
Language and Cognition Laboratory
Department of Psychology, Columbia University

Methods report 2003.04
Revision 2, 18 November, 2003
Revision 1, 12 September, 2003
Original version, 9 April, 2003

Integrating an eye tracker with E-prime via serial port communication

Jim Magnuson
magnuson@psych.columbia.edu

Abstract

This technical report documents our experiences integrating an ISCAN ETL-400 remote eye tracker with E-Prime. The methods described here should generalize easily to other eye trackers that allow serial communication (e.g., ASL products).

Revision details, 11/18//2003

1. Increased the recommended serial baud rates
2. Added *brief* descriptions of included scripts

Revision details, 9/12/2003

3. Fixed typos in directions for configuring the ET for serial data output
4. Added details on what data to output from the ISCAN to E-Prime

Acknowledgements

I thank Andrew Hollingworth for the sample E-Prime script for reading continuous data from that got us started, and technical support staff at PST (Brandon Cernicky and Sara Burgess) for assistance with E-Basic. Thanks to David January for catching the typos that prompted the revisions.

Please direct questions and comments to Jim Magnuson.

What you need (or rather, what we have)

In our lab, we have a PC running E-Prime and a PC controlling an ISCAN ETL-400. I expect that the instructions here will generalize to other eye trackers that allow serial communication (e.g., ASL eye trackers). I will refer to E-Prime as E’, to the PC running E’ as the **ECC** (experimental control computer) and the machine controlling the ISCAN as the **ETC** (eye tracking computer).

The ETC and ECC are connected by two serial cables. One is for sending signals from the ECC to the ETC (COM1 to COM1), and the other is for sending signals from the ETC to the ECC (COM2 to COM2).[1] The cables must be NULL MODEM cables (typically what you get in a ‘standard’ serial cable).

Two methods for synchronizing eye tracking and experimental control computers

The problem with almost all EC/ET products is that they require two separate computers to run both applications. This means we must determine how to synchronize the data collected by the EC software with the ET data stream. We use two methods that make use of serial communication.

Triggering ET recording, and/or marking the ET data stream

If the EC software can reliably send a serial signal at the onset of key events (at a minimum, the onset of a trial, if you can be certain of the timing [and recording] of the rest of the events in the trial), a simple solution is to mark the ET data stream: the ET software is doing its thing, recording eye position at some rate, while continuously monitoring the serial port. When it receives meaningful data, it inserts it into its time-stamped data record. After the experiment, you must integrate the EC data file with the ET data file by matching up the times of sending (EC side) and receiving (ET side). I will describe one scheme using this technique shortly.

Receiving the ET signal on the EC computer

Alternatively, we can send data from the ET computer to the EC computer. If our EC software can monitor the serial port in a semi-continuous fashion, this method has some real advantages. You need not synch up the EC and ET data after the experiment; instead, you have a single data file with the EC and ET data right on the EC computer.[2] The tradeoff is that your post-processing must include formatting the file to be used with ISCAN’s fixation analysis software, or you must develop your own software for fixation analysis. I will describe our lab’s software for this purpose in a later methods report.

In the next two sections, I will describe implementations of these two methods.[3]

---

[1] N.B.: it doesn’t matter whether you use COM1 for ECC→ETC or ETC→ECC, or even if you connect COM1 to COM2 for one of those purposes, as long as you configure E’ and the ISCAN software accordingly.

[2] The prudent experimenter might use both the triggering method and the receiving method in order to have two records of the experiment, just in case. See the section on *Combining the Two Methods*.

[3] **If you plan to use the ET→EC method, I highly recommend the package described in methods report 2003.5**, which provides package access to common ET functions, like calibration, evaluating goodness-of-calibration, and checking the serial connection.

Method 1 (EC→ET): Sending signals from EC to ET

If the EC software can reliably send a serial signal at the onset of key events (at a minimum, the onset of a trial, if you can be certain of the timing [and recording] of the rest of the events in the trial), a simple solution is to mark the ET data stream: the ET software is doing its thing, recording eye position at some rate, while continuously monitoring the serial port. When it receives meaningful data, it inserts it into its time-stamped data record. After the experiment, you must integrate the EC data file with the ET data file by matching up the times of sending (EC side) and receiving (ET side). I will describe one scheme using this technique in this section.

In the case of the ISCAN, what it can handle is integers in the range 0,127. But it also has another nice function: the ability to use external trigger signals to turn recording on and off. When it receives integer 132, it turns recording on. When it receives integer 136, it turns recording off. So what you might do is set up an E' script in which eye tracking is turned on, say 100 ms before you expect to get informative data. At that time, you might also send a series of integers to, e.g., insert the trial number into the ET record. Assuming E' is able to (a) send the signals with minimal lag, (b) do things like play sound files with minimal lag (see PST's notes on sound cards, and their recommendation of SoundBlasterLive but not, e.g., SoundBlaster16), and (c) there are no subject-triggered events after your eye tracker start trigger (e.g., an event waits to play a sound until the subject clicks something), your data files will be easy to synchronize.[4]

Doing this in E' is pretty simple. In the simplest case, suppose you have a task where the subject sees some pictures or words, and the task is to click on one of them. You could go ahead and set up, e.g., a Slide event to display your stimuli and monitor for the mouse click that will end the trial. To turn the eye tracker on before the Slide event, insert an Inline event. The code for the Inline event can be as simple as:

```
     Serial.WriteInteger 132  'Tell ISCAN to start recording
```

This sends the eye tracker the 'start' trigger, and it will being to append data to the datafile you have specified on the ETC. To stop recording at the end of the trial, insert another Inline event immediately after the Slide event, and insert the following code:

```
     Serial.WriteInteger 136  'Tell ISCAN to stop recording
```

I'll describe a slightly more complex method below, but this is basically it. Let's walk through the steps you have to go through first, though, in order to get the ET and EC computers talking.

---

[4] If there are subject-triggered events following your start trigger, as long as you are able to record the timing of those events in the E' data file, everything will still be fine. Alternatively, you could send another signal to the ET.

EC→ ET: Connect the EC and ET computers

You need:

1. To be able to identify COM ports (typically DB9 Male ports on the back of your computer). Find COM1 on both computers.
2. A serial (null-modem) cable. It should be DB9 Female on both ends, assuming the COM ports are both DB9M. Connect COM1 to COM1.

EC→ ET: Configure E' for serial communication

Once you have the computers connected, configure E'. In a new experiment or an existing one:

1. Open the Experiment control panel (press CTRL-e or select Experiment from the Edit menu)
2. Select the Devices tab
3. Click the Add button, and select Serial.
4. Edit the Serial device, and set the options as follows (note that 115200 is the highest baud rate the ISCAN currently supports; set it to the maximum allowed by both the ET and EC software):
    a. COM Port: 1
    b. Bits per sec: 115200
    c. Data bits: 8
    d. Parity: None
    e. Stop bits: 1

EC→ ET: Configure the ISCAN software for serial input

1. Under the config menu, select AUX SERIAL DATA INPUT, and set the options as follows:
    a. Input status: ON
    b. Input port: COM1
    c. Baud rate: 115200
2. Under the config menu, select SERIAL OUTPUT, and **either** turn OUTPUT STATUS off or set the COM port to anything but COM1 (I'd recommend turning it off for now)
3. Under the config menu, select GENERAL
4. If Rec Trig is not set to SERIAL, you need to change it. To toggle through the options, press SHIFT-y (assuming what you see as the toggle is a greyed-out X).
5. Press 'o' to toggle through options shown in the lower-left corner of the screen. Under 'RECORDING SELECT CONTROL', set Samp to 240 (or whatever you are using), Parm Bank to 1, and then the parameters to: POR.H1, POR.V1, BlnkSt1, and SerIn0. The last item is not set by default, and is crucial if you want to send other info to the ISCAN datafile.
6. Under the POR CALIBRATOR #1 menu, adjust POR Out Avgs according to your application (e.g., set it to 1 if you intend to do saccade-contingent updating or if you want access to the raw data; note that by using a value greater than 1, you will be working with smoothed data).
7. You should **save these settings in a .def file**. It seems that the ISCAN Daq program automatically loads the last settings when you start it, whether you

explicitly saved them or not. By saving the settings, you'll be able to reload them even if you or someone else makes other changes.

EC→ ET: E' example: triggering the ISCAN

Once your EC and ET computers are connected and your options are set as described above, you should be ready to trigger recording from E'. A sample script, triggerexample.es, is meant to accompany this document (send email to Jim Magnuson <magnuson@psych.columbia.edu> if you do not have it).

In this example, a cross is displayed at the center of the screen. When you click the mouse, the eye tracker is turned on, the trial number is sent to the eye tracker, and a word is displayed on the screen for 2 seconds. Then the eye tracker is turned off. This is repeated 4 times. The InLine events, StartEt and StopEt, simply send the serial triggers to the ISCAN. To make sure it's working, toggle through the option window in the lower-left cornder of the ISCAN monitor (by pressing 'o') until you see the 'EYE DATA RECORDING CONTROL' window. When you click the mouse, you should see the 'Record' radio button turn on, and then turn off at the end of the trial. The 'runs recorded' and 'pts recorded' fields should increment accordingly.

The only heart of this example is in the StartEt InLine. This first sets up a scheme for coding trial numbers from 0 to 9999. The ISCAN can only receive integers from 0 to 127. There are various things you could do to maximize the information you could send, e.g., sending a series of integers where the first is how many times 128 goes into the trialnumber (up to 127 times, or 16,256, which ought to be plenty of trials) and the next integer is the modulus (between 0 and 127). The scheme employed here is that you break the trialnumber into two integers. The first contains the thousands and hundreds places, while the second has the tens and ones. So trial 9989 is coded as 99, 89, trial 989 is 9,89, trial 89 is 0,89 and trial 9 is 0,9.

StartEt does this trial coding, then it sends the start recording trigger to the eye tracker, and then it sends the integers representing the trial – with a pause of 25 ms between Serial.WriteInteger commands. From trial and error I've found that without a pause of at least 25 ms, you lose some integers.

After the experiment, you can save the data file on the ET computer (alt-f A to save as ASCII). If you look at the resulting file, you'll see the trial number info in the SerIn0 field, with about 5 records in between the two parts.

Method 2 (ET→EC): Sending signals from ET to EC

A more complex solution, but the one I prefer because it integrates the data during the experiment, is to send the point-of-regard (POR) data from the ET to the EC computer.

ET → EC: Connect the EC and ET computers

You need:

1. To be able to identify COM ports (typically DB9 Male ports on the back of your computer). Find COM2 on both computers.
2. A serial (null-modem) cable. It should be DB9 Female on both ends, assuming the COM ports are both DB9M. Connect COM2 to COM2.

ET → EC: Configure E' for serial communication

Once the computers are connected, configure E'. In a new experiment or an existing one:

1. Open the Experiment control panel (press CTRL-e or select Experiment from the Edit menu)
2. Select the Devices tab
3. Click the Add button, and select Serial.
4. Edit the Serial device, and set the options as follows:
    a. COM Port: 2
    b. Bits per sec: 115200
    c. Data bits: 8
    d. Parity: None
    e. Stop bits: 1

ET → EC: Configure the ISCAN software for serial output

1. Under the config menu, select SERIAL DATA OUTPUT, and set options as follows:
    a. Output status: ON
    b. Output port: COM2
    c. Baud rate: 115200
    d. Data form: BINARY
2. Under the POR CALIBRATOR #1 menu, adjust POR Out Avgs according to your application (e.g., set it to 1 if you intend to do saccade-contingent updating or if you want access to the raw data; note that by using a value greater than 1, you will be working with smoothed data).
3. Press 'o' to toggle through the options windows in the lower left of the ISCAN screen until you see the window for SERIAL OUT SELECT CONTROL. Set the first three parameters to be POR.H1, POR.V1, and **either** Pupil.D1 **or** BlinkSt1. Eventually, the package will make use of the third item, but for now, it does not. However, it does look for three parameters. If you specify fewer than 3, you are likely to miss samples. You may specify 4 or 5 parameters (all but the first 2 will be ignored, for now), but if you specify 6, you will lose samples.
4. You should **save these settings in a .def file**. It seems that the ISCAN Daq program automatically loads the last settings when you start it, whether you explicitly saved them or not. By saving the settings, you'll be able to reload them even if you or someone else makes other changes.

E' example: recording eye data in E-Prime

Once your EC and ET computers are connected and your options are set as described above, you should be ready to receive eye data in E'. A sample script, receiveexample.es, is meant to accompany this document (send email to Jim Magnuson <magnuson@psych.columbia.edu> if you do not have it).

In this example (based on an example provided by Andrew Hollingworth), you are prompted for a name for the eye data file. Then, on each trial, a cross is displayed at the center of the screen. When you click the mouse, eye tracker data is received and recorded to the datafile you specified, as a word is displayed on the screen for 2 seconds. This is repeated 4 times.

Note that the InLine event, RecordEye, is pretty short. It records the current time, and then, as long as the elapsed time is less than trialTime (ms recording should continue), it calls the subroutine getEtData. This subroutine is defined in the user script (alt-5 shows the script window; click on the user tab to see this part of it).

By putting this code in a subroutine, we can call it from multiple InLine scripts if we need to, rather than copying and pasting the code in (and changing it in every location whenever we need to make changes). The code is fairly well documented, but feel free to contact me if you need help with this.

Making the eye data analyzable

The ISCAN reports its data in 512 x 512 pixels. You need to define values for xOffset, yOffset, xMax and yMax in order to convert that data to EC pixel space. Here are the directions, based on the ones Andrew Hollingworth sent me:

The POR data received from the ISCAN is in its pixel coordinates, which are 512 x 512. We need to convert this to whatever resolution we are using on the ECC. We need to determine what ISCAN pixel coordinates correspond to the edges of our screen; these will, unfortunately, not be 0 and 512, as the ISCAN does not make use of the entire space.

1. Find the horizontal offset. Move the ISCAN eye cursor to the left edge of the EC display that is mirrored on the Scene Monitor (you'll want to switch to 'fine' adjustment when you get close to the edge by pressing F6). On our system, this value is 38; on Andrew's it is 37. This value will be referred to as **xOffset** throughout our methods reports (and scripts).
2. Now find the maximum horizontal value by moving the cursor to the right edge. On our system, this is 494. On Andrew's it is 467. I don't know what accounts for the discrepancy; just make sure you get a consistent value on your system (e.g., by rebooting both machines to make sure it wasn't a fluke). **NB**: you may need to adjust horizontal or vertical hold to see the entire EC image on the Scene monitor. This value is **xMax**.
3. Now do the same thing for vertical. Move the cursor to the top of the display. Ours is at 36; Andrew's is at 67. This value is **yOffset**.
4. Now find the bottom. Ours is 446; Andrew's is 479. This value is **yMax**.
5. Enter your values in the appropriate places in the SetUpVars InLine object.

The getEtData subroutine

If you are curious about how to write scripts to acquire data from the eye tracker, have a look at the *getEtData* subroutine. It uses the values of *xOffset*, *yOffset*, *xMax* and *yMax* to scale the ISCAN POR data into EC coordinates (by subtracting the offset and then multiplying by the screen resolution divided by the result). The values defined in the *receiveexample.es* file are the ones we're using in my lab. Follow the instructions to find the appropriate values for your setup, and then you can use the final example distributed with this report, *EyeMovesMouse.es*, to test your conversion. In this example, 9 calibration points (at locations used in our *IscanEyeTracker* E-Prime package, described in methods report 2003.5) are put up on the screen. Then, the mouse cursor is put to whatever location *eyeX* and *eyeY* (global variables holding the calculated EC eye position) point to. You can use this to control the mouse with someone's eyes, but more importantly, for our purposes, you can move the ISCAN eye cursor with the ET arrow keys. Since you can see the mouse on the Scene monitor, you'll be able to tell (roughly) how accurate your conversion is by whether or not the EC and ET cursors overlap as you move them around.

N.B.: this subroutine is here for example only. I strongly recommend using the *EtGetData routine included in the IscanEyeTracker package (Methods report 2003.05)* unless you need more precise control over the data acquisition process. **Also, please note that the *EtGetData* routine in that package has been completely rewritten. It is more efficient and a couple bugs were fixed. I believe the example distributed here is bug-free, but it has not been tested as thoroughly as the new version of *EtGetData.***

The scripts that accompany this report

Four E' scripts should be included in the .zip file containing this report:

1. triggerexample.es
   - Shows how to send signals from the EC to the ET to start and stop recording, mark the ISCAN data file, etc.
2. receiveexample.es
   - Shows how to continuously poll for new POR data from the ISCAN in an E' script
3. EyeMovesMouse.es
   - Takes receiving a step further; the mouse is set to the current POR read from the ISCAN. This is a great debugging tool, since you can use the arrow keys on the ISCAN to change eye position when you are not tracking anyone and you are not in move mode.
4. EyeMovesMouse_withPackage.es
   - Slightly revised version of the previous script that makes use of the package distributed along with Methods Report 2003.05. You won't be able to use it without installing the package that accompanies that report.