Diario di lavoro

|  |  |
| --- | --- |
| Luogo | Canobbio |
| Data | 17.02.2023 |

|  |
| --- |
| Lavori svolti |
| Installato lxml al posto di bs4 perché bs4 troppo grande e pesante, per leggere semplicemente una pagina si può usare lxml.  Tornati su BeautifulSoup perché più comodo ed è già installato.  Implementato il codice per permettere le richieste alle pagine web (sia con proxy sia senza)  Modificati dei font nell’interfaccia grafica <https://kivy.org/doc/stable/api-kivy.metrics.html>  Installato opencv <https://pypi.org/project/opencv-python/>  <https://www.tutorialspoint.com/how-to-compute-the-area-and-perimeter-of-an-image-contour-using-opencv-python>  Creato un file image.py che crea una finestra con l’immagine modificata.  Provato questo metodo per prendere i bordi delle forme dell’immagine passata, modificato il primo esempio presente per renderlo adatto al nostro progetto.  Creata una condizione per decidere se disegnare o no il bordo.  Implementato il codice di image.py nel main.  Creata una nuova schermata che conterrà l’immagine con i bordi disegnati per decidere quali parti delle immagini vanno mantenute e quali no.  Creato un algoritmo per conoscere l’enfasi delle parole.  Questo algoritmo ordina anche dal + al –  Creato file conter.py per memorizzare dove si preme con il cursore  Collegato con un’immagine e quando viene premuta l’immagine viene stampato a terminale la size dell’immagine e la posizione del cursore |

|  |
| --- |
| Problemi riscontrati e soluzioni adottate |
| Soluzione del problema dei “pip install” 🡪 Monga ha già impostate le variabili d’ambiente per il proxy, mentre Curiale e Ratti devono far partire ogni volta il file setenv.bat  Quando proviamo ad aprire una pagina web per leggere le parole il proxy ci blocca, abbiamo dovuto fare in modo che, prima di leggere la pagina, controlli se c’è bisogno o no delle variabili d’ambiente. Se sono necessarie le aggiunge nella richiesta alla pagina.  All’interno della pagina usata per il codice per il riconoscimento del bordo ci sono 2 esempi, il primo prende solamente il primo bordo che trova mentre il secondo li dovrebbe prendere tutti.  Il secondo codice non funziona perfettamente mentre il primo sembra essere quello adatto, solo che prende il primo bordo che trova. Per evitare questo problema abbiamo aggiunto un ciclo che prende e disegna tutti i bordi che trova.  I bordi che trova non sempre sono abbastanza grandi quindi, prima di disegnarli, controlliamo la grandezza dell’area.  Il codice di image.py crea una nuova finestra con l’immagine modificata, al momento dell’implementazione di questo codice nel main.py esso crea una nuova finestra con l’immagine. Per risolvere questo problema abbiamo creato fatto in modo di creare una nuova immagine con i bordi disegnati che l’algoritmo ha trovato, questa immagine viene poi in seguito letta quando si accede alla schermata di modifica dell’immagine.  Se si scrive prima all’interno del text input e poi si cambia il tipo di input non prende automaticamente i valori scritti, abbiamo fatto in modo che, ad ogni cambiamento del testo nello Spinner della scelta dell’input richiami automaticamente il metodo per cambiare l’effettivo tipo di input. |

|  |
| --- |
| Punto della situazione rispetto alla pianificazione |
| Siamo più avanti rispetto a quello che abbiamo pianificato |

|  |
| --- |
| Programma di massima per la prossima giornata di lavoro |
| Continuare lo sviluppo per l’algoritmo della scelta delle parti da mantenere |