Zespołowy Projekt Informatyczny

**Dokumentacja projektu**

Temat:

**Detektor tablic rejestracyjnych**

Skład zespołu:

1. Sławomir Piotrzkowski - lider Ocena:
2. Natalia Nowicka Ocena:
3. Marta Piotrowska Ocena:

Prowadzący: dr inż. Janusz Dorożyński

Spis treści

[I. Syntetyczny opis tematu projektowego 3](#_Toc500958147)

[II. Podział pracy w zespole 3](#_Toc500958148)

[III. Lokalizacja dokumentacji i kodu projektu 3](#_Toc500958149)

[IV. Specyfikacja zastosowanych metodyk projektowych/implementacyjnych 3](#_Toc500958150)

[V. Specyfikacja zastosowanych metodyk testowania i zarządzania jakością 3](#_Toc500958151)

[VI. Specyfikacja zastosowanych narzędzi/środowisk implementacyjnych 3](#_Toc500958152)

[VII. Pracochłonność projektu 3](#_Toc500958153)

[1. Oszacowanie metodą punktów funkcyjnych 3](#_Toc500958154)

[2. Faktyczna pracochłonność 3](#_Toc500958155)

[VIII. Wykres Gantta/harmonogram 3](#_Toc500958156)

[IX. Projekt systemu z zastosowaniem wybranej metodyki 4](#_Toc500958157)

[X. Opis systemu/instrukcja użytkownika 4](#_Toc500958158)

[XI. Podsumowanie 4](#_Toc500958159)

# Syntetyczny opis tematu projektowego

Celem projektu jest stworzenie systemu detekcji i odczytu tablic rejestracyjnych na obrazach przy użyciu języka Python. System będzie wykorzystywał bibliotekę OpenCV do wykrywania tablic rejestracyjnych na obrazach oraz bibliotekę Tesseract do odczytywania znaków z tych tablic.

Projekt składa się z kilku głównych etapów. Na początku zostanie zaimplementowany moduł detekcji tablic rejestracyjnych, który wykorzysta algorytmy przetwarzania obrazu dostępne w bibliotece OpenCV. Moduł ten będzie odpowiedzialny za lokalizację i wyodrębnienie tablic rejestracyjnych na zdjęciach.

Następnie, po wykryciu tablic rejestracyjnych, zostanie użyta biblioteka Tesseract w celu odczytania tablicy. Tesseract jest potężnym narzędziem do rozpoznawania znaków, które można skonfigurować i dostosować do konkretnych wymagań projektu.

W ramach projektu istotne będzie także przetestowanie systemu na różnych zestawach danych testowych, aby ocenić jego skuteczność i dokładność. Będzie to wymagać przygotowania odpowiedniego zbioru testowego.

Projekt detektora tablic rejestracyjnych ma na celu zastosowanie technik przetwarzania obrazu i rozpoznawania znaków w praktycznym zastosowaniu. Może zostać wykorzystany w różnych dziedzinach, takich jak monitorowanie ruchu drogowego, zarządzanie parkingami, identyfikacja pojazdów itp.

# Podział pracy w zespole

Większa część projektu była robione wspólnie. Niewielką część zadań rozdzielono na poszczególne osoby. Poniżej znajduję się lista zadań, wraz z osobami do nich przydzielonymi.

|  |  |
| --- | --- |
| Zadanie | Przydzielone osoby |
| Analiza wymagań: Przeprowadzenie szczegółowej analizy wymagań projektu, w tym zrozumienie funkcjonalności, ograniczeń i oczekiwanych rezultatów. | Natalia, Marta, Sławek |
| Planowanie projektu: Określenie harmonogramu projektu, ustalenie priorytetów, zasobów i odpowiedzialności dla poszczególnych zadań. | Natalia, Marta, Sławek |
| Przygotowanie środowiska pracy: Skonfigurowanie odpowiedniego środowiska programistycznego, w tym instalacja Pythona, bibliotek OpenCV, Tesseract, Gradio, Jupyter Notebook oraz Visual Studio Code. | Natalia, Marta, Sławek |
| Zdobywanie wiedzy na temat potencjalnych metod i oraz narzędzi, które można by wykorzystać projekcie. | Natalia, Marta, Sławek |
| Implementacja detekcji tablic rejestracyjnych: Rozpoczęcie implementacji modułu detekcji tablic rejestracyjnych, obejmującego przetwarzanie obrazu, wykrywanie konturów, segmentację i rozpoznawanie tablic rejestracyjnych. | Sławek |
| Implementacja odczytu znaków: Przeanalizowanie i zaimplementowanie modułu odczytu znaków, wykorzystując bibliotekę Tesseract lub inne metody OCR (rozpoznawanie optyczne znaków). | Marta |
| Testowanie modułów: Przeprowadzenie testów jednostkowych dla modułów detekcji tablic rejestracyjnych i odczytu znaków, sprawdzenie poprawności ich działania i ewentualna korekta. | Natalia |
| Implementacja interfejsu użytkownika: Rozpoczęcie implementacji interfejsu użytkownika za pomocą biblioteki Gradio, aby umożliwić interakcję użytkownika z detektorem tablic rejestracyjnych. | Natalia |
| Integracja modułów: Po zaimplementowaniu poszczególnych modułów przystąpienie do integracji detekcji tablic rejestracyjnych, odczytu znaków i interfejsu użytkownika, aby stworzyć kompletny system. | Natalia, Marta, Sławek |
| Testy całego systemu: Przeprowadzenie testów integracyjnych i testów całego systemu, sprawdzenie jego poprawności, wydajności i zgodności z wymaganiami. | Natalia, Marta, Sławek |
| Optymalizacja i debugowanie: Zidentyfikowanie i poprawa ewentualnych błędów, optymalizacja wydajności i jakości detekcji oraz odczytu tablic rejestracyjnych. | Natalia, Marta, Sławek |
| Dokumentacja: Dokumentowanie projektu, tworzenie instrukcji obsługi, opis funkcjonalności, instrukcji instalacji i uruchomienia aplikacji. | Marta |
| Ostateczne testy i poprawki: Przeprowadzenie ostatecznych testów, sprawdzenie zgodności z wymaganiami i wprowadzenie ewentualnych ostatnich poprawek. | Natalia, Marta, Sławek |
| Dostarczenie projektu: Ukończenie projektu i dostarczenie finalnej wersji systemu "Detektor Tablic Rejestracyjnych". | Natalia, Marta, Sławek |

# Lokalizacja dokumentacji i kodu projektu

Projekt wraz z dokumentacją został umieszczony na platformie GitHub w udostępnionym publicznie repozytorium. Link znajduje się poniżej.

<https://github.com/slaweksystem/license-plate-recognition>

Dokumentacja oraz kod projektu został również dostarczony na platformie Teams do prowadzącego zajęcia.

# Specyfikacja zastosowanych metodyk projektowych/implementacyjnych

W celu usprawnienia procesu projektowego, oraz odpowiedniego zorganizowania pracy zespołu, zminimalizowaniu ryzyka i zwiększeniu efektywności procesu implementacji wykorzystano poniższe metodyki:

1. Metodyka Agile: Projekt będzie realizowany zgodnie z metodyką Agile, która pozwoli na elastyczne i iteracyjne podejście do procesu tworzenia oprogramowania. Oznacza to, że prace będą podzielone na krótkie iteracje, nazywane sprintami, z regularnymi spotkaniami zespołu w celu omówienia postępu i dostosowania priorytetów.
2. Podział na moduły: Projekt zostanie podzielony na logiczne moduły, które będą realizować różne zadania, takie jak detekcja tablic rejestracyjnych, odczyt znaków, testowanie, interfejs użytkownika itp. Każdy moduł będzie miał dobrze zdefiniowane zadania i interfejsy, co ułatwi współpracę w zespole.
3. Wykorzystanie repozytorium kodu: Zespół będzie korzystał z systemu kontroli wersji Git. Repozytorium kodu zostanie zamieszczone na GitHubie ( <https://github.com/slaweksystem/license-plate-recognition> ) do przechowywania i zarządzania kodem projektu. Pozwoli to na śledzenie zmian, łatwe wprowadzanie poprawek i integrację zmian dokonywanych przez różne osoby.
4. Praca zespołowa: Projekt będzie realizowany jako praca zespołowa, co wymagać będzie efektywnej komunikacji i współpracy między członkami zespołu. Regularne spotkania będą miały na celu synchronizację postępów, omówienie problemów i planowanie dalszych kroków.
5. Dokumentacja: W trakcie projektu zostanie stworzona dokumentacja, która będzie opisywać wszystkie etapy procesu projektowego, wykorzystane metody, konfiguracje środowiska, a także instrukcje dotyczące instalacji, uruchamiania i korzystania z aplikacji.

# Specyfikacja zastosowanych metodyk testowania i zarządzania jakością

1. Testowanie jednostkowe: Przeprowadzane będą testy jednostkowe, które sprawdzą poprawność działania poszczególnych modułów projektu, takich jak detekcja tablic rejestracyjnych i odczyt znaków. Testy te zostaną wykonane na przykładowych danych wejściowych i porównane zostaną oczekiwane wyniki z rzeczywistymi wynikami.
2. Testy z wykorzystaniem puli zdjęć: Wykorzystana zostanie określona pula zdjęć zawierających tablice rejestracyjne, na której przeprowadzane będą testy. Będą one obejmować różne warunki oświetleniowe, perspektywę, skalę i inne czynniki wpływające na jakość obrazu. Testy te pozwolą ocenić skuteczność i dokładność detekcji oraz odczytu tablic rejestracyjnych na różnych przykładach.
3. Zarządzanie jakością: W trakcie implementacji projektu przywiązuję się dużą wagę do wysokiej jakość kodu i oprogramowania. Będzie to obejmować przestrzeganie standardów kodowania, stosowanie dobrych praktyk programistycznych oraz regularne inspekcje kodu. Celem jest zapewnienie czytelności, spójności i wysokiej jakości kodu.
4. Monitorowanie błędów: Podczas testowania aplikacji wszelkie wychwycone błędy będą rejestrowane do późniejszej analizy. Wszelkie problemy będą adresowane i naprawiane na bieżąco.

W przypadku projektu "Detektor Tablic Rejestracyjnych" testowanie będzie skupione na puli zdjęć z tablicami rejestracyjnymi, aby ocenić skuteczność detekcji i odczytu w różnych warunkach.

# Specyfikacja zastosowanych narzędzi/środowisk implementacyjnych

W celu zrealizowania projektu zastosowano następujące narzędzia:

1. Język programowania: Głównym językiem programowania używanym w projekcie jest Python. Python jest popularnym językiem programowania o czytelnej składni i bogatym ekosystemie bibliotek, co czyni go odpowiednim wyborem do implementacji detektora tablic rejestracyjnych.
2. Kontrola wersji: Git jest wykorzystywany do zarządzania kontrolą wersji kodu. Umożliwia śledzenie zmian w kodzie, łatwe wprowadzanie poprawek oraz efektywną współpracę. W tym projekcie wykorzystano platformę GitHub, która dostarcza hosting repozytoriów Git oraz narzędzia do zarządzania projektem.
3. Środowisko programistyczne: Kod źródłowy projektu jest pisany w środowisku programistycznym Visual Studio Code (VS Code). VS Code to popularne, wieloplatformowe środowisko programistyczne, które oferuje bogate funkcje, takie jak podświetlanie składni, debugowanie, integrację z Git i wiele innych.
4. Biblioteka Gradio: Do implementacji interfejsu użytkownika (GUI) została wykorzystana biblioteka Gradio. Jest to biblioteka Python’owa, która umożliwia tworzenie interaktywnych interfejsów użytkownika dla modeli ML i aplikacji. Biblioteka ta oferuje łatwe w użyciu narzędzia do budowania interaktywnych formularzy i wizualizacji wyników.
5. Jupyter Notebook: Do eksploracji danych, testowania kodu i dokumentowania projektu wykorzystano Jupyter Notebook. Jest to środowisko umożliwiające tworzenie i udostępnianie dokumentów, które łączą kod, tekst oraz wyniki obliczeń. Ułatwia to eksperymentowanie, testowanie i dokumentowanie poszczególnych fragmentów kodu w projekcie.

# Pracochłonność projektu

## Oszacowanie metodą punktów funkcyjnych

1. Zadanie detekcji tablic rejestracyjnych: Wysoka pracochłonność może być przypisana do modułu detekcji tablic rejestracyjnych, który obejmuje przetwarzanie obrazu, wykrywanie konturów, segmentację i rozpoznawanie tablic rejestracyjnych. W tym przypadku, pracochłonność może być większa ze względu na złożoność tych operacji.
2. Odczyt znaków: Moduł odczytu znaków również może być pracochłonny, szczególnie jeśli wymaga zastosowania technik przetwarzania obrazu, uczenia maszynowego lub biblioteki OCR, takiej jak Tesseract. Przetwarzanie obrazu, ekstrakcja cech i rozpoznawanie znaków mogą wymagać znaczącej ilości pracy.
3. Interfejs użytkownika: Implementacja interfejsu użytkownika przy użyciu biblioteki Gradio może być mniej pracochłonna, ponieważ biblioteka ta zapewnia prosty sposób tworzenia formularzy i wizualizacji wyników. Jednakże, w zależności od wymagań i funkcjonalności interfejsu, prace nad dostosowaniem go do potrzeb projektu mogą zwiększyć pracochłonność.

<Opis i wyliczenia>

## Faktyczna pracochłonność

<Dane wynikające z realizacji >

# Wykres Gantta/harmonogram.



A picture containing text, screenshot, line, number

Description automatically generated

A picture containing text, screenshot, line, number

Description automatically generated

# Projekt systemu z zastosowaniem wybranej metodyki

A picture containing text, diagram, line, screenshot

Description automatically generated

# Opis systemu/instrukcja użytkownika

Program został napisany w pythonie. Opracowano prosty interfejs użytkownika w bibliotece „Gradio”. Poniżej znajdują się instrukcje jak zainstalować potrzebne biblioteki oraz jak uruchomić i korzystać z programu.

**Instalacja:**

Do uruchomienia aplikacji potrzebny będzie python. Program pisano w pythonie w wersji 3.8.

Wszelkie potrzebne biblioteki są opisane w pliku „requirements.txt”. Można je zainstalować z pliku używając komendy:

python3 -m pip install -r requirements.txt

W przypadku problemów z biblioteką pytesseract należy zainstalować osobno program Tesseract, i dodać lokalizację pliku „tesseract.exe” do zmiennej środowiskowej „Path”

**Używanie Programu:**

Używając pythona należy uruchomić skrypt „start\_gui.py”. To spowoduje utworzenie lokalnej strony, której adres znajdziemy w konsoli:

A screenshot of a computer

Description automatically generated with low confidence

Na Stronie ukarze nam interfejs graficzny. Po lewej stronie możemy przeciągnąć zdjęcie z lokalnego pliku. Po naciśnięciu przycisku „Zatwierdź” po prawej stronie zobaczymy treść tablicy rejestracyjnej, o ile taką uda się znaleźć na zdjęciu. Oto przykładowe zrzuty ekranu:”

A screenshot of a computer

Description automatically generated with medium confidence

Przykład z przerobionym zdjęciem.

A screenshot of a car

Description automatically generated with medium confidence

# Podsumowanie

Projekt "Detektor Tablic Rejestracyjnych" był kompleksowym zadaniem, którego celem było opracowanie systemu do wykrywania i odczytywania tablic rejestracyjnych na obrazach. Projekt został zrealizowany przy użyciu języka Python oraz różnych bibliotek, takich jak OpenCV do detekcji tablic, Tesseract do odczytu znaków oraz Gradio do implementacji interfejsu użytkownika.

W trakcie projektu zastosowano metodyki projektowe, takie jak Agile, które umożliwiły elastyczne podejście do pracy, dostarczanie wartościowych funkcjonalności oraz regularne testowanie i dostosowywanie systemu. W ramach projektu przeprowadzono analizę wymagań, planowanie, implementację poszczególnych modułów oraz testowanie całego systemu.

Projekt wymagał znaczącej pracochłonności, szczególnie w zakresie detekcji tablic rejestracyjnych i odczytu znaków, które były kluczowymi funkcjonalnościami systemu. Wykorzystanie narzędzi takich jak Git i GitHub umożliwiło kontrolę wersji kodu i współpracę w zespole.

Ostateczny rezultat projektu to działający system, który potrafi wykrywać tablice rejestracyjne na obrazach i odczytywać zawarte na nich znaki. System został przetestowany na określonej puli zdjęć i dostarczony wraz z dokumentacją.

Projekt "Detektor Tablic Rejestracyjnych" stanowił wyzwanie, ale również okazję do zdobycia praktycznych umiejętności w dziedzinie przetwarzania obrazu, uczenia maszynowego i rozpoznawania znaków. Dzięki temu projektowi członkowie zespołu mieli możliwość rozwinięcia swoich umiejętności programistycznych oraz pracy w zespole nad kompleksowym projektem informatycznym.

Podpisy zespołu:

Natalia Nowicka

Marta Piotrowska

Sławomir Piotrzkowski