June 2002

17. Y. Ebisawa, Improved Video-Based Eye-Gaze Detection Method, *IEEE Trans. Instrumentation and Measurement*, vol. 47, no. 2, pp. 948-955, Aug. 1998.

18. Y. Ebisawa and S. Satoh, Effectiveness of Pupil Area Detection Technique Using Two Light Sources and Image Difference Method, *Proc. 15th Ann. Int'l Conf. IEEE Eng. in Medicine and Biology Soc.*, 1993: 1268-1269.

19. Y. Ebisawa, Realtime 3D Position Detection of Human Pupil, Proc. *2004 IEEE Symp. Virtual Environments, Human-Computer Interfaces and Measurement Systems*, 2004: 8-12.

20. Model, D. and Eizenman, M. User-calibration-free remote gaze estimation system. *In Proceedings of symposium on eyetracking research & applications*, 2010：29–36 NY.

21. D. Beymer and M. Flickner, "Eye Gaze Tracking Using an Active Stereo Head," Proc. IEEE Conf. Computer Vision and Pattern Recognition, vol. II, pp. 451-458, 2003

22. P. Zhang, Z. Wang, S. Zheng and X. Gu "A design and research of eye gaze tracking system based on stereovision", *Emerging Intell. Comput. Technol. Applicat., Lecture Notes Comput. Sci.*, vol. 5754, no. 4, pp.278 -286 2009

23. L. Feng, Y. Sugano, O. Takahiro, and Y. Sato. Inferring Human Gaze from Appearance via Adaptive Linear Regression. *In ICCV: International Conference on Computer Vision*, Barcelona, Spain, 2011.

24. C.H. Morimoto, D. Koons, A. Amir, and M. Flickner, "Pupil Detection and Tracking Using Multiple Light Sources, *Image and Vision Computing*, vol. 18, no. 4, pp. 331-335, 2000.

25. Zhengyou Zhang, Microsoft kinect sensor and its effect, *MultiMedia, IEEE*, vol. 19, no. 2 (2012): 4 –10.

26. Microsoft Kinect Coordinate mapping: https://msdn.microsoft.com/en-us/library/Dn785530.aspx

27. J. Shotton, A. W. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, Real-time human pose recognition in parts from single depth images, *CVPR*(2011): 1297–1304.

28. Microsoft Kinect HDFacepoints Enumeration: https://msdn.microsoft.com/enus/library/microsoft.kinect.face.highdetailfacepoints.aspx

29. J. Smisek, M. Jancosek, and T. Pajdla, 3-D with Kinect, Proc. *IEEE ICCV Workshops* (2011): 1154–1160.

30.] T. Stoyanov, A. Louloudi, H. Andreasson, and A. Lilienthal, Comparative evaluation of range sensor accuracy in indoor environments, *in Proc. Eur. Conf. Mobile Robots* (2011) :19–24.

31. 7. K. Khoshelham and S. Elberink, Accuracy and resolution of kinect depth data for indoor mapping applications, *Sensors*, vol. 12, no. 2 (2012): 1437–1454.

32. J. Han, L. Shao, D. Xu, and J. Shotton, Enhanced computer vision with microsoft kinect sensor: A review, *IEEE Trans. Cybern.*, vol. 43, no. 5 (2013): 1318–1334.

33. S. Obdržálek, G. Kurillo, F. Ofli, R. Bajacsy, E. Seto, H. Jimison, *et al.* Accuracy and robustness of Kinect pose estimation in the context of coaching of elderly population. Engineering in medicine and biology society (EMBC), *2012 annual international conference of the IEEE*, August 28 2012–September 1 2012 (2012): 1188–1193.

34. Weibel N, Rick S, Emmenegger C, Ashfaq S, Calvitti A, Agha Z. Lab-in-a-box: semi-automatic tracking of activity in the medical office. *Pers Ubiquit Comput* (2014):1–18.

35. Friesen, W.; Ekman, P. (1983). EMFACS-7: Emotional Facial Action Coding System. *Unpublished manual*, University of California, California.

36. Camera Models and Imaging: http://www.comp.nus.edu.sg/~cs4243/lecture/camera.pdf

37. Camera Calibration Toolbox for Matlab®: http://www.vision.caltech.edu/bouguetj/calib_doc/

38. E. Shen, P. K. Carr, P. Thomas, and R. Hornsey, Non-planar target for multi-camera network calibration, in *Proc. IEEE Sensors*, Oct. 2009, pp. 1410–1414.