Eye Tracker Accuracy:
Quantitative Evaluation of the Invisible Eye Center Location

Stephan Wyder* and Philippe C. Cattin†
Department of Biomedical Engineering, University of Basel, Allschwil, Switzerland
(Dated: February 1, 2017)

Purpose. We present a new method to evaluate the accuracy of an eye tracker based eye localization system. Measuring the accuracy of an eye tracker's primary intention, the estimated point of gaze, is usually done with volunteers and a set of fixation points used as ground truth. However, verifying the accuracy of the location estimate of a volunteer's eye center in 3D space is not easily possible. This is because the eye center is an intangible point hidden by the iris.

Methods. We evaluate the eye location accuracy by using an eye phantom instead of eyes of volunteers. For this, we developed a testing stage with a realistic artificial eye and a corresponding kinematic model, which we trained with μCT data. This enables us to precisely evaluate the eye location estimate of an eye tracker.

Results. We show that the proposed testing stage with the corresponding kinematic model is suitable for such a validation. Further, we evaluate a particular eye tracker based navigation system and show that this system is able to successfully determine the eye center with sub-millimeter accuracy.

Conclusions. We show the suitability of the evaluated eye tracker for eye interventions, using the proposed testing stage and the corresponding kinematic model. The results further enable specific enhancement of the navigation system to potentially get even better results.

I. INTRODUCTION

Eye tracking devices, also known as eye- or gaze trackers are used to monitor eye movement. An eye tracker is usually used to determine a person's point of gaze. In market research, for instance, a wearable, video based eye tracking system can be used to uncover which product on which shelf is attracted by a test person. Certainly, there exist other constructions of eye trackers (e.g. desktop or embedded devices) and many other eye tracking applications (e.g. in usability testing or in automotive industry) [1, 2]. Different physical principles might be behind an eye tracker, depending on the application [3]. Video based eye trackers are the most widely used devices, because of their simplicity and the wide applicability.

In recent research, eye trackers are also used in navigation systems for computer assisted eye interventions [4–6]. In these cases, the eye tracker is used to estimate the 3D-location of the patient's eye, that is the eye center and orientation. We define the eye center as the center of corneal curvature. This can be useful to align an eye for an ophthalmic examination or treatment. Furthermore, the point of gaze, estimated by the eye tracker, is automatically monitored to interrupt an examination or treatment in case of sudden eye motion.

Using an eye tracker for medical interventions demands high system accuracy. This may decide between success or failure of an intervention because of the close proximity of critical structures within the eye. For instance, an eye localization accuracy below 1 mm is required, when an eye tracker is used to target intraocular tumors.

The demand for accurate eye tracking systems also raises the need for reliable accuracy measurement methods. Accuracy measurements are crucial for the development of an eye tracking system and also for the performance specification of the device.

Conventionally, eye tracker accuracy is evaluated with volunteers, who have to focus on certain fixation points located at well-known positions. The accuracy is then given by the deviations between the true fixation point locations and the point of gaze estimates of the eye tracker. As straightforward as this evaluation can be performed on the one hand, as difficult it is to see what parts of the system contribute to a certain error on the other hand. Testing this way does not enable us validating the accuracy of an intermediate product of the eye tracking pipeline, as for instance the eye center location. Furthermore, this validation method obviously depends on the cooperation of the volunteers. Hence, measuring the accuracy with an eye phantom seems to be the ideal complement for a thorough eye tracker evaluation.

Already Via et al. [4] used an eye phantom to asses the accuracy of an eye tracking system. However, details about the exact procedure remain partially unclear. Furthermore, it is not clear how realistic their eye phantom is. Also Świrski and Dodgson recognized the lack of a comprehensive evaluation method to test and improve the individual parts of an eye tracking system. They propose completely synthetic eye data [7] for accuracy and precision evaluation of eye tracking algorithms.

Compared to Via et al. [4], we build up our ground truth data using μCT-measurements to get highly accurate absolute eye center locations. In contrast to Świrski and Dodgson [7], we do not only evaluate the algorithm, but we validate the complete eye tracking system, including the whole optical path and the external referencing

* stephan.wyder@unibas.ch
† philippe.cattin@unibas.ch

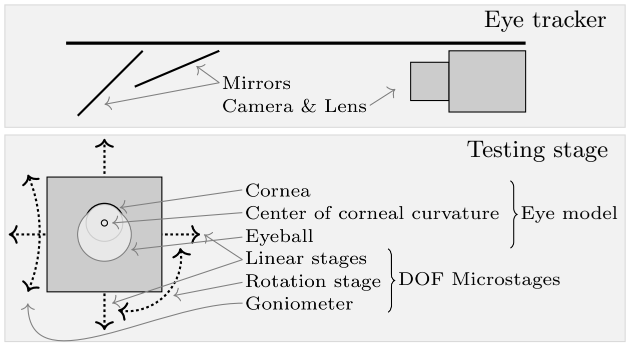


Figure 1: Eye tracker and testing stage (topdown view)

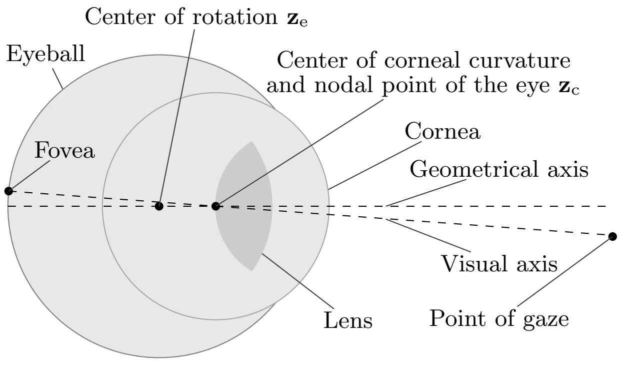

Center of rotation $\mathbf{z}_e$
Eyeball
Center of corneal curvature and nodal point of the eye $\mathbf{z}_c$
Cornea
Fovea
Geometrical axis
Visual axis
Lens
Point of gaze

Figure 2: Typical eye model used by eye trackers

to a medical device. The evaluation of such an eye localization system involving eye tracker hardware and its environment cannot be done with rendered eye images. Neither can it be done with volunteer tests, because it is not possible to accurately measure the 3D location of a volunteers invisible eye center.

Accurate ground truth data is required for the accuracy evaluation of an eye localization system. We propose a procedure to fill this gap by providing accurate 3D-locations of the invisible and intangible eye center.

The basis is formed by a testing stage with four degrees of freedom (4 DOF), a mounted artificial glass eye, and an attached, black and white checkerboard pattern for external referencing. The testing stage enables us to move the whole eye forth and back and sidewards (by two linear stages). Additionally, the testing stage enables us to rotate the eye around two axes (by a rotation stage and a goniometer), in order to simulate an arbitrary line of sight. We built the testing stage and trained the parameters of its kinematic model with μCT-data (i.e. high precision 3D volumetric data) acquired of the testing stage in several different configurations (i.e. eye positions and orientations). The μCT-data provides us with accurate information about the location and the geometry of the eye and the checkerboard. Figure 1 illustrates the two involved parts, the eye tracker we want to evaluate and the proposed testing stage to accomplish the evaluation.

Having the testing stage ready and the kinematic model trained, we position and orient the artificial eye in known locations and compare this against the eye center locations predicted by the eye tracker. The eye center locations are given by the centers of corneal curvature (i.e. center of a cornea best fit sphere). The trained kinematic model provides us with exactly the same point in a common coordinate system (CS), which is also accessible by the eye tracker. This enables us to compare the eye location estimates of the eye tracker with the ground truth data, given by the testing stage model.

With the proposed testing stage, it is possible to quantitatively evaluate the performance of any 3D model based eye tracker. Using this method, we show that a particular eye tracking system [6] estimates the eye center location with sub-millimeter accuracy.

We describe in the following sections our proposed method for the accuracy evaluation and the results achieved when testing a particular eye tracker [6].

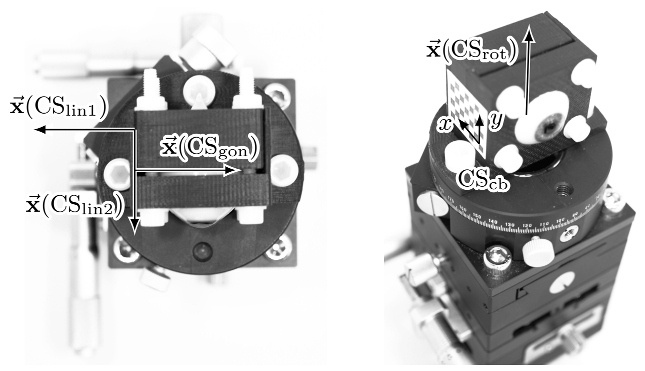
II. METHODS

We propose a custom-built hardware testing stage and an appropriate kinematic model with its calibration, to evaluate the accuracy of the center location of the corneal curvature, estimated by an eye tracker. This section consists of three parts. First, we give an insight into a typical eye tracking model based on 3D ray tracing. Second, we present the testing stage hardware with its components. The testing stage hardware enables us to position and orient the embedded artificial eye such that the eye tracker can perform its intended measurements. The testing stage hardware basically replaces the testing volunteer, with the advantage of having the exact position of the eye (i.e. ground truth). Third, we present the corresponding testing stage model, which we parametrize, train and validate with μCT data. The kinematic model enables us to determine the exact glass eye position in every possible testing stage configuration with sub-millimeter accuracy.

Consequently, this enables us to test an eye tracker on the artificial eye prothesis with several different eye positions and orientations. The 3D eye location estimate of the eye tracker can then be compared to the ground truth data of the testing stage model, which is configured according the status of the testing stage.

A. Eye Tracker Model

The complexity of existing eye tracking models vary. The eye is often modeled with two spheres. Figure 2 illustrates such a typical eye model. One sphere represents the eyeball, its center consequently corresponds to the rotation center of the eye. The second sphere, the sphere cap respectively, represents the cornea. The center of the corneal curvature corresponds to the nodal point of


$\vec{\mathbf{x}}(\mathrm{CS_{lin1}})$
$\vec{\mathbf{x}}(\mathrm{CS_{gon}})$
$\vec{\mathbf{x}}(\mathrm{CS_{lin2}})$
$\vec{\mathbf{x}}(\mathrm{CS_{rot}})$
x
y
$\mathrm{CS_{cb}}$

(a) Top-down view (b) Bird's-eye view

Figure 3: Testing stage with the glass eye, its holder with the checkerboard and the microstages

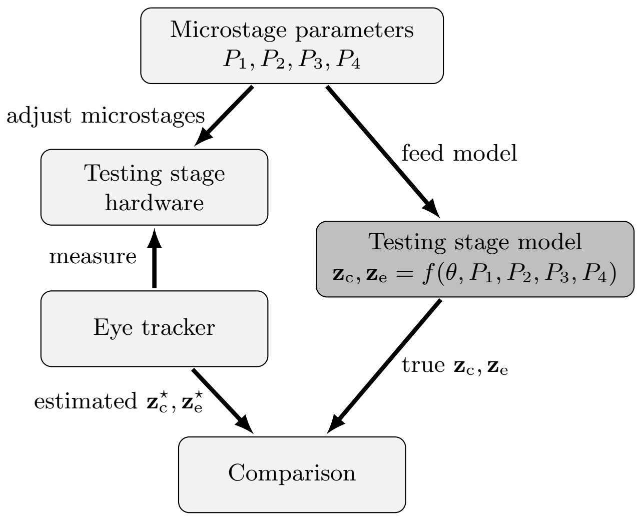

Microstage parameters P_1, P_2, P_3, P_4
adjust microstages
feed model
Testing stage hardware
measure
Testing stage model $\mathbf{z}_\mathrm{c}, \mathbf{z}_\mathrm{e} = f(\theta, P_1, P_2, P_3, P_4)$
Eye tracker
true $\mathbf{z}_\mathrm{c}, \mathbf{z}_\mathrm{e}$
estimated $\mathbf{z}_\mathrm{c}^\star, \mathbf{z}_\mathrm{e}^\star$
Comparison

Figure 4: Concept of testing stage: Comparison of the eye center location $\mathbf{z}_\mathrm{c}^\star$ estimated by the eye tracker with the ground truth $\mathbf{z}_\mathrm{c}$

the eye, where the optical rays cross, before they hit the retina [8].

The two sphere centers define the geometrical axis of the eye. Hence, the orientation of an eye in space can be determined by the geometrical axis. The location of the eye (i.e. eye center) is given by the center of the corneal curvature, which lies on the mentioned, geometrical axis and is an integral part of most of the 3D model based eye trackers.

The fovea (point of sharpest vision) is located on the retina (backside of the eyeball) but is not in line with the geometrical axis. A point we focus on with our eye gets imaged on the fovea. That is why also the visual axis plays an important role in such a model. The visual axis connects the fovea with the nodal point of the eye and the point of gaze. The angle between visual axis and geometrical axis has to be calibrated per patient.

The eye tracker [5, 6] which we test with the proposed testing stage is based on the model of E. D. Guestrin and M. Eizenman [8].

B. Testing Stage Hardware

The testing stage we developed consists of a translation stage with two axes with parameters P_1 and P_2 (*OptoSigma TADC-652WS25-M6*), a goniometer stage with parameter P_3 (*OptoSigma GOH-65A50-M6*), and a rotation stage with parameter P_4 (*OptoSigma KSW-656-M6*). The variables P_1, P_2, P_3, and P_4 represent the values, which are set for the corresponding microstages. The linear stages have a vernier scale included enabling to measure with a precision of $10\,\mu$m. The rotation stage and goniometer also contain a vernier scale enabling us to measure with a precision of angular minutes. To simulate the human eye, we use a handcrafted eye prosthesis made from glass by the *Swiss Institute For Artificial Eyes, Lucerne, Switzerland*. To interface the artificial eye with the stages we designed a rigid and robust eye holder. Since the eye prosthesis does a priori not have an exactly known geometry, we made a 3D scan of it with a μCT device (*GE phoenix nanotom m*). We segmented the surface of the eye with *Fiji*, an image processing package [9]. We afterwards used *Blender*, an open source 3D creation suite (http://www.blender.org), to design a holder accurately interfacing the eye with the stages. The holder additionally contains a black and white checkerboard stuck on its side. The checkerboard is printed with an off-the-shelf laser printer, which contains toner visible in the μCT. The stages are serially mounted and on top of them is the eye holder, which was printed on a *Stratasys Fortus 250mc* 3D printer. The testing stage is shown in Figure 3.

C. Testing Stage Kinematic Model

The aim of the kinematic model is to determine the exact center location of the corneal curvature $\mathbf{z}_\mathrm{c}$ for a certain testing stage configuration (P_1, P_2, P_3, P_4) and to transform the coordinates to a common coordinate system.

The internal model parameters θ, that have to be trained, basically consist of six right-handed coordinate systems (CS): $\mathrm{CS_{vol}}$ is the common CS for all μCT - volumes, $\mathrm{CS_{lin1}}$, $\mathrm{CS_{lin2}}$, $\mathrm{CS_{gon}}$, and $\mathrm{CS_{rot}}$ correspond to their appropriate microstage and $\mathrm{CS_{cb}}$ is the checkerboard CS. $\mathrm{CS_{vol}}$ can be seen as the CS for model input data, whereas $\mathrm{CS_{cb}}$ is the CS for the output data. $\mathrm{CS_{cb}}$ is accessible by the eye tracker and the testing stage. Additionally, θ contains $\hat{\mathbf{z}}_\mathrm{c}$ and $\hat{\mathbf{z}}_\mathrm{e}$, the center locations and the radii of the cornea and the eyeball, yet unaffected by P_1, P_2, P_3, P_4 (neutral position).

Figure 4 illustrates the role of the testing stage model within our contribution.

The origins of the CSs and the corresponding orientations are defined based on the acquired μCT data. We adjust a few positions of each individual microstage and acquire a μCT volume for each configuration. This enables us to train the internal kinematic model parameters θ.

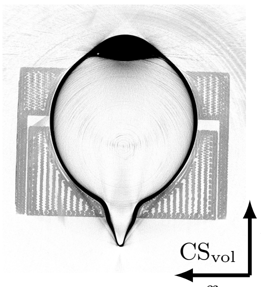
***μCT* Data Acuisition.** As seen in Tab. I, we acquired 15 μCT-volumes, which help us to define the mentioned internal model parameters θ. Furthermore, we used some μCT measurements to test the integrity of our kinematic model. The table shows the number (identifier) of the measurement (#), the state of the individual microstages during a certain scan and the type of the measurement ($\star$).

Table I: μCT-data acquisition plan

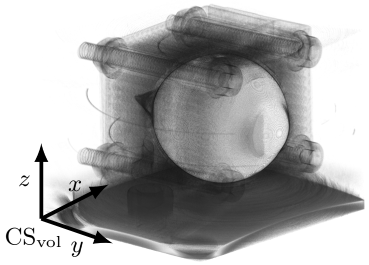
	Stages				
	P_1	P_2	P_3	P_4	
#	linear 1	linear 2	gonio.	rotation	$\star$
1	0 mm	0 mm	0°	0°	1,3
2	−7.5 mm	0 mm	0°	0°	3
3	7.5 mm	0 mm	0°	0°	5
4	0 mm	0 mm	0°	0°	1,3
5	0 mm	−7.5 mm	0°	0°	3
6	0 mm	7.5 mm	0°	0°	5
7	0 mm	0 mm	0°	0°	1,4
8	0 mm	0 mm	−15°	0°	4
9	0 mm	0 mm	8°	0°	5
10	0 mm	0 mm	15°	0°	4
11	0 mm	0 mm	0°	0°	2,4
12	0 mm	0 mm	0°	−30°	4
13	0 mm	0 mm	0°	15°	5
14	0 mm	0 mm	0°	30°	4
15	0 mm	0 mm	0°	0°	2

$\star$1 corresponds to training scans, where the microstages are in neutral position. For testing, we use $\star$2 scans, which also correspond to neutral position scans. $\star$3 scans are used to train the linear stages. $\star$4 scans are used to train the rotation and goniometer stages. $\star$5 scans are used to test the kinematic model accuracy of the individual degrees of freedom.

To acquire the required data, we use the *GE phoenix nanotom m* μCT device. In order to get a good contrast for the glass eye surface as well as for the checkerboard pattern in the acquired μCT data, we set the voltage to 50 kV and the current to 310 μA. To limit the required overall acquisition time for the 15 scans, we used a so called *fast scan mode*, for which the specimen in the nanotom rotates continuously 360° during a defined time (in our case 20 min). These settings result in 1599 projections (3072 px × 2400 px), exposed with 750 ms each. The isotropic voxels have the side length of 25 µm. The resulting reconstructions (3D volumes) of the projections are cropped to the content of importance and have the



(a) Grayscale inverted slice along z-axis: eye and 3D printed holder



(b) 3D rendering: glass eye, holder and plastic screws

Figure 5: Visualized μCT data ($\mathrm{CS_{vol}}$) acquired with *GE phoenix nanotom m*

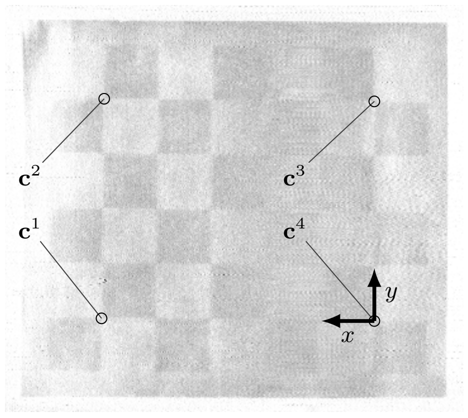


Figure 6: Checkerboard corners ($\mathbf{c}^k$) and $\mathrm{CS_{cb}}$ as seen in the μCT data

size of 2100 px × 1900 px × 1700 px. Additionally, we reduce the grayscale depth from 16 bit to 8 bit by linearly mapping the grayscale-values between 23'000 and 35'000 to the range between 0 and 255, such that both, the eye surfaces as well as the checkerboards are well visible. The whole process of reducing the volume dimensions and the grayscale depth is mainly required to reduce the amount of data for further processing. The size of one final volume is still 6.8 GB.

Figure 5 illustrates the data acquired with the μCT. Figure 5a shows one slice perpendicular to the z-axis and Figure 5b shows a volume rendering of a μCT scan. Both figures illustrate also the location and orientation of the $\mathrm{CS_{vol}}$.

In order to be able to train our kinematic model with the acquired data, we first need to segment the required features.

***μCT* Data Segemention.** We extract two different types of features from the acquired volumes, four checkerboard corners ($\mathbf{c}^k$, where $k \in \{1, 2, 3, 4\}$), as they are visible in Figure 6, and the surface of the glass eye (the black contour visible in Figure 5a).

To train the kinematic model we need to have the corner point coordinates as they are visualized in Figure 6 for all 15 data volumes. We extract the coordinates of $\mathbf{c}^k$ by hand using *Fiji's* "Big Data Viewer". This plugin enables to visualize a slice with an arbitrary orientation and to show the 3D coordinates of a given voxel. This results in $15 * 4$ 3D coordinates in CS_{vol} coordinate system.

The following procedure describes the extraction of the glass eye for all volumes in neutral configuration ($\star 1$ and $\star 2$, see Tab. I). We process the volumes (thresholding and surface extraction) again by using *Fiji* [9]. The edge of the eye is segmented by applying a threshold of 115, which is an experimentally found value. Afterwards we extract the surface from the segmented eye using the marching cubes method (using the "3D Viewer" plugin). The surface mesh can be exported as STL file directly with this plugin. This results in a mesh basically consisting of an outer and an inner surface of the glass eye along with some unwanted holes and additional artifacts.

To clean up the geometry we import the mesh into *Blender*. Within *Blender* we first create several objects by separating the imported mesh by loose parts. All but the biggest part (the eye) can be deleted. To save later processing time, we apply a mesh decimation. We extract the cornea and the eyeball separately to individually fit a sphere afterwards. The cleaned cornea- and eyeball-mesh are exported again as STL for all 5 mentioned volumes.

After μCT data acquisition and segmentation we end up with four 3D coordinates each ($\mathbf{c}^k$, $k \in \{1,2,3,4\}$) for all 15 volumes. In addition we have an extracted cornea and an eyeball mesh for five of the 15 volumes (where $\star 1$ and $\star 2$).

Kinematic Model Calibration. All data used as input (checkerboard corner points, cornea mesh, eyeball mesh) to train the internal model parameters θ are in the right-handed CS_{vol} coordinate system and are given in voxel. We also express the other coordinate systems relative to CS_{vol}.

Let $\mathbf{c}_j^k \in \mathbb{R}^3$ be a 3D vector in CS_{vol} representing a checkerboard corner point, where $k \in \{1,2,3,4\}$ encodes the checkerboard corner point number and $j \in \{1,2,3,...,15\}$ encodes the number of the measurement (#).

Let G_p^k be a group of $\mathbf{c}_j^k$, where $k \in \{1,2,3,4\}$ encodes the checkerboard corner point number and $p \in \{1,2,3,4,5\}$ encodes the type ($\star$) of the scan group (Tab. I).

A coordinate system is defined using four position vectors expressed in CS_{vol}. The first column vector represents the origin $\vec{\mathbf{o}}$ of the corresponding CS expressed in CS_{vol}. The remaining three column vectors represent the positions where the unit vectors (basis vectors) of the corresponding CS point to:

$$\text{CS} = \underbrace{\begin{pmatrix} o_x & x_x & y_x & z_x \\ o_y & x_y & y_y & z_y \\ o_z & x_z & y_z & z_z \\ 1 & 1 & 1 & 1 \end{pmatrix}}_{\text{Homogeneous coordinates in } \text{CS}_{\text{vol}}} .$$

Usually a CS is represented with a rigid 4×4-transformation matrix (isometry) consisting of a rotation and a translation. Our slightly different CS definition has the advantage, that the unit vectors can directly be extracted after a transformation is applied to the CS.

First, we define CS_{lin1} and CS_{lin2}, which represent the linear stage 1 and 2, the two stages at the bottom of the microstage stack. The origins $\vec{\mathbf{o}}$ of CS_{lin1} and CS_{lin2} are given by the median ($\widetilde{\ }$) of three corner points, where $k = 1$. These three corner points come from volumes, where the stages were in neutral position during the scan ($\star 1$ volumes):

$$\vec{\mathbf{o}}(\text{CS}_{\text{lin1}}) = \vec{\mathbf{o}}(\text{CS}_{\text{lin2}}) = \widetilde{G_1^1}.$$

The x-axes of CS_{lin1} and CS_{lin2} are pointing in the positive direction of the corresponding translational axis of the appropriate microstage. They are defined using the median ($\widetilde{\ }$) of all four translation vectors

$$\vec{\mathbf{x}}(\text{CS}_{\text{lin1}}) = \vec{\mathbf{o}}(\text{CS}_{\text{lin1}}) + \frac{\vec{\mathbf{x}_1}}{\|\vec{\mathbf{x}_1}\|},$$

where $\vec{\mathbf{x}_1} = \widetilde{\{\mathbf{c}_1^k - \mathbf{c}_2^k | k \in \{1,2,3,4\}\}}$ and

$$\vec{\mathbf{x}}(\text{CS}_{\text{lin2}}) = \vec{\mathbf{o}}(\text{CS}_{\text{lin2}}) + \frac{\vec{\mathbf{x}_2}}{\|\vec{\mathbf{x}_2}\|},$$

where $\vec{\mathbf{x}_2} = \widetilde{\{\mathbf{c}_4^k - \mathbf{c}_5^k | k \in \{1,2,3,4\}\}}$.

The y-axis $\vec{\mathbf{y}}$ and z-axis $\vec{\mathbf{z}}$ of both systems are defined in an arbitrary way using the cross product, such that we get well defined right handed CSs with orthogonal axes. Particular orientations of $\vec{\mathbf{y}}$ and $\vec{\mathbf{z}}$ are not important, since we use these two CSs only for translation along the x-axis.

Second, we define CS_{gon} and CS_{rot}, which represent the goniometer and the rotation stages, the two topmost stages of the microstage stack. The origins $\vec{\mathbf{o}}$ of CS_{gon} and CS_{rot} are given by best fit circle centers. Because all checkerboard corners k of the particular measurements lie in a plane perpendicular to the rotation axes of the stages, we take the median of the found circle centers. To find the appropriate circle centers we fit for all four corner points k a circle using three measurements per fit. The best fit circle-function (BFC) [10] returns the center of the fitted circle:

$$\vec{\mathbf{o}}(\text{CS}_{\text{gon}}) = \widetilde{\{\text{BFC}(\mathbf{c}_7^k, \mathbf{c}_8^k, \mathbf{c}_{10}^k) | k \in \{1,2,3,4\}\}},$$

$$\vec{\mathbf{o}}(\text{CS}_{\text{rot}}) = \widetilde{\{\text{BFC}(\mathbf{c}_{11}^k, \mathbf{c}_{12}^k, \mathbf{c}_{14}^k) | k \in \{1,2,3,4\}\}}.$$

To define the x-axes (rotation axes) of the two topmost stages of the microstage stack, we take the normal vector perpendicular to the plane given by the appropriate corner points:

$$\vec{\mathbf{x}}(\text{CS}_{\text{gon}}) = \widetilde{\{(\mathbf{c}_7^k - \mathbf{c}_8^k) \times (\mathbf{c}_7^k - \mathbf{c}_{10}^k) | k \in \{1,2,3,4\}\}},$$

$$\vec{\mathbf{x}}(\mathrm{CS_{rot}}) = \widetilde{\{(\mathbf{c}_{11}^k - \mathbf{c}_{12}^k) \times (\mathbf{c}_{11}^k - \mathbf{c}_{14}^k) | k \in \{1,2,3,4\}\}},$$

where $\times$ denotes the cross-product. The y-axis $\vec{\mathbf{y}}$ and z-axis $\vec{\mathbf{z}}$ of both systems are again defined in an arbitrary way using the cross product, such that we get well defined right handed CSs with orthogonal axes.

Third, we determine the center of the cornea best fit sphere, as well as the center of the eyeball best fit sphere based on the prepared mesh from measurement #1. To do so, we use the segmented and cleaned meshes and we fit a sphere in a least-square-sense [10]. We first rearrange the general equation of a sphere,

$$(x_i - x_0)^2 + (y_i - y_0)^2 + (z_i - z_0)^2 = r^2,$$

such that we can write the expression in matrix notation and solve for the unknowns x_0, y_0, z_0, and r, which represent the center coordinates and the radius of the sphere. The variables x_i, y_i, and z_i are the coordinates of any point lying on the surface of the particular sphere. This results in two vectors $\mathbf{z}_\mathrm{c}$ for the cornea center and $\mathbf{z}_\mathrm{e}$ for the eyeball center containing the best fit sphere center coordinates and the appropriate radius.

Figure 7 illustrates $\mathbf{z}_\mathrm{c}$, $\mathbf{z}_\mathrm{e}$, and the vertices of the mesh (gray dots) with the corresponding best fit spheres (BFS). The visualized checkerboard corners ($\mathbf{c}^k$) represent the median of the corners, where the stages are in neutral position($\star_1$ containing #1, #4, and #7).

The kinematic model is at this stage characterized such that we have defined four CSs corresponding to a microstage each and the centers and radii of the cornea and the eyeball. All these position vectors are expressed in $\mathrm{CS_{vol}}$. In order to get the true position of the sphere centers (cornea or eyeball), we just have to translate $\mathbf{z}_\mathrm{c}$ or $\mathbf{z}_\mathrm{e}$ along the x-axis of linear stage CSs or rotate around the x-axis of the goniometer or rotation stage according to what is adjusted at the testing stage hardware (i.e. the microstages).

Using the Kinematic Model. The trained testing stage model takes four parameters (P_1, P_2, P_3, P_4). These are the four individual microstage position settings which are set on the testing stage hardware while the eye tracker estimates the cornea center for the corresponding eye position. P_1 and P_2 are in millimeters (mm). P_3 and P_4 are in angular degrees (°). Processing these parameters, the trained kinematic model is able to return (expressed in the common $\mathrm{CS_{cb}}$) the position of the cornea center. This position acts as ground truth for the eye tracker validation (Figure 4). If we are adjusting a certain microstage position (e.g. $P_1 = +6\,\mathrm{mm}$ on the linear stage 1), then this affects not only the position of $\mathbf{z}_\mathrm{c}$ and $\mathbf{z}_\mathrm{e}$, but also the microstages (their CSs, respectively) above the microstage which gets adjusted. The microstage stack is as follows (from bottom to top): $\mathrm{CS_{lin1}}$, $\mathrm{CS_{lin2}}$, $\mathrm{CS_{gon}}$, and $\mathrm{CS_{rot}}$. And on top of the stack is the eye with $\mathbf{z}_\mathrm{c}$ and $\mathbf{z}_\mathrm{e}$.

The workflow is as follows:

1. Hardware adjustment of a microstage a ($a \in \{\mathrm{lin1}, \mathrm{lin2}, \mathrm{gon}, \mathrm{rot}\}$)
2. Basis change from $\mathrm{CS_{vol}}$ to the corresponding CS_a of all remaining CSs, which are above the current CS_a in the stack
3. Basis change to CS_a of the sphere centers ($\mathbf{z}_\mathrm{c}$ and $\mathbf{z}_\mathrm{e}$)
4. Application of the transformation matrix T_a (e.g. rotation of $+3\,°$) to all the remaining CSs and the sphere centers
5. Basis change of the CSs and the sphere centers back to $\mathrm{CS_{vol}}$

The workflow is repeated for all microstages (for all four parameters, respectively) beginning with the lowest one.

The individual rigid transformations T_a, which are applied on the corresponding local CS look as follows (translation along or rotation around x-axis):

$$\mathrm{T_{lin1}} = \begin{bmatrix} 1 & 0 & 0 & P_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathrm{T_{lin2}} = \begin{bmatrix} 1 & 0 & 0 & P_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathrm{T_{gon}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(P_3) & -\sin(P_3) & 0 \\ 0 & \sin(P_3) & \cos(P_3) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathrm{T_{rot}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(P_4) & -\sin(P_4) & 0 \\ 0 & \sin(P_4) & \cos(P_4) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

The rigid transformations ${}^{a}\mathrm{T_{vol}}$ to change the basis from $\mathrm{CS_{vol}}$ to CS_a and back are defined as follows. For this, we use a method based on singular value decomposition (SVD), which is robust in terms of noise [11]. The method returns a rigid transformation ${}^{a}\mathrm{T_{vol}}$ (rotation and translation) when passing CS_a-matrix (expressed in $\mathrm{CS_{vol}}$) and the $\mathrm{CS_{vol}}$-matrix (expressen in $\mathrm{CS_{vol}}$):

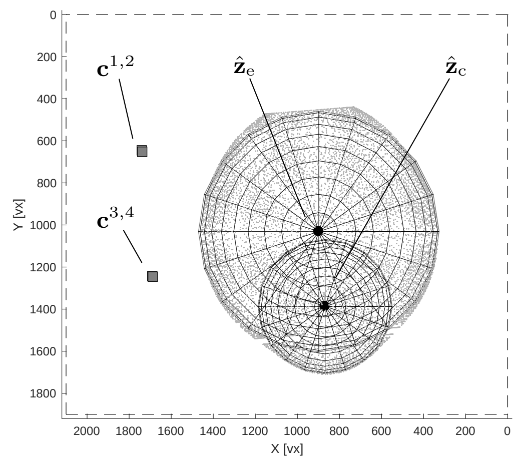
$$\mathrm{CS_{vol}} = \underbrace{\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}}_{\text{Homogeneous coordinates in } \mathrm{CS_{vol}}}.$$

Having ${}^{a}\mathrm{T_{vol}}$, we change the basis of the remaining CSs (CSs above the current one in the microstage stack), $\mathbf{z}_\mathrm{c}$, and $\mathbf{z}_\mathrm{e}$. Afterwards, we apply the transformation T_a and change the basis back to $\mathrm{CS_{vol}}$ for all CSs b, which are above CS_a:

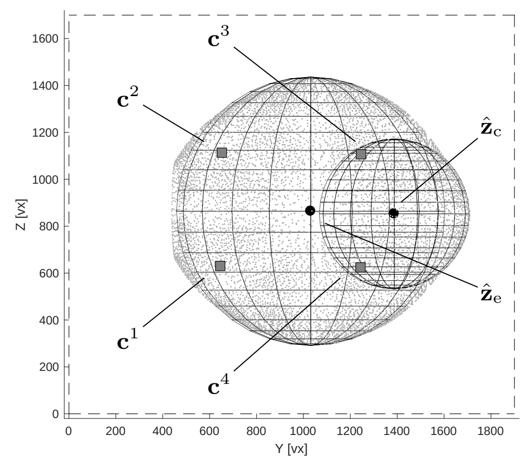
$$\mathrm{CS}_b' = (\ {}^{a}\mathrm{T_{vol}}\ )^{-1} \cdot (\ \mathrm{T}_a \cdot (\ {}^{a}\mathrm{T_{vol}} \cdot \mathrm{CS}_b\ )).$$

where a represents the CS, which we adjust (e.g. $\mathrm{CS_{lin1}}$). The sphere centers are adjusted as well for each parameter P_1, P_2, P_3, P_4:

$$\mathbf{z}_\mathrm{e}' = (\ {}^{a}\mathrm{T_{vol}}\ )^{-1} \cdot (\ \mathrm{T}_a \cdot (\ {}^{a}\mathrm{T_{vol}} \cdot \mathbf{z}_\mathrm{e}\ )),$$



(a) Top-down view



(b) Lateral view

Figure 7: Testing stage model visualization

$$\mathbf{z}'_c = (\ {}^{\alpha}\mathrm{T}_{\mathrm{vol}}\ )^{-1} \cdot (\ \mathrm{T}_a \cdot (\ {}^{\alpha}\mathrm{T}_{\mathrm{vol}} \cdot \mathbf{z}_c\ )).$$

Step-by-step, we apply all transformations for a certain testing stage configuration, until we have the position $\mathbf{z}_c$ and $\mathbf{z}_e$ for the current microstage configuration expressed in $\mathrm{CS}_{\mathrm{vol}}$. The last step is to change the basis of the sphere centers from $\mathrm{CS}_{\mathrm{vol}}$ to $\mathrm{CS}_{\mathrm{cb}}$, our common CS.

For the eye tracker tests, the tracker is rigidly mounted to a certain position, such that the checkerboard pattern (also attached to the eye holder) is completely visible by the eye tracker camera. For the external referencing of the eye tracker (here with the testing stage) we perform a homography estimation [12] based on a checkerboard pattern [5, 6]. This enables the eye tracker to express its guess about the sphere centers in $\mathrm{CS}_{\mathrm{cb}}$. We configure the testing stage (adjusting linear, rotation, and goniometer stages) such that the visibility of the checkerboard pattern from the eye tracker is well (sharp and complete pattern). This particular stage configuration enables us to access $\mathrm{CS}_{\mathrm{cb}}$ from our kinematic model. The origin lies on the corner point 4, the x-axis points towards corner point 1 and the y-axis points towards corner point 3 (Figure 6). This $\mathrm{CS}_{\mathrm{cb}}$ definition holds for both the eye tracker and the testing stage model.

The workflow described above is applied again at the very end to transform the sphere centers to $\mathrm{CS}_{\mathrm{cb}}$ according to the microstage configuration (P_1, P_2, P_3, P_4) at the time of external referencing.

III. EXPERIMENTS

A. Kinematic Model consistency

To make sure that we trained our testing stage model sufficiently accurate, we used the μCT measurements of type $\star 5$ and $\star 2$ (see Tab. I) to validate the integrity of the trained x-axes of the individual CSs. We used the median checkerboard corner points of the measurements $\star 2$ ($\{\widetilde{G_2^1}, \widetilde{G_2^2}, \widetilde{G_2^3}, \widetilde{G_2^4}\}$) to predict with our testing stage model the new checkerboard corner locations under four certain configurations. We used one configuration (P_1, P_2, P_3, P_4) for each DOF. For this, we took the four different configuration sets from the measurements $\star 5$. Having the new checkerboard corner locations calculated, we compared the model estimates (based on measurements $\star 2$) with the checkerboard corners, which we extracted manually (measurements $\star 5$). The mean error (corner-reprojection-error) of the four $\star 5$-measurements times four checkerboard corners (16 points) was 31 µm.

Additionally, we analyzed the angles between the x-axes of the trained coordinate systems (CS_{lin1}, CS_{lin2}, CS_{gon}, and CS_{rot}). Assuming the microstages are ideally mounted and aligned on top of each other, we would have to expect angles of $90\,^{\circ}$ between the x-axes. We found out that we have a $89.5\,^{\circ}$ angle between the linear stages, $90.9\,^{\circ}$ between the linear stage 2 and the goniometer rotation axis and $90.2\,^{\circ}$ between the rotation axes of the goniometer and the rotation stage.

We also performed cornea-fit-refit experiments, where we fitted a new sphere to all of the scans $\star 5$. The mean deviation between the five sphere centers was $\pm$ 36 µm.

B. Eye tracker accuracy

Setup. We tested a video based stereo eye tracker [6] with the proposed testing stage hardware and the corresponding kinematic model. For this, we rigidly mounted both the testing stage and the eye tracker on an optical bench and aligned the eye tracker such that a good visibility on to the artificial eye of the testing stage was given. We adjusted the focus and the aperture of the lens (part of the eye tracker) and performed a camera calibration [12] to get the intrinsic camera parameters (focal length, distortions). Having the camera calibrated, we adjusted the testing stage such that the holder's checkerboard was visible by the eye tracker ($P_1 = +8\,\text{mm}, P_2 = +7\,\text{mm}, P_3 = 8\,°, P_4 = +56\,°$). With the eye tracker we performed a homography estimation (based on an image snapshot of the checkerboard) in order to be able to transform the eye tracker output, the center of the corneal curvature, to the common checkerboard coordinate system CS_{cb} [5]. The camera calibration and the referencing to an external system (testing stage or a medical device) is part of the eye tracker calibration procedure.

For the actual validation, we set 20 different eye positions and orientations with the testing stage to mimic snapshots of a natural eye movement. To get a better impression of the results we only adjusted one microstage at the same time, while the three other stages were in neutral position. The microstages were set to $P1 = \{7.5, 10, 12.5, 15, 17.5\}$[mm], then $P2 = \{7.5, 10, 12.5, 15, 17.5\}$[mm], $P3 = \{-10, -5, 0, 5, 10\}$[°], and $P4 = \{290, 298, 307, 316, 324\}$[°]. This resulted in five positions per microstage and with that in 20 eye tracker estimates of the corneal curvature location $\mathbf{z}_{\text{c}}^{\star}$. We set the same parameters on our kinematic model and generated the ground truth of the center location of the corneal curvature. Figure 4 illustrates this workflow.

Results. We compared the 20 different center locations of corneal curvature from the eye tracker with the ground truth data from the testing stage. The mean deviation between two 3D points, the accuracy a respectively is as follows: The mean accuracy $\mu(a) = 0.68\,\text{mm}$, the median accuracy $\widetilde{a} = 0.67\,\text{mm}$. Subdivided into the individual orientation components: The mean accuracy $\mu(a_x) = 0.32\,\text{mm}$, the median accuracy $\widetilde{a_x} = 0.33\,\text{mm}$. The mean accuracy $\mu(a_y) = -0.09\,\text{mm}$, the median accuracy $\widetilde{a_y} = -0.09\,\text{mm}$. The mean accuracy $\mu(a_z) = -0.54\,\text{mm}$, the median accuracy $\widetilde{a_z} = -0.55\,\text{mm}$. Figure 8 and Figure 9 illustrate the distribution of the error.

Thanks to the proposed method we were able to analyze the nature of the error and unveil a slight bias of a yet unknown source. For this, we removed the average error vector from our eye tracker estimates and compared the result again with the ground truth, then we got a mean relative error $\mu(a_{\text{rel}}) = 0.32\,\text{mm}$. By eliminating this error, the overall eye tracker accuracy can even be increased.

We also evaluated the accuracy of the eye orientation. For this, we calculated the geometrical axes for the eye tracker estimate by using the pupil center and the center of corneal curvature $\mathbf{z}_c^{\star}$ and for the kinematic model by using the centers of both spheres $\mathbf{z}_c$, $\mathbf{z}_e$. In theory, all four points lie on the geometrical axis, however, it is not the case for our eye phantom. That is why we calculated the relative angle between the geometrical axes from one measurement to the next and then we compared these relative angles between the ground truth and the eye tracker estimates. The mean relative angle error is 0.50°, which indicates high angular precision.

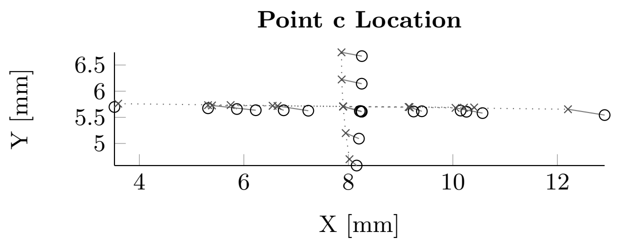


Figure 8: Accuracy in the X/Y plane (o = eye tracker estimate, x = ground truth, ... = DOF)

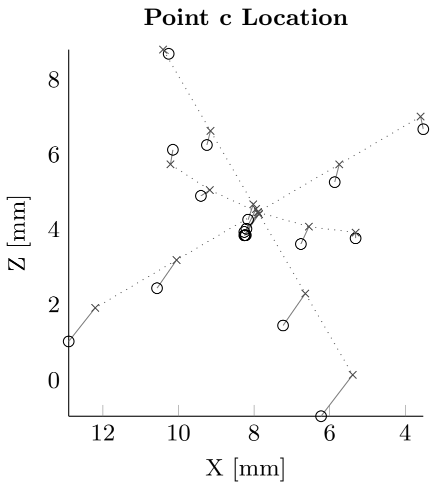


Figure 9: Accuracy in the X/Z plane (o = eye tracker estimate, x = ground truth, ... = DOF)

IV. DISCUSSION

We were able to successfully validate the eye tracker of interest with our testing stage hardware and the corresponding kinematic model. The tests showed that the eye tracker can determine the eye location (center of corneal curvature) with an accuracy below 0.7 mm. The accuracy of the validated navigation system for proton radiotherapy hence fulfills the requirements of sub-millimeter accuracy. The mean relative error $\mu(a_{\text{rel}})$ is smaller by roughly a factor of two compared to the mean error $\mu(a)$, which is a strong indication for high precision but also for a slight bias of a yet unknown source. Our system helped to detect and quantify this bias.

Figure 8 and Figure 9 show this slight systematic error along the longitudinal axis of the eye tracker.

It is difficult to compare the results to any other similar validation method, because to our best knowledge, no one did so far such a comprehensive validation of the eye location accuracy. Having for instance a closer look at [4], it is not clear how exactly the ground truth was generated.

A. Testing stage hardware and kinematic model

The systematic error from the eye tracking tests may be explained by an imprecise cornea best fit sphere. We prepared the cornea mesh in a way, where we only had limited influence on the vertex distribution. Fitting a sphere with another method than with a least-square method might be more accurate.

Maybe the most important error source is the manual segmentation of the checkerboard corner points. To improve this, we suggest exchanging the checkerboard pattern, which is used on the one hand for the external referencing of the eye tracker (homography) and on the other hand to train and validate the whole testing stage model. Hence, the pattern, its segmentation respectively, is central for the validation. A better pattern might be dots in a certain arrangement (similar to the squares in the checkerboard pattern). This pattern could easily be segmented automatically, by choosing the center of mass of the circles or the ellipsoids, respectively, taking the thickness of the ink into account.

B. Eye tracker

Depending on the application different levels of accuracy are required. Our achieved sub-millimeter accuracy in determining the eye location is sufficient for our medical application with especially high demands. If there should be higher demands, the detailed validation results, for instance the distribution of the error, might provide helpful information for eye tracker improvement.

V. CONCLUSION

Using an eye tracker to localize the eye in space can potentially improve today's eye interventions. For instance, when treating eye tumors with protons, our non-invasive eye tracker based solution might some day replace the state-of-the-art invasive navigation method.

We proposed a quantitative evaluation method with which we showed that our eye tracker is able to fulfill the requirements, namely, to determine the location of the eye with sub-millimeter accuracy. Our proposed evaluation method does not replace the eye tracker tests with volunteers that are used nowadays, but it complements the validation, enabling new eye tracking applications: eye localization.

We are sure, that in the future more and more applications, especially in ophthalmology, will benefit from an eye localization system.

ACKNOWLEDGMENTS

We would like to thank the members of the *Biomaterials Science Center (BMC)* of the University of Basel for their support with the data acquisition with the μCT system. We thank also Otto E. Martin from the *Swiss Institute For Artificial Eyes* for providing us a handcrafted eye and for sharing his profound knowledge. The work is funded by the *Swiss National Science Foundation (SNSF)*.

[1] Narcizo Fabricio Batista, Queiroz Jose Eustaquio, Gomes Herman Martins. Remote Eye Tracking Systems: Technologies and Applications in *2013 26th Conference on Graphics, Patterns and Images Tutorials*:15–22IEEE 2013.

[2] Hansen Dan Witzner, Ji Qiang. In the Eye of the Beholder: A Survey of Models for Eyes and Gaze *Pattern Analysis and Machine Intelligence, IEEE Transactions on.* 2010;32:478–500.

[3] Duchowski Andrew T. *Eye tracking methodology - theory and practice (2. ed.).* London: Springer London 2007.

[4] Via Riccardo, Fassi Aurora, Fattori Giovanni, et al. Optical eye tracking system for real-time noninvasive tumor localization in external beam radiotherapy *Medical Physics.* 2015;42:2194–2202.

[5] Wyder Stephan, Hennings Fabian, Pezold Simon, Hrbacek Jan, Cattin Philippe C. With Gaze Tracking Toward Noninvasive Eye Cancer Treatment *Biomedical Engineering, IEEE Transactions on.* 2016;63:1914–1924.

[6] Wyder Stephan, Cattin Philippe C. Stereo Eye Tracking with a Single Camera for Ocular Tumor Therapy in *Proceedings of the Ophthalmic Medical Image Analysis International Workshop*:81–88 2016.

[7] Świrski Lech, Dodgson Neil. Rendering synthetic ground truth images for eye tracker evaluation in *Proceedings of the 2014 Symposium on Eye-Tracking Research & Applications*(New York, New York, USA):219–222ACM 2014.

[8] Guestrin Elias Daniel, Eizenman Moshe. General theory of remote gaze estimation using the pupil center and corneal reflections *Biomedical Engineering, IEEE Transactions on.* 2006;53:1124–1133.

[9] Schindelin Johannes, Arganda-Carreras Ignacio, Frise Erwin, et al. Fiji: an open-source platform for biological-image analysis *Nature Methods.* 2012;9:676–682.

[10] Leon Steven J. Linear algebra with applications Macmillan New York 1980.

[11] Besl P J, McKay H D. A method for registration of 3-D shapes *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 1992;14:239–256.

[12] Zhang Zhengyou. A flexible new technique for camera calibration *Pattern Analysis and Machine Intelligence, IEEE Transactions on.* 2000;22:1330–1334.