



# OPEN GAZE API BY GAZEPOINT



Version 2.0

## Contents

1	Introduction .....	3
2	Basic Structure .....	3
2.1	Controlling the Server .....	3
2.2	Client Commands .....	3
2.3	Server Commands .....	4
2.4	Delimiters .....	4
3	Configuration Commands .....	4
3.1	ENABLE_SEND_DATA .....	4
3.2	ENABLE_SEND_COUNTER .....	5
3.3	ENABLE_SEND_TIME .....	5
3.4	ENABLE_SEND_TIME_TICK .....	5
3.5	ENABLE_SEND_POG_FIX .....	5
3.6	ENABLE_SEND_POG_LEFT .....	5
3.7	ENABLE_SEND_POG_RIGHT .....	5
3.8	ENABLE_SEND_POG_BEST .....	5
3.9	ENABLE_SEND_PUPIL_LEFT .....	5
3.10	ENABLE_SEND_PUPIL_RIGHT .....	5
3.11	ENABLE_SEND_EYE_LEFT .....	5
3.12	ENABLE_SEND_EYE_RIGHT .....	5
3.13	ENABLE_SEND_CURSOR .....	5
3.14	ENABLE_SEND_USER_DATA .....	5
3.15	CALIBRATE_START .....	5
3.16	CALIBRATE_SHOW .....	6
3.17	CALIBRATE_TIMEOUT .....	6
3.18	CALIBRATE_DELAY .....	6
3.19	CALIBRATE_RESULT_SUMMARY .....	7
3.20	CALIBRATE_CLEAR .....	7
3.21	CALIBRATE_RESET .....	7
3.22	CALIBRATE_ADDPOINT .....	8
3.23	USER_DATA .....	8
3.24	TRACKER_DISPLAY .....	8

3.25	TIME_TICK_FREQUENCY .....	9
3.26	SCREEN_SIZE .....	9
3.27	CAMERA_SIZE.....	9
3.28	PRODUCT_ID .....	10
3.29	SERIAL_ID .....	10
3.30	COMPANY_ID .....	10
3.31	API_ID.....	10
4	Calibration Records.....	11
4.1	CALIB_START_PT .....	12
4.2	CALIB_RESULT_PT .....	12
4.3	CALIB_RESULT .....	13
5	Data Records.....	14
5.1	Counter .....	14
5.2	Time .....	14
5.3	Time Tick .....	15
5.4	Fixation POG.....	15
5.5	Left Eye POG.....	16
5.6	Right Eye POG .....	16
5.7	Best POG .....	16
5.8	Left Eye Pupil.....	17
5.9	Right Eye Pupil .....	17
5.10	Left Eye 3D Data.....	18
5.11	Right Eye 3D Data.....	18
5.12	Cursor position.....	19
5.13	User data.....	19
6	API Version 2.0 Revisions.....	20

## 1 Introduction

The Open Gaze Application Programming Interface (API) was first published in 2010 as an open-source alternative to proprietary vendor formats for communicating with eye-tracking devices. The vision is to allow developers a common interface to integrate into their application without having to add customizations for each eye-tracking system on the market.

A key requirement of the API is that it not require any DLL's, libraries, or any programming language or platform specific components. The API uses a standard TCP/IP socket for communication between a client (the application) and the server (the source of eye-tracking data). The data format uses the extensible markup language (XML) to format the data transmit between the client and server. Since both TCP/IP and XML are open standards, they can be readily implemented in any language or operating system.

## 2 Basic Structure

A client-server communication session would typically begin with client initiating a connection to the server. The client would then request which data records it is interested in receiving, then commanding the server to begin streaming the data records out. The client then simply listens for the data records packaged in XML formatted strings sent by the server.

### 2.1 Controlling the Server

A connection to the server is created by opening a TCP/IP socket with the IP address of the server (typically use "127.0.0.1" as the localhost IP address if both client and server are running on the same machine). The server default port value is 4242 although the server software may allow the port number to be modified. The client and server do not need to be on the same computer.

### 2.2 Client Commands

The client has two XML tags for communicating with the server, GET and SET. These read and write to data variables on the server. The data variables that may be manipulated are listed in Sections 3 and 4.

An example of the SET command is:

```
CLIENT SEND: <SET ID="CALIBRATE_SHOW" STATE="1" />
SERVER SEND: <ACK ID="CALIBRATE_SHOW" STATE="1" />
```

The '<' and '/>' characters identify the start and end of the XML text string respectively. The SET indicates the command is setting a value to a variable. The variable being modified is identified by its ID CALIBRATE\_SHOW in this case, and the value set is '1' identified by the STATE parameter. A GET command is formatted in a similar manner but without the STATE parameter. The server response will be discussed in the following section.

```
CLIENT SEND: <GET ID="CALIBRATE_SHOW" />
SERVER SEND: <ACK ID="CALIBRATE_SHOW" STATE="1" />
```

## 2.3 Server Commands

The server can send data to the client with four different tags, ACK, NACK, CAL, and REC. The ACK and NACK responses are generated when responding to GET and SET commands. The CAL responses are generated based on the operations the calibration routine is currently performing. The REC strings hold the data record that was transmitted.

In the example above, the client requested the calibration window be shown and the server responded that the variable was set to '1' or TRUE.

```
CLIENT SEND: <SET ID="CALIBRATE_SHOW" STATE="1" />
SERVER SEND: <ACK ID="CALIBRATE_SHOW" STATE="1" />
```

ACK indicates a successful command while NACK would indicate a failure.

The server CAL and REC tags will be described in greater detail in the Sections 4 and 0.

## 2.4 Delimiters

Each XML record is delimited by carriage return ('CR' or '\n') and a line-feed characters ('LF', '\n') one after the other. The actual characters transmit would appear as shown below between the quotes "".

```
CLIENT SEND: "<SET ID="CALIBRATE_SHOW" STATE="1" />\r\n"
SERVER SEND: "<ACK ID="CALIBRATE_SHOW" STATE="1" />\r\n"
```

# 3 Configuration Commands

The configuration commands are used to set and get configuration settings from the server. The configuration commands are described in the following sections.

## 3.1 ENABLE\_SEND\_DATA

**Description:** Start or stop the streaming of data from the server to client.

**Parameter:** STATE

**Parameter type:** Boolean (0 or 1)

**Permissions:** Read and Writable

**Example:** '...' the data being sent is described in Section 5

```
CLIENT SEND: <SET ID="ENABLE_SEND_DATA" STATE="1" />
SERVER SEND: <ACK ID="ENABLE_SEND_DATA" STATE="1" />
SERVER SEND: <REC ... />
SERVER SEND: <REC ... />
SERVER SEND: <REC ... />
```

### 3.2 ENABLE\_SEND\_COUNTER

### 3.3 ENABLE\_SEND\_TIME

### 3.4 ENABLE\_SEND\_TIME\_TICK

### 3.5 ENABLE\_SEND\_POG\_FIX

### 3.6 ENABLE\_SEND\_POG\_LEFT

### 3.7 ENABLE\_SEND\_POG\_RIGHT

### 3.8 ENABLE\_SEND\_POG\_BEST

### 3.9 ENABLE\_SEND\_PUPIL\_LEFT

### 3.10 ENABLE\_SEND\_PUPIL\_RIGHT

### 3.11 ENABLE\_SEND\_EYE\_LEFT

### 3.12 ENABLE\_SEND\_EYE\_RIGHT

### 3.13 ENABLE\_SEND\_CURSOR

### 3.14 ENABLE\_SEND\_USER\_DATA

**Description:** Enable a data record variable in the data record string (described in Section 5 below).

**Parameter:** STATE

**Parameter type:** boolean (0 or 1)

**Permissions:** Read and Writable

**Example:**

```
CLIENT SEND: <SET ID="ENABLE_SEND_COUNTER" STATE="1" />
SERVER SEND: <ACK ID="ENABLE_SEND_COUNTER" STATE="1" />
```

### 3.15 CALIBRATE\_START

**Description:** Start or stop the calibration process

**Parameter:** STATE

**Parameter type:** boolean (0 or 1)

**Permissions:** Read and Writable

**Example:**

```
CLIENT SEND: <GET ID="CALIBRATE_START" />
SERVER SEND: <ACK ID="CALIBRATE_START" VALUE="0" />

CLIENT SEND: <SET ID="CALIBRATE_START" VALUE="1" />
SERVER SEND: <ACK ID="CALIBRATE_START" VALUE="1" />
```

### 3.16 CALIBRATE\_SHOW

**Description:** Show or hide the calibration graphical window

**Parameter:** STATE

**Parameter type:** boolean (0 or 1)

**Permissions:** Read and Writable

**Example:**

```
CLIENT SEND: <GET ID="CALIBRATE_SHOW" />
SERVER SEND: <ACK ID="CALIBRATE_SHOW" VALUE="0" />

CLIENT SEND: <SET ID="CALIBRATE_SHOW" VALUE="1" />
SERVER SEND: <ACK ID="CALIBRATE_SHOW" VALUE="1" />
```

### 3.17 CALIBRATE\_TIMEOUT

**Description:** Set or get the duration of the calibration point (not including animation time below)

**Parameter:** VALUE

**Parameter type:** float (> 0)

**Permissions:** Read and Writable

**Example:**

```
CLIENT SEND: <GET ID="CALIBRATE_TIMEOUT" />
SERVER SEND: <ACK ID="CALIBRATE_TIMEOUT" VALUE="1.25" />

CLIENT SEND: <SET ID="CALIBRATE_TIMEOUT" VALUE="2" />
SERVER SEND: <ACK ID="CALIBRATE_TIMEOUT" VALUE="2" />
```

### 3.18 CALIBRATE\_DELAY

**Description:** Set or get the duration of the calibration animation (before calibration at a point begins)

**Parameter:** VALUE

**Parameter type:** float (>= 0)

**Permissions:** Read and Writable

**Example:**

```
CLIENT SEND: <GET ID="CALIBRATE_DELAY" />
SERVER SEND: <ACK ID="CALIBRATE_DELAY" VALUE="0.5" />

CLIENT SEND: <SET ID="CALIBRATE_DELAY" VALUE="1.0" />
SERVER SEND: <ACK ID="CALIBRATE_DELAY" VALUE="1.0" />
```

### 3.19 CALIBRATE\_RESULT\_SUMMARY

**Description:** Get a summary of the calibration results

**Parameter:** AVE\_ERROR (average error over all calibration points)

**Parameter:** VALID\_POINTS (number of successful calibration points)

**Permissions:** Read only

**Example:**

```
CLIENT SEND: <GET ID="CALIBRATE_RESULT_SUMMARY" />
SERVER SEND: <ACK ID="CALIBRATE_RESULT_SUMMARY" AVE_ERROR="19.43"
VALID_POINTS="5" />
```

### 3.20 CALIBRATE\_CLEAR

**Description:** Clear the internal list of calibration points

**Parameter:** PTS (number of calibration points returned only)

**Permissions:** Read and Writeable

**Example:**

```
CLIENT SEND: <GET ID="CALIBRATE_CLEAR" />
SERVER SEND: <ACK ID="CALIBRATE_CLEAR" PTS="5" />

CLIENT SEND: <SET ID="CALIBRATE_CLEAR" />
SERVER SEND: <ACK ID="CALIBRATE_CLEAR" PTS="0" />
```

### 3.21 CALIBRATE\_RESET

**Description:** Reset the internal list of calibration points to the default values

**Parameter:** PTS (number of calibration points returned only)

**Permissions:** Read and Writeable

**Example:**

```
CLIENT SEND: <GET ID="CALIBRATE_RESET" />
SERVER SEND: <ACK ID="CALIBRATE_RESET" PTS="0" />

CLIENT SEND: <SET ID="CALIBRATE_RESET" />
SERVER SEND: <ACK ID="CALIBRATE_RESET" PTS="5" />
```



### 3.22 CALIBRATE\_ADDPOINT

**Description:** Add a point to the internal list of calibration points

**Parameter:** X (calibration point position as percentage of screen width)

**Parameter:** Y (calibration point position as percentage of screen height)

**Parameter type:** float

**Permissions:** Read and Writeable

**Example:**

```
CLIENT SEND: <GET ID="CALIBRATE_ADDPOINT" />
SERVER SEND: <ACK ID="CALIBRATE_ADDPOINT" PTS="5" X1="0.50000"
Y1="0.50000" X2="0.85000" Y2="0.15000" X3="0.85000" Y3="0.85000"
X4="0.15000" Y4="0.85000" X5="0.15000" Y5="0.15000" />
```

```
CLIENT SEND: <SET ID="CALIBRATE_ADDPOINT" X="0.5" Y="0.1" />
SERVER SEND: <ACK ID="CALIBRATE_ADDPOINT" PTS="6" X1="0.50000"
Y1="0.50000" X2="0.85000" Y2="0.15000" X3="0.85000" Y3="0.85000"
X4="0.15000" Y4="0.85000" X5="0.15000" Y5="0.15000" X6="0.50000"
Y6="0.10000" />
```

### 3.23 USER\_DATA

**Description:** Set the value of the user data field for embedding custom data into the data stream

**Parameter:** VALUE (user defined)

**Parameter type:** string

**Permissions:** Read and Writeable

**Example:**

```
CLIENT SEND: <GET ID="USER_DATA" />
SERVER SEND: <ACK ID="USER_DATA" VALUE="0" />

CLIENT SEND: <SET ID="USER_DATA" VALUE="TEST1" />
SERVER SEND: <ACK ID="USER_DATA" VALUE="TEST1" />
```

### 3.24 TRACKER\_DISPLAY

**Description:** Show or hide the eye-tracker display window

**Parameter:** STATE

**Parameter type:** boolean (0 or 1)

**Permissions:** Read and Writeable

**Example:**

```
CLIENT SEND: <GET ID="TRACKER_DISPLAY" />
SERVER SEND: <ACK ID="TRACKER_DISPLAY" STATE="1" />

CLIENT SEND: <SET ID="TRACKER_DISPLAY" STATE ="0" />
SERVER SEND: <ACK ID="TRACKER_DISPLAY" STATE ="0" />
```

### 3.25 TIME\_TICK\_FREQUENCY

**Description:** Get the time-tick frequency to convert the TIME\_TICK variable to seconds

**Parameter:** FREQ

**Parameter type:** longlong

**Permissions:** Read only

**Example:**

```
CLIENT SEND: <GET ID="TIME_TICK_FREQUENCY" />
SERVER SEND: <ACK ID="TIME_TICK_FREQUENCY" FREQ="4704405731611246592" />
```

### 3.26 SCREEN\_SIZE

**Description:** Get the gaze tracking screen position and size or set the screen on which the gaze tracking is to be performed. Provides the ability to work with multi-monitor systems.

**Parameter:** X (screen X position in pixels)

**Parameter:** Y (screen Y position in pixels)

**Parameter:** WIDTH (screen width in pixels)

**Parameter:** HEIGHT (screen height in pixels)

**Parameter type:** integer

**Permissions:** Read and Writable

**Example:**

```
CLIENT SEND: <GET ID="SCREEN_SIZE" />
SERVER SEND: <ACK ID="SCREEN_SIZE" X="0" Y="0" WIDTH="1920"
HEIGHT="1080" />

CLIENT SEND: <SET ID="SCREEN_SIZE" X="-1920" Y="0" WIDTH="1920"
HEIGHT="1080"/>
SERVER SEND: <ACK ID="SCREEN_SIZE" X="-1920" Y="0" WIDTH="1920"
HEIGHT="1080" />
```

### 3.27 CAMERA\_SIZE

**Description:** Get the size of the camera sensor in pixels

**Parameter:** WIDTH (camera width in pixels)

**Parameter:** HEIGHT (camera height in pixels)

**Parameter type:** integer

**Permissions:** Read only

**Example:**

```
CLIENT SEND: <GET ID="CAMERA_SIZE" />
SERVER SEND: <ACK ID="CAMERA_SIZE" WIDTH="752" HEIGHT="480" />
```

### 3.28 PRODUCT\_ID

**Description:** Get the identifier of the current eye-tracker being used

**Parameter:** VALUE (product name)

**Parameter type:** string

**Permissions:** Read only

**Example:**

```
CLIENT SEND: <GET ID="PRODUCT_ID" />
SERVER SEND: <ACK ID="PRODUCT_ID" VALUE="GP3" />
```

### 3.29 SERIAL\_ID

**Description:** Get the serial number of the eye-tracker

**Parameter:** VALUE (serial number)

**Parameter type:** string

**Permissions:** Read only

**Example:**

```
CLIENT SEND: <GET ID="SERIAL_ID" />
SERVER SEND: <ACK ID="SERIAL_ID" VALUE="123456789" />
```

### 3.30 COMPANY\_ID

**Description:** Get the identifier of the eye-tracker manufacturer

**Parameter:** VALUE (company name)

**Parameter type:** string

**Permissions:** Read only

**Example:**

```
CLIENT SEND: <GET ID="COMPANY_ID" />
SERVER SEND: <ACK ID="COMPANY_ID" VALUE="GAZEPOINT" />
```

### 3.31 API\_ID

**Description:** Get the API version number

**Parameter:** VALUE (API version)

**Parameter type:** string

**Permissions:** Read only

**Example:**

```
CLIENT SEND: <GET ID="API_ID" />
SERVER SEND: <ACK ID="API_ID" VALUE="2.0" />
```

## 4 Calibration Records

When the user calibration takes place, a sequence of calibration specific data is sent from the server to the client, identified with the <CAL /> tag. This data includes the timing of when calibration points are shown and move to the next point, as well as the final result of the calibration process. To begin calibration through the API the following commands would be used:

```
CLIENT SEND: <SET ID="CALIBRATE_SHOW" STATE="1" />
SERVER SEND: <ACK ID="CALIBRATE_SHOW" STATE="1" />
CLIENT SEND: <SET ID="CALIBRATE_START" STATE="1" />
SERVER SEND: <ACK ID="CALIBRATE_START" STATE="1" />
```

As calibration proceeds, the following data will be sent from the server:

```
SERVER SEND: <CAL ID="CALIB_START_PT" PT="1" CALX="0.5000"
CALY="0.5000" />
SERVER SEND: <CAL ID="CALIB_RESULT_PT" PT="1" CALX="0.5000"
CALY="0.5000" />
SERVER SEND: <CAL ID=" CALIB_START_PT" PT="2" CALX="0.8500"
CALY="0.1500" />
SERVER SEND: <CAL ID=" CALIB_RESULT_PT" PT="2" CALX="0.8500"
CALY="0.1500" />
SERVER SEND: <CAL ID=" CALIB_START_PT" PT="3" CALX="0.8500"
CALY="0.8500" />
SERVER SEND: <CAL ID=" CALIB_RESULT_PT" PT="3" CALX="0.8500"
CALY="0.8500" />
SERVER SEND: <CAL ID=" CALIB_START_PT" PT="4" CALX="0.1500"
CALY="0.8500" />
SERVER SEND: <CAL ID=" CALIB_RESULT_PT" PT="4" CALX="0.1500"
CALY="0.8500" />
SERVER SEND: <CAL ID=" CALIB_START_PT" PT="5" CALX="0.1500"
CALY="0.1500" />
SERVER SEND: <CAL ID=" CALIB_RESULT_PT" PT="5" CALX="0.1500"
CALY="0.1500" />
SERVER SEND: <CAL ID="CALIB_RESULT" CALX1="0.50000" CALY1="0.50000"
LX1="0.50229" LY1="0.50279 LV1="1" RX1="0.51467" RY1="0.50870 RV1="1"
CALX2="0.85000" CALY2="0.15000" LX2="0.84943" LY2="0.14930 LV2="1"
RX2="0.84600" RY2="0.14763 RV2="1" CALX3="0.85000" CALY3="0.85000"
LX3="0.84942" LY3="0.84929 LV3="1" RX3="0.84627" RY3="0.84779 RV3="1"
CALX4="0.15000" CALY4="0.85000" LX4="0.14943" LY4="0.84930 LV4="1"
RX4="0.14616" RY4="0.84772 RV4="1" CALX5="0.15000" CALY5="0.15000"
LX5="0.14944" LY5="0.14931 LV5="1" RX5="0.14689" RY5="0.14815 RV5="1"
/>
```

#### 4.1 CALIB\_START\_PT

**Description:** Sent at the start of the specified calibration point. Each calibration point has an animation time and then a calibration time. The CALIB\_START\_PT is sent at the beginning of the animation time.

**Parameter:** PT (point number 1 to N)

**Parameter type:** integer

**Parameter:** CALX, CALY (X- and Y- coordinates of the calibration point)

**Parameter type:** float

**Example:**

```
SERVER SEND: <CAL ID="CALIB_START_PT" PT="1" CALX="0.5000"  
CALY="0.5000" />
```

#### 4.2 CALIB\_RESULT\_PT

**Description:** Sent at the end of the specified calibration point. The CALIB\_RESULT\_PT is sent at the end of the calibration time.

**Parameter:** PT (point number 1 to N)

**Parameter type:** integer

**Parameter:** CALX, CALY (X- and Y- coordinates of the calibration point)

**Parameter type:** float

**Example:**

```
SERVER SEND: <CAL ID="CALIB_RESULT_PT" PT="1" CALX="0.5000"  
CALY="0.5000" />
```

### 4.3 CALIB\_RESULT

**Description:** Sent at the end of the entire calibration process and lists the estimated point-of-gaze at each of the calibration points for both the left and right eyes, as well as a valid flag indicating if the calibration for a particular eye was successful or not at a particular calibration point.

**Parameter:** CALX?, CALY? (X- and Y- coordinates of the calibration point number ?)

**Parameter type:** float

**Parameter:** LX?, LY? (X- and Y- coordinates of the left eye point-of-gaze at calibration point ?)

**Parameter type:** float

**Parameter:** LV? (valid flag for left eye at calibration point ?)

**Parameter type:** boolean

**Parameter:** RX?, RY? (X- and Y- coordinates of the right eye point-of-gaze at calibration point ?)

**Parameter type:** float

**Parameter:** RV? (valid flag for right eye at calibration point ?)

**Parameter type:** boolean

**Example:**

```
SERVER SEND: <CAL ID="CALIB_RESULT" CALX1="0.50000" CALY1="0.50000"
LX1="0.50229" LY1="0.50279 LV1="1" RX1="0.51467" RY1="0.50870 RV1="1"
CALX2="0.85000" CALY2="0.15000" LX2="0.84943" LY2="0.14930 LV2="1"
RX2="0.84600" RY2="0.14763 RV2="1" CALX3="0.85000" CALY3="0.85000"
LX3="0.84942" LY3="0.84929 LV3="1" RX3="0.84627" RY3="0.84779 RV3="1"
CALX4="0.15000" CALY4="0.85000" LX4="0.14943" LY4="0.84930 LV4="1"
RX4="0.14616" RY4="0.84772 RV4="1" CALX5="0.15000" CALY5="0.15000"
LX5="0.14944" LY5="0.14931 LV5="1" RX5="0.14689" RY5="0.14815 RV5="1"
/>
```

## 5 Data Records

Data is transmit from the server to the client in a sequence of records formatted as XML strings. A data record is identified with the <REC /> tag. An example for the data record counter would look as follows:

```
<REC CNT="1484" />
```

In this example there is one data field CNT with a value of 1484. Each data field is described below.

### 5.1 Counter

**Description:** The counter data variable is incremented by 1 for each data record sent by the server. Useful to determine if any data packets are missed by the client.

**Parameter ID:** CNT

**Parameter type:** int

**Enable:** ENABLE\_SEND\_COUNTER

**Example:**

```
<REC CNT="1484" />
<REC CNT="1485" />
<REC CNT="1486" />
<REC CNT="1487" />
```

### 5.2 Time

**Description:** The time elapsed in seconds since the last system initialization or calibration. The time stamp is recorded at the end of the transmission of the image from camera to computer. Useful for synchronization and to determine if the server computer is processing the images at the full frame rate. For a 60 Hz camera, the TIME value should increment by 1/60 seconds.

**Parameter ID:** TIME

**Parameter type:** float

**Enable:** ENABLE\_SEND\_TIME

**Example:**

```
<REC TIME="4.99716" />
<REC TIME="5.01366" />
<REC TIME="5.02991" />
<REC TIME="5.04653" />
```

### 5.3 Time Tick

**Description:** This is a signed 64-bit integer which indicates the number of CPU time ticks. This allows high precision synchronization of gaze data with other data on the same CPU.

**Parameter ID:** TIME\_TICK

**Parameter type:** LONGLONG

**Enable:** ENABLE\_SEND\_TIME\_TICK

**Example:**

```
<REC TIME_TICK="2096547271623" />
<REC TIME_TICK="2096547490186" />
<REC TIME_TICK="2096547654197" />
```

### 5.4 Fixation POG

**Description:** The Fixation POG data provides the user's point-of-gaze as determined by the internal fixation filter.

**Parameter ID:** FPOGX, FPOGY

**Parameter type:** float

**Parameter description:** The X- and Y-coordinates of the fixation POG, as a fraction of the screen size. (0,0) is top left, (0.5,0.5) is the screen center, and (1.0,1.0) is bottom right.

**Parameter ID:** FPOGS

**Parameter type:** float

**Parameter description:** The starting time of the fixation POG in seconds since the system initialization or calibration.

**Parameter ID:** FPOGD

**Parameter type:** float

**Parameter description:** The duration of the fixation POG in seconds.

**Parameter ID:** FPOGID

**Parameter type:** integer

**Parameter description:** The fixation POG ID number

**Parameter ID:** FPOGV

**Parameter type:** boolean

**Parameter description:** The valid flag with value of 1 if the fixation POG data is valid, and 0 if it is not.

**Enable:** ENABLE\_SEND\_POG\_FIX

**Example:**

```
<REC FPOGX="0.48439" FPOGY="0.50313" FPOGS="1891.86768"
FPOGD="0.49280" FPOGID="1599" FPOGV="1" /REC TIME_TICK="2096547490186"
/>
```



## 5.5 Left Eye POG

**Description:** The POG data for the user's left eye.

**Parameter ID:** LPOGX, LPOGY

**Parameter type:** float

**Parameter description:** The X- and Y-coordinates of the left eye POG, as a fraction of the screen size.

**Parameter ID:** LPOGV

**Parameter type:** boolean

**Parameter description:** The valid flag with value of 1 if the data is valid, and 0 if it is not.

**Enable:** ENABLE\_SEND\_POG\_LEFT

**Example:**

```
<REC LPOGX="0.21336" LPOGY="0.44548" LPOGV="1" />
```

## 5.6 Right Eye POG

**Description:** The POG data for the user's right eye.

**Parameter ID:** RPOGX, RPOGY

**Parameter type:** float

**Parameter description:** The X- and Y-coordinates of the right eye POG, as a fraction of the screen size.

**Parameter ID:** RPOGV

**Parameter type:** boolean

**Parameter description:** The valid flag with value of 1 if the data is valid, and 0 if it is not.

**Enable:** ENABLE\_SEND\_POG\_RIGHT

**Example:**

```
<REC LPOGX="0.43623" LPOGY="0.53243" RPOGV="1" />
```

## 5.7 Best POG

**Description:** The 'best' POG data, which is the average of the left eye and right eye POG if both are available, or if not, then the value of either the left or right eye, depending on which one is valid.

**Parameter ID:** BPOGX, BPOGY

**Parameter type:** float

**Parameter description:** The X- and Y-coordinates of the best eye POG, as a fraction of the screen size.

**Parameter ID:** BPOGV

**Parameter type:** boolean

**Parameter description:** The valid flag with value of 1 if the data is valid, and 0 if it is not.

**Enable:** ENABLE\_SEND\_POG\_BEST

**Example:**

```
<REC BPOGX="0.47175" BPOGY="0.43360" BPOGV="1" />
```

## 5.8 Left Eye Pupil

**Description:** The image data relating to the left eye.

**Parameter ID:** LPCX, LPCY

**Parameter type:** float

**Parameter description:** The X- and Y-coordinates of the left eye pupil in the camera image, as a fraction of the camera image size.

**Parameter ID:** LPD

**Parameter type:** float

**Parameter description:** The diameter of the left eye pupil in pixels.

**Parameter ID:** LPS

**Parameter type:** float

**Parameter description:** The scale factor of the left eye pupil (unitless). Value equals 1 at calibration depth, is less than 1 when user is closer to the eye tracker and greater than 1 when user is further away.

**Parameter ID:** LPV

**Parameter type:** boolean

**Parameter description:** The valid flag with value of 1 if the data is valid, and 0 if it is not.

**Enable:** ENABLE\_SEND\_PUPIL\_LEFT

**Example:**

```
<REC LPCX="0.40525" LPCY="0.32822" LPD="15.23866" LPS="1.04834"  
LPV="1" />
```

## 5.9 Right Eye Pupil

**Description:** The image data relating to the right eye.

**Parameter ID:** RPCX, RPCY

**Parameter type:** float

**Parameter description:** The X- and Y-coordinates of the right eye pupil in the camera image, as a fraction of the camera image size.

**Parameter ID:** RPD

**Parameter type:** float

**Parameter description:** The diameter of the right eye pupil in pixels.

**Parameter ID:** RPS

**Parameter type:** float

**Parameter description:** The scale factor of the right eye pupil (unitless). Value equals 1 at calibration depth, is less than 1 when user is closer to the eye tracker and greater than 1 when user is further away.

**Parameter ID:** RPV

**Parameter type:** boolean

**Parameter description:** The valid flag with value of 1 if the data is valid, and 0 if it is not.

**Enable:** ENABLE\_SEND\_PUPIL\_RIGHT

**Example:**

```
<REC RPCX="0.79375" RPCY="0.54131" RPD="12.69461" RPS="1.12750"  
RPV="1" />
```

## 5.10 Left Eye 3D Data

**Description:** The computed 3D data for the left eye position.

**Parameter ID:** LEYEX, LEYEY, LEYEZ

**Parameter type:** float

**Parameter description:** The X-, Y- and Z-coordinates of the left eye with 3D space with respect to the camera focal point, in units of meters.

**Parameter ID:** LPUPILD

**Parameter type:** float

**Parameter description:** The diameter of the left eye pupil in units of meters.

**Parameter ID:** LPUPILV

**Parameter type:** boolean

**Parameter description:** The valid flag with value of 1 if the data is valid, and 0 if it is not.

**Enable:** ENABLE\_SEND\_EYE\_LEFT

**Example:**

```
<REC LEYEX="-0.04796" LEYEY="0.00305" LEYEZ="0.69235"  
LPUPILD="0.00210" LPUPILV="1" />
```

## 5.11 Right Eye 3D Data

**Description:** The computed 3D data for the right eye position.

**Parameter ID:** REYEX, REYEY, REYEZ

**Parameter type:** float

**Parameter description:** The X-, Y- and Z-coordinates of the right eye with 3D space with respect to the camera focal point, in units of meters.

**Parameter ID:** RPUPILD

**Parameter type:** float

**Parameter description:** The diameter of the right eye pupil in units of meters.

**Parameter ID:** RPUPILV

**Parameter type:** boolean

**Parameter description:** The valid flag with value of 1 if the data is valid, and 0 if it is not.

**Enable:** ENABLE\_SEND\_EYE\_RIGHT

**Example:**

```
<REC REYEX="0.04321" REYEY="0.00213" REYEZ="0.66543" RPUPILD="0.00240"  
RPUPILV="1" />
```

## 5.12 Cursor position

**Description:** The position of the mouse cursor.

**Parameter ID:** CX, CY

**Parameter type:** float

**Parameter description:** The X- and Y-coordinates of the mouse cursor, as percentage of the screen size.

**Parameter ID:** CS (not currently implemented)

**Parameter type:** integer

**Parameter description:** Mouse cursor state, 0 for steady state, 1 for left button down, 2 for right button down.

**Enable:** ENABLE\_SEND\_CURSOR

**Example:**

```
<REC CX="0.12500" CY="0.32500" CS="0" />
```

## 5.13 User data

**Description:** A custom data field that may be set by the user to contain any desired information such as synchronization markers.

**Parameter ID:** USER

**Parameter type:** string

**Parameter description:** A user defined value.

**Enable:** ENABLE\_SEND\_USER\_DATA

**Example:**

```
<REC USER="TRIG1" />
```

## 6 API Version 2.0 Revisions

- Replaced GPI with simpler USER data field
- Added CALIBRATE\_CLEAR, CALIBRATE\_ADDPOINT, CALIBRATE\_RESET
- Removed CALIBRATE\_FAST
- Removed TRACK\_WINDOW
- Removed MFG\_ID from API\_ID result