

System do odczytywania hieroglifów

Wykonali :

Dawid Kubacki

Jakub Jabłoński

Spis treści

1.	Wstęp.....	3
1.1.	Wprowadzenie	3
1.2.	Cel pracy	3
1.3.	Zakres pracy	3
2.	Rozwinięcie	4
2.1.	Wykorzystane technologie	4
2.2.	Przygotowanie danych	4
2.3.	Działanie programu	5
2.4.	Przykładowe wyniki	7
3.	Podsumowanie	8
	Bibliografia.....	9

1. Wstęp

1.1. Wprowadzenie

Hieroglify, czyli najstarsze pismo Egipcjan, stosowane były do zapisywania tekstów religijnych, administracyjnych oraz monumentalnych inskrypcji w starożytnym Egipcie. Były także wykorzystywane na papirusach oraz innych materiałach piśmienniczych w celach administracyjnych i handlowych.

Pismo egipskie łączyły elementy pisma fonetycznego (reprezentującego dźwięki mowy, np. sylaby lub spółgłoski), ideograficznego (przedstawiającego konkretne idee lub przedmioty za pomocą symboli) oraz determinatywów (znaków kontekstowych), co czyni je wyjątkowo trudnymi do odczytania.

1.2. Cel pracy

Celem projektu jest opracowanie systemu, który umożliwi automatyczne odczytywanie i interpretację egipskich hieroglifów, jednego z najstarszych i najbardziej złożonych systemów pisma w historii ludzkości.

1.3. Zakres pracy

Zakres pracy :

- analiza systemu pisma hieroglificznego;
- zebranie danych dotyczących alfabetu;
- implementacja programu, który umożliwi rozpoznawanie hieroglifów i ich tłumaczenie;
- testowanie działania programu.

2. Rozwinięcie

2.1. *Wykorzystane technologie*

W projekcie zastosowano kilka technologii oraz bibliotek, które umożliwiły realizację projektu. Poniżej znajduje się opis kluczowych narzędzi oraz technologii użytych w projekcie:

Python został wykorzystany w projekcie ze względu na swoją wszechstronność, dużą ilość bibliotek oraz czytelną składnię. Pozwolił na szybkie prototypowanie oraz implementację algorytmów przetwarzania obrazów.

- Biblioteka NumPy ułatwiła przetwarzanie danych numerycznych i macierzowych, co było kluczowe podczas manipulacji obrazami zapisanymi w formie tablic wielowymiarowych.
- Biblioteka Matplotlib została wykorzystana do wizualizacji wyników detekcji. Umożliwiła rysowanie obrazów oraz oznaczanie wykrytych liter prostokątami, co ułatwiało analizę wyników.
- Główną biblioteką użytą do przetwarzania obrazów była scikit-image. Najważniejsze wykorzystane metody to:
 - `oimread` do ładowania obrazów,
 - `omatch_template` do porównywania wzorców (ang. template matching),
 - `opeak_local_max` do detekcji lokalnych maksimów w wynikach dopasowania wzorców. Dzięki niej możliwe było przeprowadzenie dokładnej analizy obrazów i lokalizacja liter.
- Moduł `argparse` umożliwił wprowadzenie elastyczności w użytkowaniu skryptu poprzez definiowanie parametrów wejściowych, takich jak próg detekcji (`threshold`) oraz współczynnik skalowania (`ratio`).
- Dekorator `@dataclass` pozwolił na łatwe zarządzanie danymi obrazów, zapewniając przejrzystość oraz wygodę w ich obsłudze.

2.2. *Przygotowanie danych*

Przygotowanie danych polega na wyodrębnieniu hieroglifów z obrazu (Rys.2.1) oraz przyporządkowaniu im odpowiednich transliteracji. Proces obejmuje zebranie obrazu hieroglifów, następnie przeprowadzenie wstępnej obróbki, takiej jak poprawa kontrastu i segmentacja, aby wyróżnić poszczególne znaki. Dane te są następnie organizowane w bazę, umożliwiając automatyczne rozpoznawanie i interpretację tekstów.

Kiedy obrazy są już wczytane, kod przechodzi do procesu ich analizy. Dla każdego obrazu tekstowego, kod próbuje wykryć litery z przygotowanego zestawu. W tym celu każda litera (wzorzec) jest dopasowywana do obrazu tekstowego za pomocą funkcji `match_template` z biblioteki `skimage.feature`. Funkcja ta generuje mapę korelacji, która wskazuje, jak dobrze dany wzorzec pasuje do różnych fragmentów obrazu tekstowego. Następnie za pomocą funkcji `peak_local_max` wyszukiwane są lokalne maksima w tej mapie, które odpowiadają potencjalnym pozycjom liter. Aby uniknąć błędów w detekcji, w funkcji tej wykorzystuje się próg korelacji (określony przez argument `--threshold`).

Jeśli dla danego wzorca wykryto jakiekolwiek pozycje, informacje o nich (współrzędne i nazwa litery) są zapisywane w słowniku. Słownik ten przechowuje współrzędne liter jako klucze i ich nazwy jako wartości. Na koniec współrzędne są sortowane według osi X (położenie w poziomie), aby litery były odczytywane w kolejności, w jakiej pojawiają się w tekście. Na podstawie tej posortowanej kolejności kod odtwarza słowo.

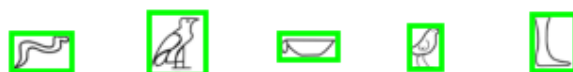
```
for text in text_images:
    letters_positions: dict[tuple[int, int], str] = {}
    for letter in letters:
        result: np.ndarray = match_template(text.data, letter.data)
        peaks: np.ndarray = peak_local_max(result,
                                           threshold_abs=args.threshold,
                                           min_distance=10)
        if len(peaks) == 0:
            continue
        for y, x in peaks:
            letters_positions[(x, y)] = letter.name
```

Dla każdego obrazu wyniki są wizualizowane. Wyświetlane jest okno zawierające analizowany obraz wraz z zaznaczonymi literami oraz wypisane całe słowo.

2.4. Przykładowe wyniki

Na rysunku 2.5 oraz 2.6 zostały przedstawione translacje wybranych ciągów znaków.

JAKUB



Rys.2.5 Przykład działania programu dla słowa „JAKUB”

DAWID



Rys.2.6 Przykład działania programu dla słowa „DAWID”

3. Podsumowanie

Dzięki dokładnemu przygotowaniu danych, w tym wyodrębnieniu znaków z obrazów oraz przyporządkowaniu im transliteracji, system jest w stanie analizować i interpretować teksty starożytnego Egiptu.

Bibliografia

<https://zamek-lublin.pl/magia-starozytnego-egiptu/pismo-starozytnych-egipcjan/>

<https://jakadatoursegypt.com/hieroglyphic-alphabet/>

https://scikit-image.org/docs/stable/auto_examples/segmentation/plot_peak_local_max.html

https://scikit-image.org/docs/stable/auto_examples/features_detection/plot_template.html

Collier, Manley, 'How to read Egyptian hieroglyphs' British Museum Press