Poprawa jakości zdjęć we współczesnych smartfonach

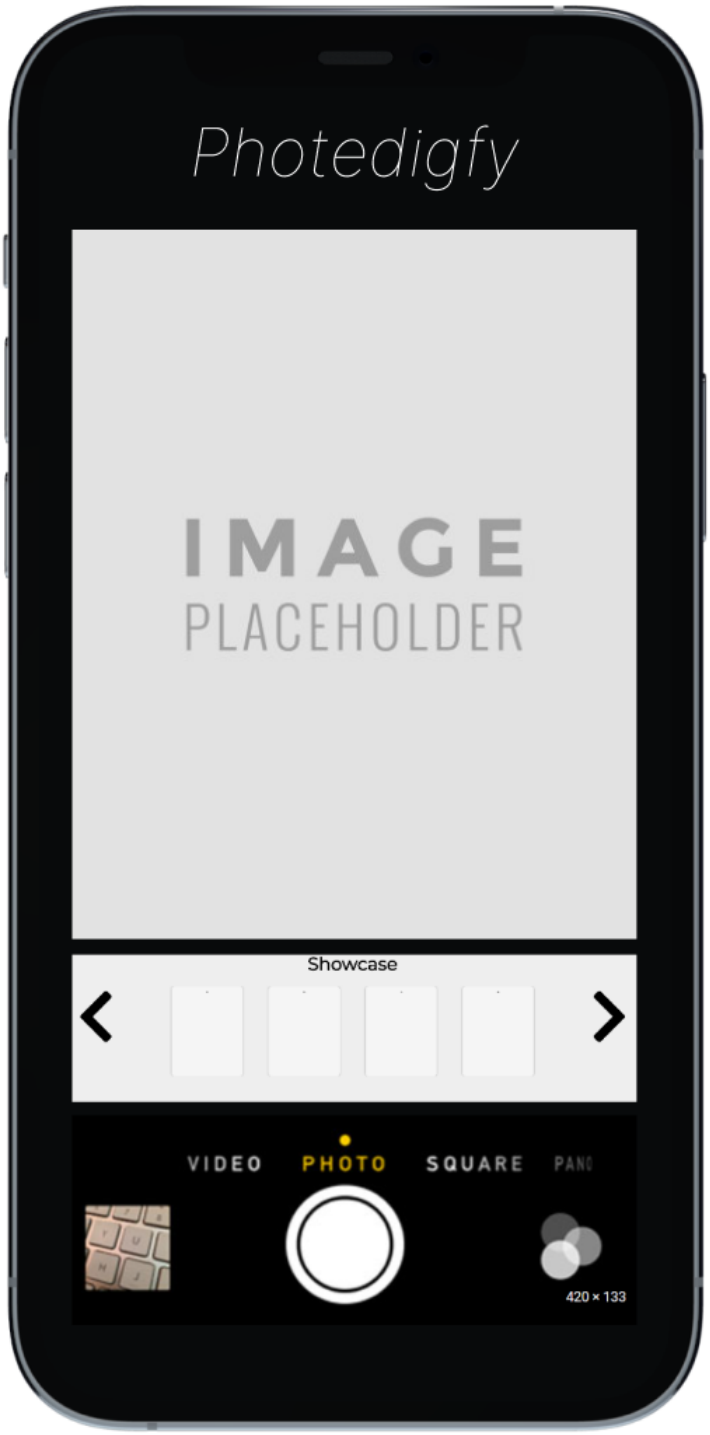
Multimedialne Systemy Interaktywne
Jakub Sachajko, 179976

September 5, 2023

Spis treści

1 Wybór tematu

Tematem niniejszego projektu jest: **"4.Poprawa jakości zdjęć we współczesnych smartfonach"**. Jest to ciekawe zagadnienie, ze względu na wykorzystanie nowoczesnych technologii stosowanych w smartfonach. Dodatkowo składa się na niego również temat obróbki zdjęć, którą można wykonać na wiele różnych sposobów.



2 Analiza wymagań

2.1 Cel aplikacji

Celem aplikacji jest wykorzystanie najnowszych rozwiązań technologicznych zaimplementowanych w aparatach smartfonów typu iPhone oraz wykorzystanie algorytmów do przetwarzania obrazu w celu poprawy jakości obrazu oraz osiągnięcia efektów zmiany stylu zdjęcia tak jak w przypadku filtrów w aplikacji Instagram. Liczba filtrów, która będzie wymagana wynosi 5.
Lista implementowanych filtrów:

- odszumianie zdjęcia na podstawie wykonanej serii zdjęć
- modyfikacja motywu kolorów zdjęcia pierwszego na podstawie innego wybranego zdjęcia
- nakładanie przedmiotów znalezionych dzięki kamerze deep sense na inne zdjęcie
- możliwość zaszumienia drugoplanowych/pierwszoplanowych scen.
- usuwanie przedmiotów znajdujących się na pierwszym planie na podstawie wykonanej serii zdjęć

2.2 Docelowa grupa użytkowników

Biorąc pod uwagę prostotę aplikacji, docelową grupą użytkowników są osoby należące do każdej grupy wiekowej. Aplikacja znalazłaby zastosowanie między innymi u fanów fotografii, ze względu na możliwość poprawy jakości zdjęć, czy u młodzieży chcącej podzielić się ze znajomymi odszumionym zdjęciem dobrej jakości na portalach społecznościowych.

2.3 Docelowa platforma

Docelową platformą jest iPhone posiadający "true depth camera", czyli każdy iPhone w wersji X oraz wszystkie nowsze wersje. Telefon powinien także posiadać appstore w celu zainstalowania aplikacji.

2.4 Środowisko działania aplikacji

Aplikacja będzie działać w środowisku swift posiłkując się biblioteką open cv oraz w przypadku konieczności wykonania trudnych obliczeń serwerem zewnętrznym.

2.5 Określenie wymagań

2.5.1 Funkcjonalne

Stworzenie aplikacji do wgrywania zdjęć i implementacja menu wyboru, ponadto możliwość przetwarzania zdjęć w wybranych wariantach

2.5.2 Wydajnościowe

Aplikacja powinna implementować optymalne algorytmy oraz przerabiać zdjęcia w czasie nie dłuższym niż dwie sekundy

2.5.3 Jakościowe

Aplikacja powinna działać płynnie i nie powinna zamykać się w niewyjaśnionych okolicznościach (zabezpieczenia względem zapytań bez odpowiedzi, klikanie kilku opcji na raz). Serwer powinien sprawnie odpowiadać na zapytania.

2.5.4 Sprzętowe

Aplikacja powinna być dostępna na najnowszych Ipadach oraz na telefonach iPhone w wersji X i wyższej.

2.5.5 Skalowalność

Aplikacja powinna być w stanie obsługiwać praktycznie nieskończoną liczbę osób. Jedynym problemem, czy też wąskim gardłem może być połączenie z serwerem(w przypadku implementacji), które będzie kolejkował zapytania. W razie większej potrzeby przepustowości serwera występuje możliwość wrzucenia serwera na platformę AWS.

2.5.6 Rozszerzalność

Implementacja nowych funkcjonalności nie powinna stanowić żadnego problemu, a nawet jest oczekiwana. Dodatkowo zmiany powinny zostać wprowadzone w przypadku nowych technologii, czy też potrzebie implementacji kreatora filtrów.

2.5.7 Otwartość systemu

Aplikacja będzie ogólnodostępna w sklepie AppStore i będzie bezpłatna. W przypadku chęci modyfikacji kodu, czy też wykorzystania jego części, kod aplikacji będzie dostępny w serwisie GitHub.

2.5.8 Niezawodność

Aplikacja powinna być dostępna praktycznie cały czas. Serwer do obliczeń powinien sprawnie odpowiadać na zapytania. Na szczęście system nie przechowuje trwale danych użytkownika więc nie trzeba robić kopii zapasowych danych, czy też używać baz danych.

2.6 Definicja interfejsów użytkownika

Użytkownik będzie posiadał możliwość zrobienia zdjęcia z podglądem czterech poprzednich edycji. Scroll niżej będzie dawał możliwość zmiany algorytmu, czy też sposobu na odszumianie. Po lewej stronie będzie znajdował się przycisk, który pozwoli użytkownikowi wybrać zdjęcie z galerii. Ponadto, z prawej strony użytkownik będzie posiadał opcję zmiany robienia zdjęcia na nagrywanie filmu. Ostatnią częścią interfejsu będzie przycisk do wykonania zdjęcia.

2.7 Opcjonalne rozwiązania funkcjonalne

Możliwość wgrywania zdjęć oraz filmów poprzez podanie linku do zasobu. Dostawienie serwera do obliczania trudnych operacji (niektórych przekształceń potrzebujących większej mocy obliczeniowej w celu zaoszczędzenia czasu).

2.8 Inne założenia poczynione z użytkownikiem

Chęć wyboru czy dane zdjęcie powinno zostać zapisane w bibliotece czy nie. Aby korzystać z aparatu, czy też zdjęć z biblioteki, użytkownik musi zaakceptować dostęp do podanych zasobów.

2.9 Odporność na awarie, bezpieczeństwo systemu

Za odporność na awarie aplikacji będzie odpowiadał sklep appstore, jedynym zmartwieniem będzie utrzymywanie serwera. W przypadku elementów losowych jak i ataków ddos, serwer będzie musiał zostać zrestartowany aby przywrócić działanie. System nie przechowuje danych, dlatego bezpieczeństwo użytkownika można uznać za bardzo wysokie.

2.10 Określenie niezbędnej dokumentacji

Dokumentacja całego kodu jak i funkcjonalności będzie udostępniona w publicznym repozytorium na platformie GitHub. Confluence nie jest zalecanym rozwiązaniem ze względu na prostotę zastosowanych rozwiązań.

2.11 Zakładane ograniczenia aplikacji

Aplikacja będzie przeznaczona tylko na systemy mobilne z systemem IOS - Android oraz Windows nie będą wspierane. Rozszerzenie aplikacji na inne platformy nie będzie możliwe ze względu na zastosowane technologie.

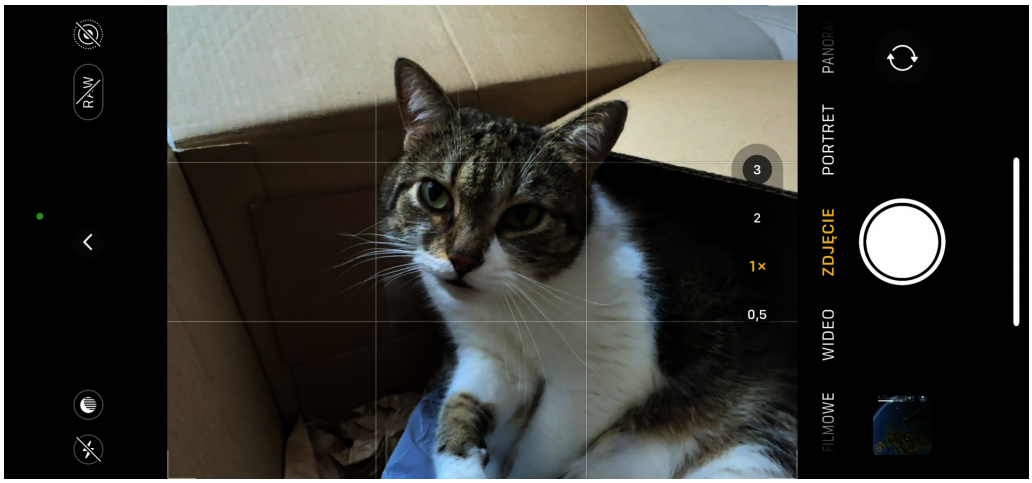
3 Projekt systemu

3.1 Architektura systemu oraz interfejst użytkownika

Interfejs użytkownika powinien być stosunkowo prosty. Będą dostępne dwa ekrany. Pierwszy będzie aparatem, który umożliwi robienie zdjęć. Będzie on pozwalał na:

- zrobienie zdjęcia
- wybranie zdjęcia z biblioteki
- zmianę typu zdjęcia

Są to proste operacje, jednak sa one bardzo przydatne.



Na drugim Ekranie będzie całe centrum aplikacji. Będzie tam można:

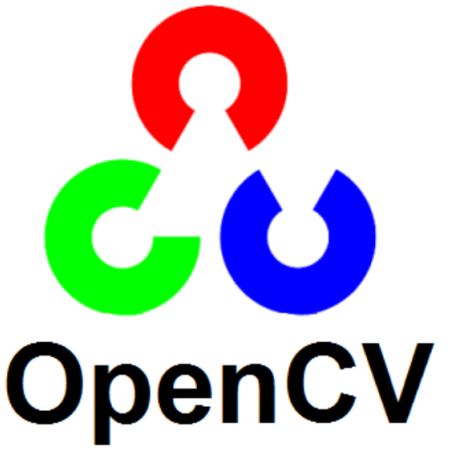
- wybrać sliderem sposób przerabiania zdjęcia
- wybrać zdjęcia zapisane w pamięci
- przejść do ekranu robienia zdjęcia
- przycisk pozwalający na rozpoczęcie przetwarzania zdjęcia

Są to główne zadania. Wszystko będzie odbywało się na telefonie więc architektura będzie stosunkowo prosta. Wszystko przekazywane będzie do poszczególnych funkcji i wywoływane od razu po naciśnięciu przycisku czy też wybraniu opcji. Więc tak naprawdę architektura będzie maksymalnie uproszczona, dzięki czemu dodanie nowej funkcjonalności powinno być stosunkowo proste.

Dodatkowo w pamięci aplikacji trzymane będą ostatnio zrobione czy też wybrane z biblioteki zdjęcia. Ich liczebność nie powinna przekraczać 4 ostatnio używanych + 1 aktualnie edytowanego. Zdjęcia te będą w oryginalnych rozdzielczościach i wszystkie parametry dla struktury pliku zdjęcia mogą zostać wykorzystane w dalszym przetwarzaniu zdjęcia.

3.2 Środowisko tworzenia aplikacji

- Aplikacja będzie tworzona w aplikacji xcode na systemy IOS dla wersji X oraz nowszych modeli, iPad nie będzie brany pod uwagę.
- Główny język programowania w którym będzie wykonywana aplikacja do SWIFT, który dzięki natywnemu przeznaczeniu na urządzenia IOS pozwoli na szybkie operacje w przeciwieństwie do zdobywającego popularność języka dart.
- Wykorzystana zostanie również biblioteka openCV, która zostanie zintegrowana z aplikacją w celu osiągnięcia możliwości szybkiego i dokładnego przetwarzania zdjęć w czasie podchodzącym pod rzeczywisty
- Aplikacja ta będzie testowana przy użyciu emulatora. Dodatkowo wersją alpha będzie udostępniona za pomocą TestFlight dla wszystkich ochotników.



3.3 Projekt najważniejszych struktur danych

W tym projekcie występują dwie ważne struktury danych i są to zdjęcia oraz moduły do przetwarzania zdjęć. Ich pola będą prezentowały się następująco

3.3.1 Zdjęcia

- nazwa
- wysokość
- szerokość
- rozszerzenie
- film czy zdjęcie

3.3.2 moduły do przetwarzania zdjęć

- nazwa
- funkcja robiąca operację na zdjęciu
- dodatkowe zmienne w zależności od potrzeby

3.4 Wykorzystane zasoby

- Biblioteka openCV
- Wbudowane funkcje aparatu
- Format zapisu pliku w języku swift
- Algorytmy potrzebne do operacji na zdjęciach:
  - Synthetic Boken
  - HDR
  - Shallow Boken

3.5 Realizowane główne funkcje systemu

Pierwsze dwie opcje zapewnione są dzięki bibliotekom zawartym w języku swift. Następne opcje natomiast muszą zostać zaimplementowane i wywoływane będą poprzez widgety zawarte w aplikacji. W przypadku wykorzystania modeli do przetwarzania zdjęć integracja na drodze C++ oraz swift może stanowić wyzwanie.

- robienie zdjęć
- ładowanie zdjęć
- przetwarzanie zdjęć
- zapisywanie ostatnich zdjęć
- wykorzystanie modelu do przetwarzania zdjęć

4 Wersja beta

4.1 Aktualny stan zaawansowania projektu

Brak postępu.

4.2 Trudności w realizacji

Zrezygnowałem z pierwotnego planu, który zakładał wykorzystanie bazowego kodu aplikacji, otrzymanego w celu przyśpieszenia podstaw produkcji aplikacji. Po instalacji i zmianie ustawień w celu zapobiegania błędom zatrzymałem się w momencie w którym biblioteki były zbudowane na inne systemy telefonów (jedna na x86 64 druga na arm64). Z czego wynika że w momencie w którym aplikacja powinna już mieć wersję beta nie udało mi się nic zaimplementować. Wynikiem problemu z włączeniem aplikacji plan całkowicie uległ zmianie.

4.3 Zmiany w projekcie

Na ten moment rozważane są sposoby znajdowania pierwszego planu. Pierwotnie funckjonalność trueDepth była przewidywaná podstawą do znajdowania pierwszego planu, jednak od momentu IOS 16.0 Apple wprowadzio funckję "cropowania" zdjęć dzięki prostemu modelowi sztucznej inteligencji. Dlatego sposób rozwiązania problemu nadal jest rozważany.
Budowa interfejsu użytkownika jak i cała aplikacja budowane będą od samego początku ze względu na problemy związane z włączeniem aplikacji pomocniczej.

5 Raport koncowy i dokumentacja projektu

5.1 Stopień realizacji wymagań początkowych

Większość wymagań końcowych zostało wykonanych, jednak niektóre zostały poddane całkowitej modyfikacji. Wytworzona została aplikacja mobilna która posiada podstawowe funkcje:

- robienie zdjęć
- Wybranie zdjęcia z biblioteki
- przetwarzanie zdjęć
- zapisywanie ostatnich zdjęć
- wykorzystanie modelu do przetwarzania zdjęć
- wybór sliderem sposobu przerabiania zdjęcia
- przejść do ekranu robienia zdjęcia
- przycisk pozwalający na rozpoczęcie przetwarzania zdjęcia
- przycisk pozwalający na reset aplikacji

Zrezygnowałem z odtwarzania filmów video ponieważ dodają one poziom trudności do manipulacji na różnych typach danych.

Bazując na założeniach znalezionych w internecie stwierdziłem że użycie modelu mobilenet w celu znajdowania maski i przeklejania pierwszego planu zdjęć będzie znacznie lepszym rozwiązaniem niż AVFoundation dlatego stwierdziłem że zaimplementuję to rozwiązanie. (Ze względu na problemy z libraw stwierdziłem że praca z openCV się nie opłaca, dlatego dla użycia innego sposobu niż model)

Operacje zaimplementowane w celu przetwarzania zdjęć:
1 zdjęcie

- Znajdź maskę osoby
- Znajdź maskę obiektu
- Zaszumienie tła

- Odszumianie

2 zdjęcia

- Przeklejenie pierwszego planu na drugie zdjęie (obiekt)
- Przeklejenie pierwszego planu na drugie zdjęie (osoba)

5.2 Wyniki przeprowadzonych testów wydajnościowych

Moim jedynym założeniem był czas przeprowadzania operacji i jestem w stanie powiedzieć że nawet podczas pierwszego użycia modelu, aplikacja wciąż przeprowadza wszystkie potrzebne operacje w mniej niż 1.2 sekundy w przypadku najgorszym bazowanym na 10 próbach.

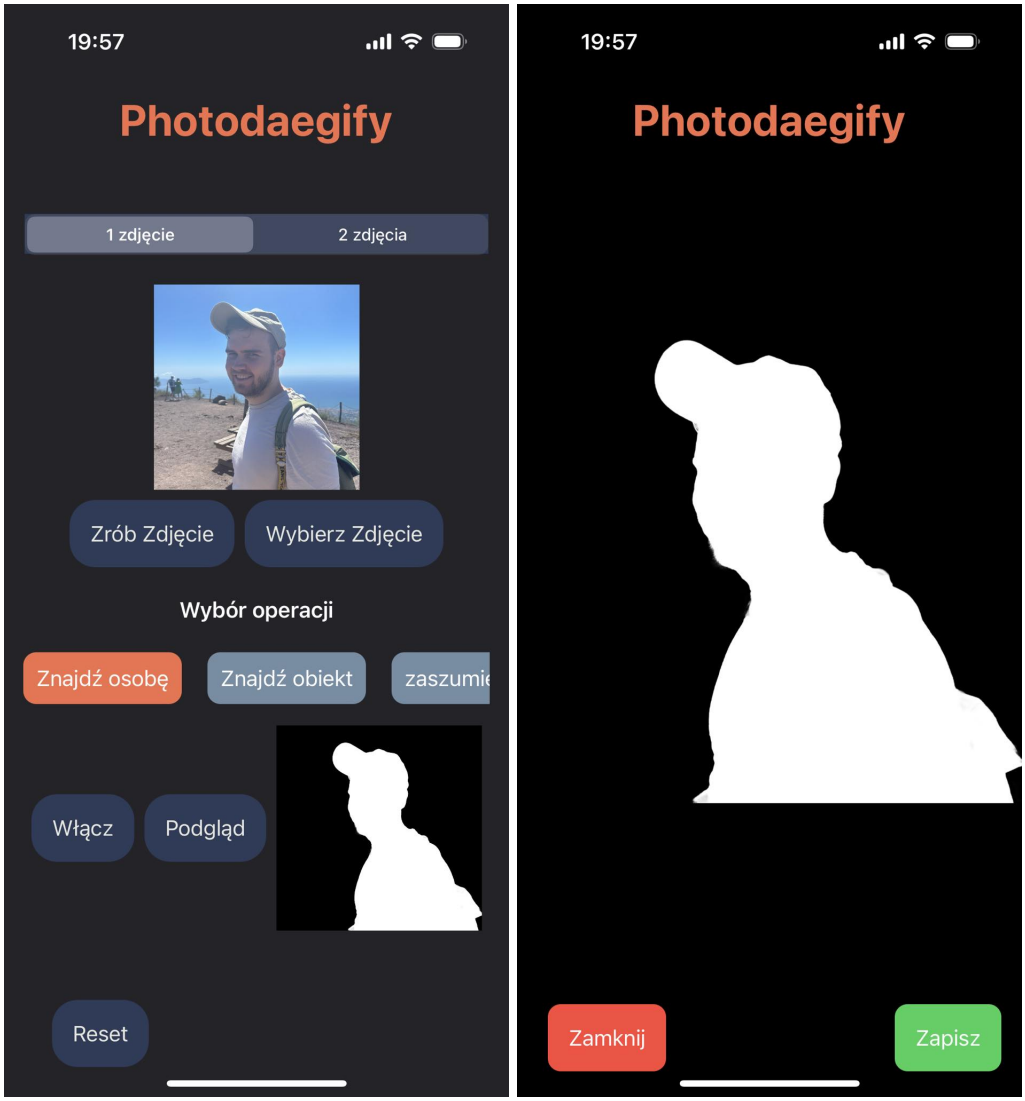
5.3 Raport z walidacji programu w rzeczywistym środowisku docelowym, w tym z działania głównych funkcji systemu

Aplikacja działa w środowisku IOS minimalnie z systemem IOS 15.0. Aplikacja była pisana głównie z myślą o rozmiarach telefonu Iphone 14, więc możliwe że przy próbie użycia na innych wersjach telefonu układ przycisków będzie dosyć problematyczny.

Aplikacja działa bez problemu w środowisku rzeczywistym.

Zdjęcia interfejsu:


19:56
Photodaegify
1 zdjęcie
2 zdjęcia
Zrób Zdjęcie
Wybierz Zdjęcie
Wybór operacji
Znajdź osobę
Znajdź obiekt
zaszumie
Włącz
Reset



Na zdjęcia przedstawionych powyżej można zaobserwować sposób działania aplikacji:

1. Wybór jedno czy dwa zdjęcie
2. wybór zrobienia zdjęcia lub wybrania zdjęcia z biblioteki
3. wybór operacji przeprowadzanej na zdjęciu
4. możliwość resetu wybranych i otrzymanych zdjęć oraz możliwość włączenia operacji
5. opcja podglądu
6. możliwość zapisania lub zamknięcia podglądu

5.4 Raport o wszelkich odstępstwach od założeń i projektu

- Niewykorzystanie biblioteki openCV
- Zrezygnowanie z operacji na plikach video
- brak potrzeby komunikacji z zewnętrznym serwerem
- użycie modelu (segmentation.mlmodelc)
- rezygnacja z wdrożenia do AppStore
- Skalowalność jest niepotrzebna, ponieważ serwer został pominięty

5.5 Informacja o ciekawych obserwacjach, nietypowych rozwiązaniach, błędach wykorzystywanych narzędzi występuj podczas realizacji projektu

Bardzo zainteresowało mnie rozwiązanie z użyciem modelu mobilenet. Głównie ze względu na to że flutter nie posiada takiej możliwości. tensorflow wycofał wersję mobilną więc tak naprawdę wydaje mi się że iphone ma przewagę w wykorzystaniu modeli sztucznej inteligencji zoptymalizowanych pod niewielką ilość własnych urządzeń.

Pisanie aplikacj w języku dart i swift są w miarę podobne, więc po kilku tygodniach użytkowania widzę podobne zależności.

Dokumentacja TrueDepth jest znikoma, więc ciężko używać oraz nauczyć się jak powinna wyglądać implementacja

Ostatnio wprowadzono zmiany symulatorów w środowisku xcode. Teraz wszystkie symulatory działają na systemie arm64, więc budowa na x86 64 nie pozwala na podgląd oraz uruchomienie aplikacji w symulatorze.

5.6 Podsumowanie projektu i wnioski

Jeżeli miałbym określić poziom zadowolenia z finalnego produktu, określiłbym go na poziomie 4/10. Błędy zatrzymywały mnie w każdym możliwym miejscu oraz brak wiedzy z zakresu wybranego projektu okazał się dużym wyzwaniem (Głównie openCV, Swift oraz wiedza podstawowa z zakresu korzystania z truedepth). Patrząc na ten projekt jako próbę nauczenia się wszystkich wymienionych rzeczy, najbardziej zadowolony jestem z poznanych podstaw języka swift. Zrozumiałem także że dokumentacja tak jak i wiele wpisów na stackoverflow są bardzo potrzebne i czasem kiedy społeczność jest niewielka ciężko znaleźć nawet proste rozwiązanie i szuka się go wiele godzin. Chęć używania zewnętrznych serwerów była niepotrzebna co mnie absolutnie zdziwiło, ponieważ użycie modelu bywa często zasobowo oraz czasowo wymagające.

Myślę że pomimo trudności nie wybrałbym innego tematu projektu, ponieważ chciałem się wiele dowiedzieć i to właśnie osiągnąłem.

6 Link do projektu

Link do Projektu: https://github.com/eska5/Photedigfy