DDM elaborato finale

Progetto eye tracker

sviluppo prototipo con generazione di metadati
per l'analisi della lettura

Emanuele Vivoli
vivoli.emanuele@gmail.com

Francesca Del Lungo
dellungo.francesca2@gmail.com

Gennaio, Febbraio, Marzo
2019

1 Sommario

La conoscenza dell'interazione tra processi visivi e processi cognitivi durante la lettura è essenziale per capire come questa si sviluppi e sia influenzata da fattori distinti.

Lo sviluppo di tecnologie di eye tracking costituisce un campo estremamente attuale ed innovativo. I movimenti oculari sono ora ampiamente utilizzati per studiare i processi cognitivi durante la lettura, la percezione della scena e la ricerca visiva.

Durante questo progetto è stata sviluppata un'applicazione che mette a disposizione in maniera aggregata i dati provenienti dall'eye tracker, a seguito della visione di immagini con del testo scritto. Il testo si presenta in lingua italiana, con un font ben leggibile ed una grandezza dei caratteri che permettono una buona lettura; ma è possibile modificare qualsiasi parametro, compreso il testo stesso.

Il progetto sviluppato si pone come strumento di partenza per l'analisi della scrittura mediante dati provenienti da tecnologia eye tracker.

2 Introduzione

La lettura è un processo psicologico piuttosto complesso in cui i sistemi visivo e linguistico interagiscono per permettere alla persona di riconoscere ciò che vede come testo (fase di decodifica) e di comprenderne il significato (fase di comprensione).[6]

La maggior parte degli essere umani impara in pochi anni a leggere e a scrivere e così queste attività vengono realizzate con una naturalezza che non rispecchia la complessità dei processi coinvolti nel loro esercizio.

Negli ultimi due decenni sono stati sviluppati sistemi di registrazione del movimento oculare che rendono possibile la raccolta di informazioni di alta qualità sia di bambini in età scolare che di soggetti adulti.

Gli studi effettuati utilizzano strumenti per il tracciamento oculare più o meno accurati come ad esempio occhiali, computer con eye-tracker integrato o da aggiungere come unità periferica.

3 Lettura

Come già detto l'attività di lettura di un testo si compone di due fasi: una prima di decodifica, in cui sono importanti velocità e correttezza, e la seconda di comprensione.

Ogni volta che esploriamo visivamente un ambiente, guardiamo un'immagine o leggiamo un testo, facciamo movimenti oculari rapidi e continui chiamati movimenti saccadici (*saccades*). Questi durano tipicamente da 50 ms a 70 ms. Tra un movimento saccadico e un altro i nostri occhi restano fermi per circa 200-300 ms, questo è chiamato fissazione (*fixation*).

Le fissazioni sono necessarie all'occhio per inquadrare le parole ponendole ogni volta in corrispondenza del centro della retina, dove si trova l'area più sensibile dell'occhio: la fovea.

Un bambino o un adulto con uno sviluppo tipico della lettura tende a fissare lo sguardo sulle parole contenuto (nomi, verbi) e a non fissare parole funzione, considerate secondarie (di, per..) le cui caratteristiche ortografiche vengono colte con una visione periferica (visione parafoveale, comunemente detta "coda dell'occhio").

I movimenti saccadici sono più brevi o più ampi in relazione alla lunghezza della stringa successiva[5], ma in generale i nostri occhi si spostano di circa 7-9 caratteri con ogni movimento saccadico.

Durante l'analisi della posizione dello sguardo non vengono usate le informazioni relative ai movimenti saccadici, ma quelli relativi alle fissazioni, come riportato in [7].

Secondo molti studi fatti (ad esempio [3], [4]) il testo allineato a sinistra permette una lettura più semplice rispetto al testo centrato o allineato a destra. Infatti quando si centra il testo la posizione di partenza di ogni riga cambia sempre e questo costringe l'utente a fare un maggiore sforzo per trovare dove inizia ogni riga successiva quella che è stata appena letta.

Anche il testo giustificato risulta essere una pessima scelta per effettuare una lettura veloce in quanto gli spazi tra le parole sono irregolari e introducono un ulteriore grado di difficoltà e disturbo per la lettura fluente del lettore.

4 Strumentazione: Eye tracker

Il tracciamento degli occhi (*eye tracking*) è una tecnologia che consente ad un dispositivo di sapere esattamente dove sono focalizzati gli occhi in un determinato istante. L'eye tracker è capace di determinare la presenza dell'utente, la sua attenzione, concentrazione o altri stati mentali. Tali dati possono essere utilizzati per ottenere informazioni approfondite sul comportamento dei consumatori o per progettare nuove interfacce utente su vari dispositivi. [1]

Usando gli occhi come "puntatore" su uno schermo, l'eye tracking facilita le interazioni con il computer e altri dispositivi quando l'utente non può o non desidera utilizzare le mani come modulo di input.

Oltre a questi utilizzi un nuovo ed importante ambito di ricerca è quello che si occupa dell' analisi, attraverso l'uso di eye tracker, della lettura e della psicologia che sta dietro ad essa. Questo comprende le modalità, la velocità e le zone di lettura di un testo per scopi commerciali o di ricerca.

L'eyetracker risulta essere uno strumento indispensabile per l'analisi del comportamento dell' occhio durante l'attività di lettura in quanto è poco invasivo e l'utente non dovrebbe essere influenzato da esso.
Per i test effettuati è stato utilizzato Tobii Eye Tracker 4C [2] che permette di ottenere dati relativi alla posizione degli occhi sullo schermo e quelli relativi a posizione e rotazione della testa.

La tecnologia tobii riesce a stimare la posizione degli occhi sullo schermo grazie alle telecamere infrarossi, ed è capace di fornire anche molte più informazioni di quelle utilizzate; tra le quali posizione della testa e rotazione rispetto all'asse centrale di riferimento, posizione dei singoli occhi rispetto allo schermo, posizione degli occhi sullo schermo e dati inerenti le fixations.

Nel progetto utiliziamo soltanto le informazioni riguardo alle fixation.

Figura 1: Tobii eye tracker 4C

5 Progetto

L'obiettivo del progetto è quello di fornire un supporto per interfacciarsi con l'eye tracker Tobii, analizzare i dati prodotti in modo da favorire l'analisi statistica delle componenti visualizzate da un utente.

5.1 intro SDK

Una componente fondamentale del progetto è scritta in c# poichè, nel momento in cui stamo scrivendo, le librerie e l'SDK di questa tecnologia proprietaria non

sono disponibili in altri linguaggi.

In commercio fruibili agli utenti ci sono due tipi si SDK: Core SDK e Pro SDK; queste differiscono sia nei dati che forniscono con i metodi e le API, sia nei linguaggi di programmazione con i quali vengono resi disponibili, infatti la Core SDK è scritta per .NET, mentre la Pro supporta più linguaggi come Python, .NET, C++, Matlab. La Core SDK fornisce la posizione della testa rispetto allo schermo, cosa che la Pro SDK non fa, ma tutti i dati che vengono forniti con la Core SDK sono stati precedentemente filtrati dal software di Tobii mentre la Pro fornisce i dati raw (grezzi) che preleva. Inoltre per lavorare con la SDK Core non c'è bisogno di alcuna licenza, mentre la SDK Pro deve essere attivata con una licenza che va posizionata in una cartella specifica, altrimenti le funzionalità non possono essere utilizzate.

Questo è il motivo della nostra scelta di utilizzare la SDK Cor, per altre informazioni è possibile far riferimento alle email degli autori.

5.2 Struttura

Il progetto è composto di tre attori principali:

- eye tracker Tobii 4C;
- Client c# ;
- Server python.

Vi sono poi un insieme di componenti e programmi che ricoprono vari ruoli: un programma per la generazione delle immagini ed i relativi txt contenenti il testo nelle immagini, un programma che genera le immagini di heatmap e fixation, un programma per la ricerca delle componenti connesse in una immagine ed un programma per associare i caratteri del txt alle componenti connesse precedentemente trovate.

Questi oggetti interagiscono fra loro per far fluire i dati dall'eye tracker al server che li elabora associandoli ad ogni immagine visionata, e li accorpa per comprendere quali lettere delle parole vengono visualizzate ed in quale ordine.

All'avvio del server questo si mette in ascolto sulla porta 12345 in localhost, attendendo una richiesta di connessione tcp da parte di un client. A questo punto deve avvenire l'esecuzione del Client .exe (eseguibile con build effettuata in precedenza). Il Client si occuperà di richiedere una connessione tcp al server, poi crea una connessione stream con l'eye tracker in modo da gestire gli eventi che provengono dallo stream stesso (descritti in Sezione 5.3), e ad ogni evento inoltra sulla connessione tcp al Server i dati delle fixations, delegando la loro elaborazione.

Il Server appena accettata la richiesta di connessione, provvede al caricamento delle immagini da mostrare sottoforma di slider. Le immagini devono essere precedentemente create attraverso il programma "Images_generation" scritto

FS	x	y	-
DF	x	y	-
FE	x	y	-
TIME	-	-	time

Tabella 1: Struttura file csv

anch'esso in python. Nel momento in cui viene mostrata la prima immagine, il server provvede al salvataggio di ogni dato proveniente dalla tcp in un file specifico per la relativa immagine aperta. L'utente può scorrere le immagini dello slider in avanti ed indietro, ed il server avrà la responsabilità di creare nuovi file sui quali salvare i dati relativi alle altre immagini. Al termine dell'esecuzione dello slider vi sono in una cartella csvs tutti i file con il nome che fa riferimento all'immagine associata.

Vengono adesso create le immagini di heatmap e gazeplot, e salvate in una cartella specifica per ogni singola immagine dello slider. Vengono poi caricate le immagini dello slider e trovate le componenti connesse, scartate quelle non rilevanti, ed associate le componenti trovate ad ogni carattere delle parole del testo. Tale testo viene caricato dalla cartella txts dove sono presenti tutti i file txt di ogni immagine. A questo punto vengono creati i documenti csv finali.

La componente di elaborazione sarà mostrata con maggior dettaglio nella Sezione 5.4.

5.3 Raccolta dati

Grazie all' utilizzo dell' eyetracker Tobii è possibile raccogliere molte informazioni circa la posizione dello sguardo dell' utente sullo schermo. In particolare all' avvio del programma Client viene creato un nuovo host che ricerca il dispositivo connesso e crea uno stream di dati relativi alle fissazioni: fixationDataStream.

I dati relativi alle fissazioni vengono gestiti come eventi:

Begin((x, y, _) contiene i dati (coordinate x,y) relativi all' inizio di una fissazione, che vengono inoltrati dal client al server con il prefisso **FS** (Fixation Start);

Data((x, y, _) contiene i dati relativi alle coordinate degli occhi durante la fissazione stessa (sempre in un intorno del punto iniziale), che vengono in questo caso inoltrati al server con il preffiso **DF** (Data Fixation);

End((x, y, _) composto dalle coordinate del punto finale, codificato con **FE** (Fixation End); per ogni fissazione, viene misurato il tempo (in ms) della durata di questa, che viene inviato al server subito dopo l'evento FE con il codice **TIME**.

All' interno del progetto tutti questi dati vengono inviati attraverso una connessione TCP al Server che si occuperà del salvataggio dei dati ricevuti in

file csv. Infatti il Server Python deve essere attivo prima dell'esecuzione del Client C#, per consentire ai due di creare la comunicazione.

Viene, quindi, creato un file csv come in Tabella 1 per ogni immagine che è stata visualizzata. Questi vengono salvati nella stessa cartella chiamata csvs.

5.4 Elaborazione dati

I dati salvati nei file csv, effettivamente utilizzati per l'elaborazione successiva, sono quelli relativi ad inizio, fine e durata delle fissazioni (sono tralasciate le DF). Per ogni immagine dello slider vengono create due tipi di visualizzazione grafiche:

1. **Gazeplot** che mostra l'ordine, la posizione e la durata delle fissazioni. Queste vengono rappresentate graficamente attraverso cerchi, il cui raggio aumenta all' aumentare della durata della fissazione; questi sono inoltre numerati per permettere di ricostruire l'ordine cronologico e le fissazioni successive sono collegate da segmenti. Anche il colore delle circonferenze segue una scala relativa alla durata della fissazione;

2. **Heatmap** con la quale è possibile effettuare un' analisi qualitativa. Questa mostra come lo sguardo è stato distribuito sull' immagine. Una heatmap, infatti, sarà colorata con colori più caldi (rosso, arancione, giallo, ecc.) nei punti guardati con maggiore intensità (in termini di tempo), via via andando fino ai colori più freddi (verde, azzurro, blu, ecc.) che invece indicano zone di attività minore. Da notare che non esistono soglie di tempo oltre le quali si raggiunge un determinato colore, ma i tempi vengono normalizzati per ogni immagine.

Per ogni immagine vengono poi trovate le componenti connesse scartando i simboli di interpunzione (punti, virgole e punto-e-virgola), i pallini sulle i, gli apostrofi, gli accenti ecc. Così facendo, ogni componente connessa trovata corrisponderà ad una lettera del testo; ciò permette di definire una corrispondenza univoca fra lettere del file txt e componenti connesse trovate nell'immagine.

Sarà inoltre possibile raggruppare le componenti connesse relative alle lettere di una stessa parola; questo è necessario per la produzione dei documenti finali.

A questo punto si procede trovando, per ogni fixation, la corrispondente componente connessa, ovvero quale è la lettera che è stata effettivamente guardata, e a quale parola questa appartiene.

Vengono così creati i documenti finali:

3. **Vista** file csv. Vengono registrate le parole guardate per ogni fixation. Ogni riga conterrà: il numero della fixation, l'indice della componente associata ovvero l'indice della parola letta, la parola corrispondente e l'indice della precisa lettera letta all'interno della parola considerata.

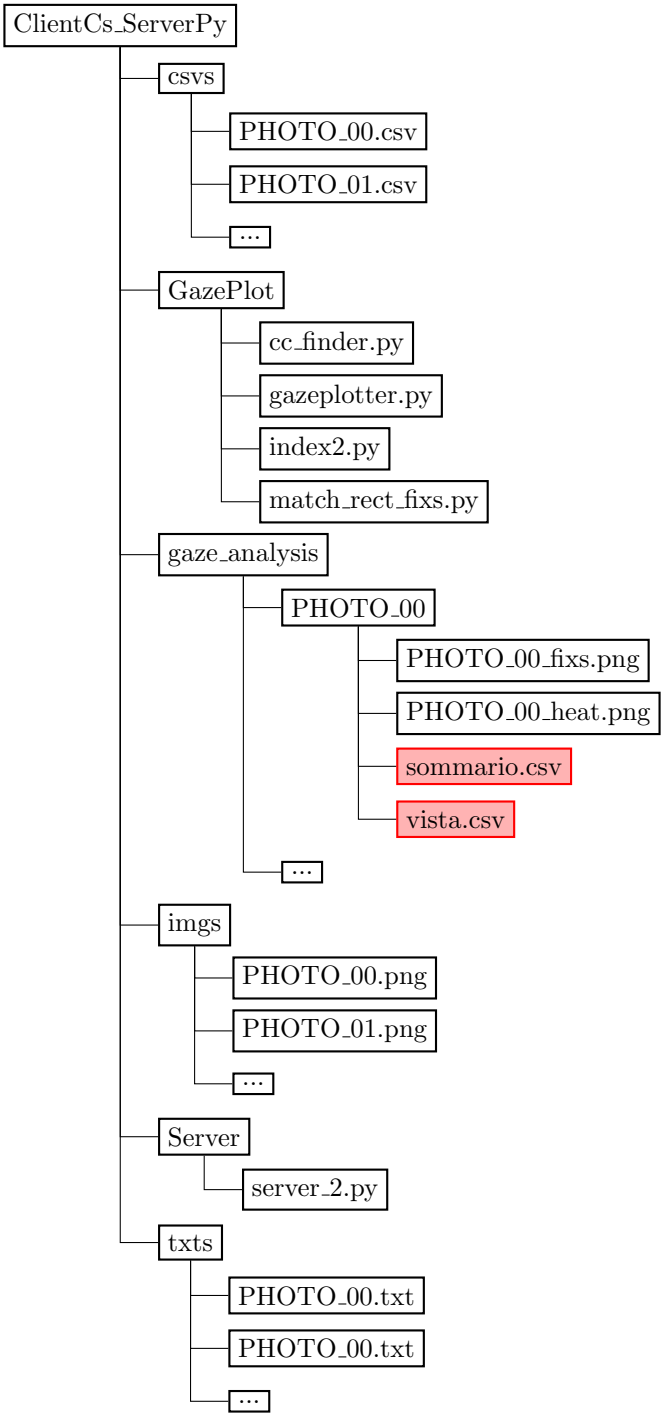

Figura 2: Struttura delle cartelle, visulizzazione ad albero (tree)

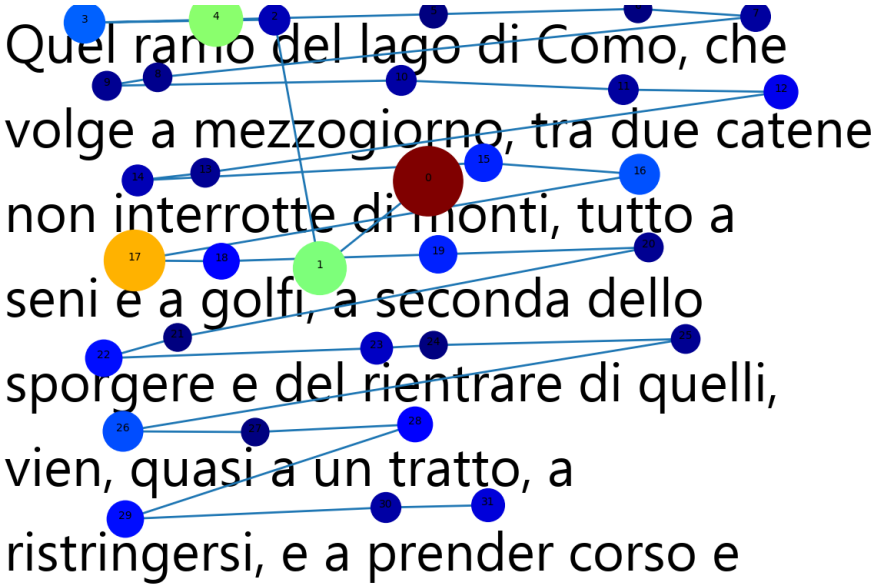

Figura 3: Esempio di gaze plot

(es. [0, 14, ciao, 2] per indicare che la prima fixation riguarda "ciao", che è la 14-esima parola all'interno del testo, e nello specifico è stata osservata la lettera numero 2, cioè "a");

4. **Sommario** file csv. Qui vengono registrate le fixation fatte per ogni parola del testo, quindi, ogni riga conterrà: l'indice della componente (parola) considerata, la componente stessa ed una lista degli indici delle lettere della parola che sono state lette. Nel caso in cui la parola non sia mai stata letta si troverà una lista vuota.

   (es. [14, ciao, [2, 3]] per indicare che della parola "ciao", 14-esima parola all'interno del testo, sono state osservate le lettere di indice 2 e 3 , cioè "a" ed "o");

5.5 Visualizzazione dati

Nel progetto github (link al progetto)[8] vi è tutto il codice esempio per la generazione dei dati aggregati. Vi è inoltre la cartella gaze_analysis_example nella quale si trovano i dati a seguito di un esecuzione del progetto. Si riportano in questa Sezione degli esempi da quella cartella.

La Figura 3 mostra un esempio di gazeplot, dove ogni circonferenza è centrata nel punto di fixation, la dimensione ed il colore indicano quanto tempo è durata ed i numeri scritti in sovraimpressione rappresentano la cronologicità delle fixation successive.

Figura 4: Esempio di heatmap

La Figura 4 rappresenta un esempio di heatmap, costruita sui soliti valori di fixation sui quali è stata costruito il gazeplot. Questi usano un kernel gaussiano per la rappresentazione dei colori e la sovrapposizione degli stessi.

La Tabella 2 offre una visione del contenuto del file **vista.csv**. Questo rappresenta per ogni fixation le informazioni principali come la parola coinvolta, la posizione della parola nel testo dell'immagine e la posizione del carattere osservato. Ovviamente se di una parola sono stati visualizzati due caratteri, vi saranno due righe con la componente **parola** uguale.

La Tabella 3 offre una visione del contenuto del file **sommario.csv**. Questo rappresenta per ogni parola le informazioni principali delle fixation, nonchè la posizione della parola nel testo dell'immagine, la parola stessa e una lista contenenti le posizioni dei caratteri osservati.

6 Analisi future

Gli sviluppatori restano a disponibilità per la visione del codice, per la proposta di idee riguardo a studi e sviluppi futuri che comprendano l'utilizzo di quest'elaborato finale. Si rimanda per il codice stesso al link su github.

id fixation	pos. parola	parola	pos. carattere
0	16	monti	0
1	22	golfi	4
2	2	del	0
3	0	Quel	2
4	1	ramo	2
5	3	lago	3
6	5	Como	2
7	6	che	1
8	1	ramo	1
9	0	Quel	2
10	3	lago	2
11	11	due	0
12	12	catene	2
13	9	mezzogiorno	0
14	14	interrotte	1
15	9	mezzogiorno	10
...	...	...	...

Tabella 2: Esempio di file **vista.csv**

id parola	parola	lista caratteri con fixation
0	Quel	”[2.0]”
1	ramo	”[1.0, 2.0]”
2	del	”[0.0]”
3	lago	”[2.0, 3.0]”
4	di	”[]”
5	Como	”[2.0]”
6	che	”[1.0]”
7	volge	”[]”
8	a	”[]”
9	mezzogiorno	”[0.0, 10.0]”
10	tra	”[]”
11	due	”[0.0]”
12	catene	”[2.0]”
13	non	”[]”
14	interrotte	”[1.0, 4.0]”
15	di	”[]”
...	...	...

Tabella 3: Esempio di file **sommario.csv**

Riferimenti bibliografici

[1] Tobii. https://www.tobii.com/.

[2] Tobii eye tracker 4c. https://gaming.tobii.com/product/tobii-eye-tracker-4c/.

[3] Why you should never center align paragraph text. https://uxmovement.com/content/why-you-should-never-center-align-paragraph-text/, 2011.

[4] Centered text vs flush left. https://trevellyan.biz/centered-text-vs-flush-left/, 2017.

[5] Antonio Calvani and Luciana Ventriglia. *Insegnare a leggere ai bambini. Gli errori da evitare.* Carocci Editore, 2017.

[6] Rossana De Beni, Lerida Cisotto, and Barbara Carretti. *Psicologia della lettura e della scrittura. L'insegnamento e la riabilitazione.* Erickson, 2001.

[7] Keith Rayner, Alexander Pollatsek, Jane Ashby, and Charles Clifton Jr. *Psychology of Reading.* Psychology Press, 2012.

[8] E. Vivoli and F. Del Lungo. Link github del progetto. https://github.com/emanuelevivoli/ClientCs_ServerPy, 2019.