

POLITECHNIKA WROCŁAWSKA  
WYDZIAŁ ELEKTRONIKI

---

KIERUNEK: Automatyka i Robotyka (AIR)  
SPECJALNOŚĆ: Systemy informatyczne w automatyce (ASI)

**PRACA DYPLOMOWA  
INŻYNIERSKA**

Zastosowanie algorytmów fotogrametrii dla  
estymacji położenia obserwatora w przestrzeni  
trójwymiarowej

A photogrammetric approach for observer's  
position estimation in 3D

**AUTOR:**  
Lev Sergeyev

**PROWADZĄCY PRACĘ:**  
Dr hab. inż. Przemysław Śliwiński

**OCENA PRACY:**



# Spis treści

<b>1 Wstęp</b>	<b>5</b>
1.1 Problem lokalizacji . . . . .	5
1.2 Cel pracy . . . . .	6
<b>2 Fotogrametria. Wprowadzenie teoretyczne</b>	<b>7</b>
2.1 Wyznaczanie punktów kluczowych. Transformata SIFT . . . . .	8
2.1.1 Filtr Gaussa . . . . .	8
2.1.2 Różnica filtrów Gaussa . . . . .	9
<b>3 Budowa środowiska</b>	<b>11</b>
3.1 Platforma AliceVision . . . . .	11
3.1.1 CameraInit . . . . .	12
3.1.2 FeatureExtraction . . . . .	12
3.1.3 ImageMatching . . . . .	13
3.1.4 FeatureMatching . . . . .	13
3.1.5 StructureFromMotion . . . . .	14
3.2 Implementacja środowiska . . . . .	14
3.2.1 Proces rekonstrukcji sceny . . . . .	15
3.2.2 Odczyt danych sceny . . . . .	15
3.2.3 Rysowanie danych . . . . .	16
3.2.4 Pomiar czasu . . . . .	17
<b>4 Przebieg badań</b>	<b>19</b>
4.1 Odtwarzanie sceny z różną zawartością. Wykrycie i dopasowanie cech . . . . .	20
4.2 Wpływ parametrów rozpoznawania cech na wynik . . . . .	21
4.3 Czas obliczenia . . . . .	22
<b>5 Podsumowanie</b>	<b>33</b>
<b>Literatura</b>	<b>34</b>



# Spis rysunków

2.1	Obraz przed i po rozmyciu Gaussa . . . . .	9
2.2	Schemat stosowania przekształcenia DoG dla przestrzeni skal . . . . .	10
2.3	Obraz przed i po filtracji DoG . . . . .	10
4.1	Kostka Rubika na podstawce, scena "c0", 4 zdjęcia . . . . .	19
4.2	Kostka Rubika z oświetleniem zmiennym, scena "c1", 20 zdjęć . . . . .	19
4.3	Kostka Rubika z oświetleniem punktowym, scena "c2", 10 zdjęć . . . . .	20
4.4	Znaki specjalne, scena "mark", 13 zdjęć . . . . .	20
4.5	Budynek A1 PWr, scena "A1", 19 zdjęć . . . . .	21
4.6	Wybrane punkty charakterystyczne, zdjęcie 03 i 07, scena "mark" . . . . .	21
4.7	Wybrane punkty charakterystyczne, scena "mark" . . . . .	22
4.8	Wybrane punkty charakterystyczne, scena "A1" . . . . .	23
4.9	Wybrane punkty charakterystyczne, zdjęcie 0, scena "A1" . . . . .	24
4.10	Wybrane punkty charakterystyczne, zdjęcie 10, scena "A1" . . . . .	24
4.11	Wybrane punkty charakterystyczne, zdjęcie 15, scena "A1" . . . . .	25
4.12	Odtwarzana scena "c2", parametr: normal . . . . .	25
4.13	Odtwarzana scena "c2", parametr: high . . . . .	26
4.14	Odtwarzana scena "c2", parametr: ultra . . . . .	26
4.15	Odtwarzana scena "mark", parametr: normal . . . . .	27
4.16	Odtwarzana scena "mark", parametr: high . . . . .	27
4.17	Odtwarzana scena "mark", parametr: ultra . . . . .	28
4.18	Odtwarzana scena "A1", parametr: medium . . . . .	28
4.19	Odtwarzana scena "A1", parametr: normal . . . . .	29
4.20	Odtwarzana scena "A1", parametr: high . . . . .	29
4.21	Odtwarzana scena "A1", parametr: ultra . . . . .	30
4.22	Czas wykonania, scena "c0" . . . . .	30
4.23	Czas wykonania, scena "c1" . . . . .	30
4.24	Czas wykonania, scena "c2" . . . . .	31
4.25	Czas wykonania, scena "mark" . . . . .	31
4.26	Czas wykonania, scena "A1" . . . . .	31



# Rozdział 1

## Wstęp

Praca jest podzielona na pięć rozdziałów. W tym rozdziale są rozpatrzone problemy, które spróbowano rozwiązać i pytania na które spróbowano dać odpowiedź. Trzy następne rozdziały opisują przebieg pracy. Ostatni rozdział jest poświęcony analizie przerobionej pracy.

### 1.1 Problem lokalizacji

W wielu sferach życia, jak i w procesach technologicznych występuje potrzeba określania położenia w przestrzeni — lokalizacji. To może dotyczyć jak wyznaczania położenia obserwatora tak i lokalizacji obiektu.

W tej pracy proponowanym rozwiązaniem problemu lokalizacji jest stosowanie **pozycjonowania fotogrametrycznego**. Taki rodzaj lokalizacji polega na szacowaniu pozycji obserwatora (obiektywu kamery) względem sceny na podstawie dwóch lub więcej zdjęć. Pozycjonowanie na podstawie zdjęć przewiduje także obecność ruchomej kamery względem sceny.

Zaproponowane rozwiązanie umożliwiło by lokalizację w takich miejscach i przypadkach, gdzie korzystanie z takich narzędzi i rozwiązań jak na przykład **nawigacja satelitarna**, lokalizacja za pomocą sieci **stacji BTS**, **systemy radiolokacji**, **systemy sonarowe** staje się niemożliwym lub małoskutecznym.

Podejście fotogrametryczne w pozycjonowaniu mogłoby pozwolić na dopełnienie już istniejących systemów lokalizacji w celach zwiększenia dokładności i niezawodności systemu.

Takie podejście też mogłoby pozwolić rozbudowę funkcjonalności lokalizacji systemów już posiadających kamerę. Zaletą takiego rozwiązania może być niski koszt rozbudowy, brak konieczności przerobienia układów systemu, elastyczność konfiguracji.

Przykładem takich systemów może być dron, robot lub pojazd autonomiczny.

Wadą zastosowania pozycjonowania fotogrametrycznego mogą być małe możliwości zasobów obliczeniowych lub wysokie wymagania co do jakości danych.

**Podejście fotogrametryczne** może być zastosowane jak dla **lokalizacji absolutnej** (kiedy znany jest układ odniesienia i skala odległości) tak i **lokalizacji względnej**. W rozprawie, jednak, rozpatrzono tylko lokalizację względową. Oznacza to, że rzeczywista pozycja obserwatora, pozycja sceny, skala odległości pozostają nieznane. W rozdziale 5 będzie zaproponowany sposób przejścia od lokacji względnej do lokacji absolutnej.

## 1.2 Cel pracy

Jednym z zadań pracy była implementacja środowiska, które umożliwia: wyznaczenie pozycji obserwatora względem sceny na podstawie zdjęć, odtworzenie sceny fotogrametrycznej i przedstawienie wyników przetwarzania w czytelny sposób. Dodatkowo środowisko fotogrametryczne powinno umożliwić porównywanie wyników przetwarzania.

Kolejnym zadaniem było użycie gotowego środowiska przedwarzania różnych zestawów danych przy różnych parametrach i przeprowadzić analizę uzyskanych wyników.

W tej rozprawie także podjęto próbę udzielenia odpowiedzi na takie pytania:

- Czy można i jeśli tak, to w jaki sposób wyznaczyć względna pozycję kamery na podstawie zbioru zdjęć?
- Jaki jest algorytm zastosowanego pozycjonowania?
- Jakich rozwiązań można użyć dla estymacji położenia?
- Jakie zasoby są w stanie przeprowadzić obliczenia dla lokalizacji?
- Ile czasu zajmują obliczenia dla otzymania zadowalającego wyniku?
- Jak zależą czas i wynik lokalizacji od danych wejściowych i parametrów fotogrametryzacji?

## Rozdział 2

# Fotogrametria. Wprowadzenie teoretyczne

Fotogrametria - dziedzina nauki, rodzaj sztuki i technologia mająca na celu wyznaczanie informacji i odtwarzanie kształtów jednego lub więcej obiektów fizycznych na podstawie zdjęć obiektów. Fotogrametrię można podzielić na dwa rodzaje: płaską i przestrzenną. Fotogrametria płaska wymaga co najmniej jednego zdjęcia i jej wynikiem jest charakterystyka obiektu w płaszczyźnie. Fotogrametria przestrzenna wymaga dwóch lub więcej zdjęć, jest używana do odtwarzania obiektów w przestrzeni trójwymiarowej.

W tej pracy stosowana jest wyłącznie fotogrametria przestrzenna.

Pozwala ona wyznaczyć takie charakterystyki jak:

- kształt
- rozmiar
- położenie wzajemne w przestrzeni
- kolor i teksturę

Dodatkowo technologia fotogrametrii pozwala na wyznaczenie położenia kamery (obserwatora) w chwili robienia zdjęcia fotogrametrycznego. Położenie kamery w przestrzeni trójwymiarowej jest wyznaczane na podstawie bazy sceny i wybranego zdjęcia zawierającego całość lub część sceny.

**Cecha**, punkt kluczowy — element charakterystyczny obiektu wyróżniający się na tyle, aby można było zidentyfikować ten sam element na zdjęciu z innego ujęcia.

Pojęcie **sceny fotogrametrycznej**, często wykorzystywane w tej rozprawie, oznacza zbiór statycznych obiektów fizycznych, powiązane zdjęcia których są używane do odtwarzania kształtów tego zbioru w przestrzeni trójwymiarowej. Wymaganie dla zbioru zdjęć sceny jest takie, aby zdjęcia zawierały różne ujęcia sceny, przy czym posiadały jeden lub więcej elementów wspólnych i przy takich samych warunkach: oświetlenie, stan sceny, ustawienia kamery i obiektywu, taki sam aparat fotograficzny.

Odtwarzanie kształtów oznacza umieszczenie punktów charakterystycznych obiektów w przestrzeni 3D w taki sposób, aby położenie cech względem siebie odpowiadało położeniu w rzeczywistości. Odbywa się na podstawie wykrycia cech charakterystycznych, w tym ich skali, obrotu i położenia na zdjęciu, a następnie szacowaniu położenia wspólnych cech jednej lub więcej par zdjęć na podstawie informacji o skali, o obrocie i o położeniu cechy na zdjęciu. Proponowane podejście lokalizacji kamery(obserwatora) w chwili zrobienia zdjęcia opiera się o wyznaczeniu pozycji względem cech charakterystycznych obiektu sceny.

Zaproponowany proces fotogrametryczny dla wyznaczania cech sceny i budowy sceny w przestrzeni 3D wraz z lokalizacją kamer odzwierciedla podejście wykorzystane w platformie AliceVision (opisana jest w rozdziale 3.1) dla rozwiązania analogicznego problemu. Proces ten można rozbić na poszczególne etapy, każdy etap jest opisany w rozdziałach 2.1-2.5.

## 2.1 Wyznaczanie punktów kluczowych. Transformata SIFT

Obraz z zestawu zdjęć fotogrametrycznych posiada cechy kluczowe, które mogą być na tyle charakterystyczne, że można z wysokim prawdopodobieństwem je znaleźć w bazie cech innego zdjęcia zawierającego ten sam obiekt.

Jednym ze sposobów wyznaczania i zapisu takich elementów jest stosowanie opatentowanych algorytmów i deskryptorów **SIFT** (Scale Invariant Feature Transform - skaloniezmenna transformata cech, Skaloniezmienne przekształcenie cech). Ta transformata jest powszechnie stosowana w przetwarzaniu i rozpoznawaniu obrazów do wykrywania punktów charakterystycznych.

Zastosowanie takiej transformaty na przeciętnym zdjęciu o rozmiarze 500x500 pikseli może wykryć około 2000 cech charakterystycznych, oczywiście liczba wykrytych elementów zależy jak od zawartości i jakości zdjęcia, tak i od parametrów stosowanej transformaty.

Aby zminimalizować zapotrzebowanie zasobów na wyznaczanie elementów kluczowych obraz jest poddawany szeregu zabiegów optymalizacji i filtracji. Proces wyznaczania cech można przedstawić stosując taki szereg kroków:

1. Obliczenie przestrzeni skal Gaussa
2. Obliczenie DoG (Różnicy filtrów Gaussa)
3. Wyznaczenie punktów pretendujących
4. Lokalizacja punktów kluczowych z subpikselową dokładnością
5. Filtracja punktów niestabilnych pod względem szumu
6. Filtracja punktów niestabilnych co leżą na krawędziach
7. Przydział obrotu(orientacji) dla punktu kluczowego
8. Utworzenie deskryptora punktu kluczowego

W tym rozdziale szczegółowo będą rozpatrzone algorytmy pierwszych dwóch kroków, ponieważ stanowią one podstawę wyznaczania punktów charakterystycznych.

### 2.1.1 Filtr Gaussa

Filtr Gaussa, nazywany także rozmyciem Gaussa, to filtr splotowy, który można zapisać w sposób konwolucji ciągłej:

$$\mathbf{F}_g(\mathbf{u}(\mathbf{x})) = (\mathbf{G}_\sigma * \mathbf{u})(\mathbf{x}) = \int \mathbf{G}_\sigma(\mathbf{x}') \mathbf{u}(\mathbf{x} - \mathbf{x}') d\mathbf{x}' \quad (2.1)$$

gdzie  $\mathbf{u}(\mathbf{x})$ ,  $\mathbf{x} = (x, y) \in \mathbb{R}^2$  obraz wejściowy, a  $\mathbf{G}_\sigma(\mathbf{x}) = \frac{1}{2\pi\sigma^2} e^{-\frac{|\mathbf{x}|^2}{2\sigma^2}}$ ,  $\sigma \in \mathbb{R}^+$  funkcja jądra konwolucji.

W danym przypadku mamy do czynienia z obrazami rastrowymi, więc stosowany jest splot dyskretny. Konwolucja dyskretna takiego filtra z parametrem  $\sigma$  dla obrazu  $\mathbf{u}$  o rozmiarze  $M \times N$  ma postać:

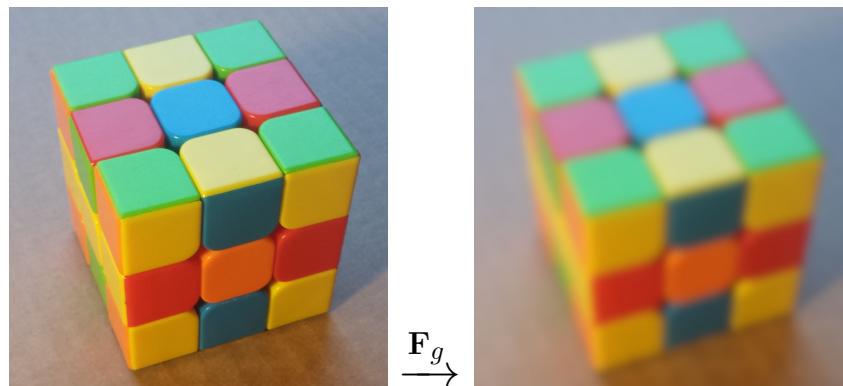
$$\mathbf{F}_g(\mathbf{u}(k, l)) = (\mathbf{G}_\sigma * \mathbf{u})(k, l) = \sum_{k'=-\lceil 4\sigma \rceil}^{\lceil 4\sigma \rceil} \mathbf{g}_\sigma(k') \sum_{l'=-\lceil 4\sigma \rceil}^{\lceil 4\sigma \rceil} \mathbf{g}_\sigma(l') \bar{\mathbf{u}}(k - k', l - l'), \quad (2.2)$$

$$\mathbf{g}_\sigma(k) = K e^{-\frac{k^2}{2\sigma^2}}, -\lceil 4\sigma \rceil \leq k \leq \lceil 4\sigma \rceil, k \in \mathbb{Z}$$

gdzie  $\lceil a \rceil$  — funkcja zaokrąglająca w góre,  $K$  jest takie, że  $\sum \mathbf{g}_\sigma(k) = 1$

$$\begin{aligned} \bar{\mathbf{u}}(k, l) &= \bar{\mathbf{u}}(s_M(k), s_N(l)), \\ s_M(k) &= \min(\text{mod}(k, 2M), 2M - 1 - \text{mod}(k, 2M)), \end{aligned} \quad (2.3)$$

gdzie  $\text{mod}(a, b)$  — reszta z dzielenia  $a$  przez  $b$ .



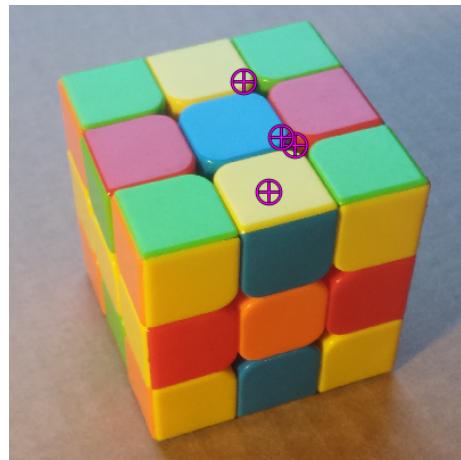
Rysunek 2.1 Obraz przed i po rozmyciu Gaussa

Utworzenie przestrzeni skal w algorytmie **SIFT** polega na stworzeniu rodziny skal  $v$  poprzez kilkukrotnie skalowanie obrazu wejściowego, a następnie kilkukrotniej filtracji każdej skali  $\delta$  z różnym parametrem rozmycia  $\sigma$ . Standardowo krok skalowania wynosi  $S = 2$  ( $M_{n+1} = M_n/2$ ), liczba skal wynosi  $n_\delta = 3$ , a liczba poziomów rozmycia  $n_\sigma = 4$ .

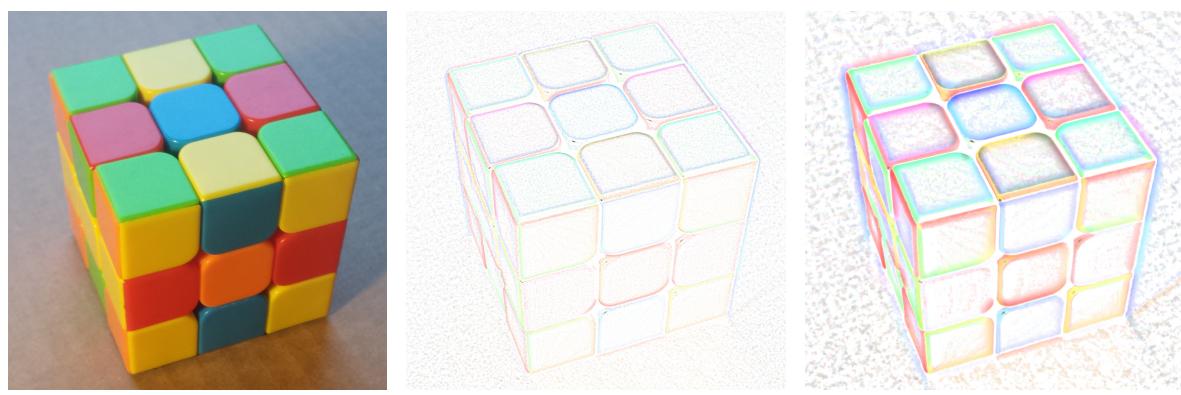
## 2.1.2 Różnica filtrów Gaussa

: Funkcja **DoG** (Difference of Gaussians - Różnica filtrów Gaussa) wyznacza obszary na obrazie, które mogą zawierać elementy kluczowe (krok 2 transformacji **SIFT**). Funkcja **DoG** jest przybliżeniem **LoG** (Laplasjan filtra Gaussa) - funkcją filtrującą, która pozwala wykryć krawędzie. Filtracja obrazu (Rysunek 2.3) wykrywa krawędzie niezależnie od ich skali i obrotu, w taki sposób można odrzucić obszary obrazu, dalsza analiza których jest zbędna. Ponieważ w poprzednim kroku została stworzona przestrzeń skal z różnym rozmyciem, transformacja **DoG** jest obliczana dla każdej kolejnej pary zdjęć w każdym zestawie:

$$\mathbf{DoG}(\mathbf{u}) = \mathbf{G}_{\sigma_n}(\mathbf{u}) - \mathbf{G}_{\sigma_{n+1}}(\mathbf{u}) \quad (2.4)$$



Rysunek 2.2 Schemat stosowania przekształcenia DoG dla przestrzeni skal



(a) Oryginal (b)  $\sigma_1 = 1, \sigma_2 = 4$  (c)  $\sigma_1 = 2, \sigma_2 = 16$

Rysunek 2.3 Obraz przed i po filtracji DoG

# Rozdział 3

## Budowa środowiska

Praktyczna realizacja zaproponowanego rozwiązania składa się z dwóch części: platformy fotogrametrycznej i zestawu skryptów.

### 3.1 Platforma AliceVision

**AliceVision** — framework fotogrametryczny z otwartym kodem źródłowym, udostępniany pod licencją **MPL2**. Platforma została stworzona aby umożliwić odtwarzanie sceny fotogrametrycznej na podstawie zbioru zdjęć lub sekwencji klatek wideo zawierających scenę. Do wybranych możliwości platformy należą:

- Wyznaczanie punktów charakterystycznych — cech sceny
- Parowanie zdjęć — para jest tworzona, gdy zdjęcia zawierają ten sam obiekt
- Zestawienie cech zdjęć
- Odtwarzanie struktury cech w przestrzeni 3D
- Lokalizacja względna i bezwzględna kamery
- Kalibracja kamery
- Utworzenie mapy głębokości sceny
- Odwzorzenie powierzchni obiektów sceny
- Teksturowanie obiektów sceny

Aby wykorzystać framework należy pobrać kod źródłowy projektu i zbudować go w swoim środowisku. Istnieje również możliwość pobrania już gotowego projektu w postaci narzędzia **Meshroom**. Program ten jest swojego rodzaju powłoką graficzną dla **AliceVision** i pozwala w łatwy sposób zapoznać się z możliwościami i sposobem działania platformy. Pliki frameworku znajdują się w katalogu głównym programu.

Zbudowany framework przedstawia sobą zestaw plików binarnych - pliki wykonywalne i biblioteki dynamiczne. Każdy plik wykonywalny przedstawia sobą osobne narzędzie.

Wykorzystanie i komunikacja z **AliceVision** odbywa się w sposób uruchomienia odpowiedniego pliku wykonywalnego w powłoce systemu (np linii poleceń) z odpowiednim zestawem parametrów. Wszystkie operacje odczytu/zapisu odbywają się na plikach, ścieżka których jest określana w parametrach. Na potrzeby pracy wykorzystano następujące narzędzia:

1. **CameraInit**: inicjacja danych, odczyt parametrów kamery,
2. **FeatureExtraction**: wyszukiwanie cech,
3. **ImageMatching**: parowanie zdjęć,
4. **FeatureMatching**: parowanie cech,
5. **StructureFromMotion**: odtwarzanie struktury cech w przestrzeni, lokalizacja kamery.

Narzędzia są opisane w kolejności odpowiedniej do kolejności uruchomienia, taka kolejność zapewnia przetwarzanie potokowe danych. Każde narzędzie jest opisane w poszczególnych rozdziałach pod względem danych wejścia/wyjścia i parametrach w zastosowanych przetwarzaniu danych.

Dla wszystkich wykorzystanych instrumentów można zadać poziom logowania (rejestracji zdarzeń) używając parametru `--verboseLevel`, który przyjmuje wartości (`fatal`, `error`, `warning`, `info`, `debug`, `trace`).

### 3.1.1 CameraInit

`aliceVision_cameraInit` — narzędzie inicjalizacji procesu odtwarzania. Wykrywa i zapisuje listę zbioru zdjęć oraz definiuje parametry kamery, takie jak **długość ogniskową** przeszukując bazę kamer na podstawie modelu odczytanego z metadanych zdjęcia. Istnieje możliwość obliczenia parametrów na podstawie kąta widzenia obiektywu.

- Parametry wejścia:
  - `--imageFolder` — ścieżka folderu zawierającego zestaw obrazów.
- Parametry wyjścia:
  - `--output` — "`cameraInit.sfm`", plik zawierający listę danych i ustawienia początkowe sceny
- Dodatkowe parametry
  - `--sensorDatabase` — baza parametrów kamer, standardowa ścieżka: "`aliceVision/share/aliceVision/cameraSensors.db`",
  - `--defaultFieldOfView` — kąt widzenia obiektywu, użytą kamera ma kąt 56 stopni.

### 3.1.2 FeatureExtraction

`aliceVision_featureExtraction` — narzędzie dla rozpoznawania i utworzenia zbioru deskryptorów cech dla każdego zdjęcia — transformany **SIFT**.

- Parametry wejścia:
  - `--input` — plik "`cameraInit.sfm`"
- Parametry wyjścia:

- **--output** — ścieżka folderu dla przechowania deskryptorów, "FeatureExtraction"
- Dodatkowe parametry
  - **--describerPreset** — ustawienie transformaty **SIFT**, przyjmuje wartości (low, medium, normal, high, ultra)

### 3.1.3 ImageMatching

`aliceVision_imageMatching` — narzędzie dla utworzenia zbioru par zdjęć. Stosowanie tego narzędzia nie jest obowiązkowe. Zalecane jest, jednak, dokonanie parowania zdjęć, gdyż pozwala to ograniczyć liczbę danych wejściowych **FeatureMatching** i tym samym przyspieszyć wykonanie procesu fotogrametrii i zredukować prawdopodobieństwo wystąpienia błędnej pary cech.

- Parametry wejścia:
  - **--input** — plik "cameraInit.sfm"
  - **--featuresFolders** — folder "FeatureExtraction"
- Parametry wyjścia:
  - **--output** — plik przechowujący wykryte pary, "ImageMatching.txt"
- Dodatkowe parametry
  - **--tree** — plik drzewa słownika, standardowa ścieżka:  
"aliceVision/share/aliceVision/vlfeat\_K80L3.SIFT.tree"

### 3.1.4 FeatureMatching

`aliceVision_featureMatching` — narzędzie dokonujące odnalezienia i zapisania par cech dla każdej pary zdjęć.

- Parametry wejścia:
  - **--input** — plik cameraInit.sfm
  - **--featuresFolders** — folder "FeatureExtraction"
- Parametry wyjścia:
  - **--output** — folder dla przechowania wykrytych par cech "FeatureMatching"
- Dodatkowe parametry
  - **--imagePairsList** — jeśli dokonano parowania zdjęć, plik "ImageMatching.txt"

### 3.1.5 StructureFromMotion

`aliceVision_incrementalSfM` — narzędzie dla rekonstrukcji sceny. Pozwala odwzorować położenie punktów charakterystycznych w przestrzeni i odpowiadające im pozycje kamery.

- Parametry wejścia:
  - `--input` — plik `cameraInit.sfm`
  - `--featuresFolders` — folder "FeatureExtraction"
  - `--matchesFolders` — folder "FeatureMatching"
- Parametry wyjścia:
  - `--output` — plik przechowujący rekonstruowaną scenę: cechy i pozycje kamery, "StructureFromMotion.sfm"

Zaletą **AliceVision** można nazwać wieloplatformowość, framework można zbudować dla takich systemów operacyjnych, jak Windows, Linux lub OSX. Do zalet platformy AliceVision można również odnieść ten fakt, iż jej narzędzia zaimplementowane z wykorzystaniem takich technologii, jak na przykład **Mosek**, stosowanie których pozwala zoptymalizować i przyspieszyć wykonanie części algorytmów zwłaszcza algorytmy liniowe. Stosowanie **OpenMP** pozwala na obliczenia wielowątkowe, a **CUDA** czy **OpenCL** umożliwiają przyrost poprzez wykonanie prostych operacji algebraicznych na procesorach graficznych.

Framework powstał i jest rozwijany jako projekt wspólny przez takie centra naukowe, jak:

- Czech Technical University (CTU) in Prague, Czech Republic
- Institut National Polytechnique de Toulouse (Toulouse INP), France
- Mikros Image, Post-Production Company in Paris, France
- Simula Research Laboratory AS in Oslo, Norway
- Quine in Oslo, Norway
- Wspierany przez European Union's Horizon 2020

## 3.2 Implementacja środowiska

Dla realizacji zaproponowanego algorytmu i wyświetlania wyników został napisany zestaw skryptów w języku **Python** wersji 3.8.1 pozwalające na:

Dla skryptów wykonujących wprowadzono system odczytu argumentów zaimplementowany w bibliotece **argparse**. Przykład implementacji takiego systemu:

```
arg_parser = argparse.ArgumentParser()

arg_parser.add_argument('--input', '-i', default="")
arg_parser.add_argument('--select_pose', nargs='+')
arg_parser.add_argument('--marker_size', default=MARKES_SIZE, type=float)
arg_parser.add_argument('--select_features', type=int)

args = arg_parser.parse_args()
```

### 3.2.1 Proces rekonstrukcji sceny

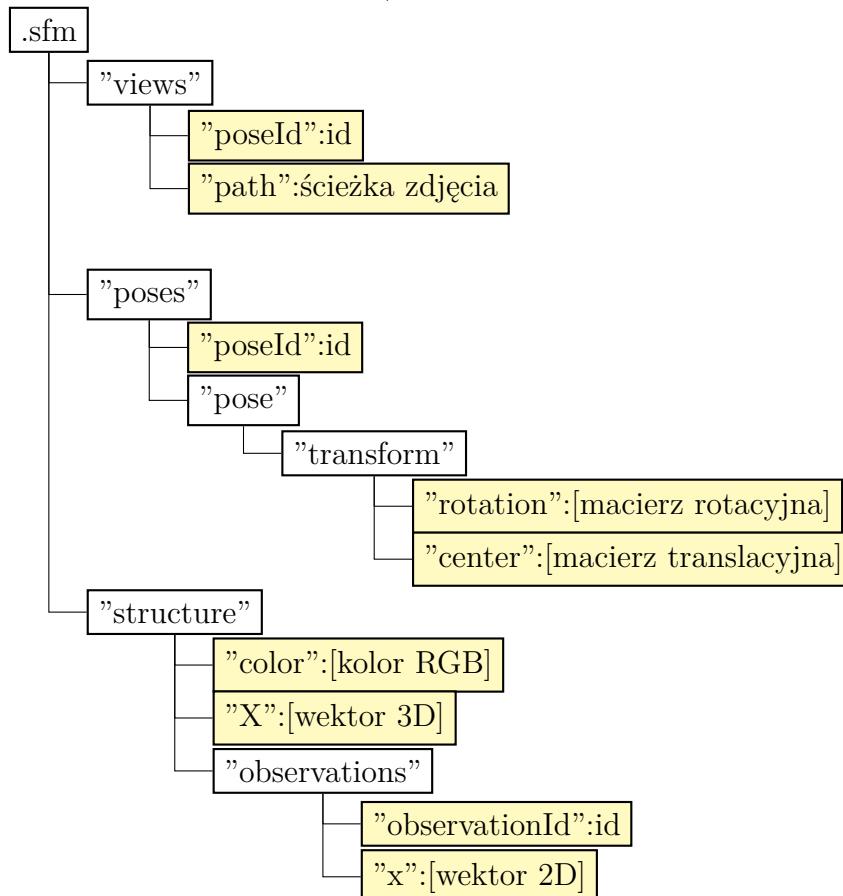
Dla każdego narzędzia platformy **AliceVision** została stworzona funkcja przedstawiająca uproszczony interfejs narzędzia. Przykład funkcji która tworzy polecenie i uruchamia narzędzie frameworku:

```
def cameraInit(
    data_dir,
    cameraInit_file,
    fieldOfView = D_FIELD_OF_VIEW,
    v_level = D_VERBOSE_LEVEL,
    log_dir = D_LOG_DIR
):
    cmd = av + "cameraInit --imageFolder " + data_dir + " -o " +
          cameraInit_file
    cmd += " --defaultFieldOfView " + fieldOfView
    cmd += " --sensorDatabase " + SENSOR_DB
    return run("O_CameraInit", cmd, v_level, log_dir)
```

Wszystkie funkcje narzędzi były połączone w "pipeline" — funkcję która przedstawia cały proces odtwarzania sceny. Wejściem takiej funkcji są zdjęcia sceny i parametry przetwarzania, a wyjściem jest struktura opisująca rekonstruowaną scenę.

### 3.2.2 Odczyt danych sceny

Wynik odtwarzania sceny przez narzędzie **StructureFromMotion** jest zapisywany w postaci drzewa **json** do pliku "**StructureFromMotion.sfm**". Struktura użytych węzłów takiego pliku jest następująca (żółty kolor oznacza węzeł przechowujący dane):



Węzeł ["views"] zawiera zbiór wszystkich zdjęć zestawu, każdemu zdjęciu przydzielony jest numer ujęcia id.

Węzeł ["poses"] zawiera zbiór informacji o wykrytej pozycji. Każda pozycja odpowiada jednemu z zdjęć w zestawie i zawiera macierz rotacji i translacji obserwatora.

Węzeł ["structure"] zawiera zbiór punktów kluczowych. Każdy z tych punktów przechowuje wektor pozycji w przestrzeni, kolor cechy i zbiór pozycji których dotyczy wraz z wektorem położenia na zdjęciu każdej pozycji.

Stosowanie biblioteki **json** pozwala w łatwy sposób odczytać wybrane pola takiego pliku. Przykład kodu wyświetlający ścieżki plików ujęć:

```
with open(input_json, "r") as read_file:
    data = json.load(read_file)

    for item in data['views']:
        print(item['path'])
```

### 3.2.3 Rysowanie danych

Do rysowania sceny 3D, a także rysowania wykresów i punktów na zdjęciach wykorzystana jest biblioteka **matplotlib**

Ponieważ dokładny wektor i kolor cechy są już znane, wystarczy je podać jako argument dla funkcji rysującej zbiór punktów **scatter**. W podobny sposób rysowane są punkty kluczowe na płaszczyźnie zdjęcia(np. Rysunek 4.9).

Aby narysować ostrosłup przedstawiający kamerę, najpierw definiowane są współrzędne wektorów ostrosłupa w punkcie (0,0,0) a następnie wektory są obracane i przesuwane poprzez mnożenie o macierz rotacji i dodanie macierzy translacji odpowiedniej pozycji.

Część kodu rysująca zbiór punktów kolorowych przedstawiających cechy i ostrosłup przedstawiający kamerę na scenie 3D (np. Rysunek 4.12):

```
# Zainicjowac wykres w trybie trojwymiarowym:
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

# Dodac punkty:
ax.scatter(points.xs, points.ys, points.zs, marker='o', c=points.cs, s=point_size)

# Dodac kamere:
# Narysowac uklad wspolrzednych:
ax.quiver(*origin, a[:,0], a[:,1], a[:,2], color='b')

# Narysowac plaszczyzny ostroslupa:
ax.add_collection3d(Poly3DCollection(cam, facecolors='#00ff00',
                                         edgecolors='r', alpha=.25))
ax.add_collection3d(Poly3DCollection(cam_face, facecolors='#ffff00',
                                         edgecolors='r', alpha=.5))

# Podpisac kamere:
ax.text(*txt_pos, name, color='#cc0000', fontsize=14)
```

Funkcja **ax2 = ax1.twinx()** pozwala dodać nowy obszar danych do istniejącego wykresu. W wyniku otrzymując wykres z podwójną osią Y, którego dane mogą mieć różną skalę (np. Rysunek 4.22).

### 3.2.4 Pomiar czasu

Dla porównania przebiegu procesu odtwarzania wprowadzono pomiar wykonywania dla każdego kroku "pipeline". Po czym zmierzony czas się sumuje i zwraca się wraz z wynikiem odtwarzania sceny funkcją "pipeline".

Proces uruchomienia funkcji "pipeline" jest zautomatyzowany dla każdego elementu macierzy utworzonej wektorami:

- Danych wejściowych,
- Parametru transformaty **SIFT**,
- Próby(aby uśrednić pomiar czasu).

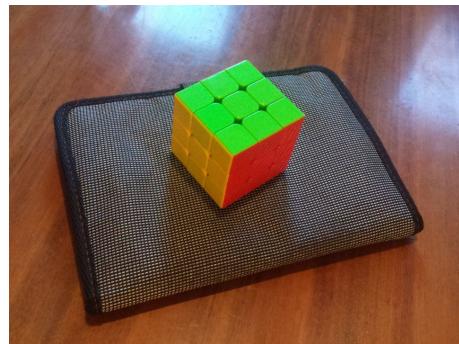


# Rozdział 4

## Przebieg badań

W badaniu przeprowadzono analizę wyników przetwarzania zestawów zdjęć, każdy zestaw przedstawia następujące sceny:

- Kostka Rubika na podstawce, "c0" (Rysunek 4.1).
- Kostka Rubika z oświetleniem zmiennym, "c1" (Rysunek 4.2).
- Kostka Rubika z oświetleniem punktowym, "c2" (Rysunek 4.3).
- Znaki specjalne, "mark" (Rysunek 4.4).
- Budynek A1 PWr (wybrzeże Wyspiańskiego 27), "A1" (Rysunek 4.5).



Rysunek 4.1 Kostka Rubika na podstawce, scena "c0", 4 zdjęcia

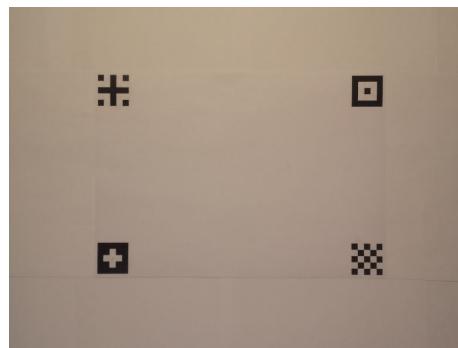


Rysunek 4.2 Kostka Rubika z oświetleniem zmiennym, scena "c1", 20 zdjęć

Wyniki są wygenerowane w następujących postaciach:



Rysunek 4.3 Kostka Rubika z oświetleniem punktowym, scena "c2", 10 zdjęć



Rysunek 4.4 Znaki specjalne, scena "mark", 13 zdjęć

- Rekonstrukcja sceny w przestrzeni trójwymiarowej, zawiera zbiór punktów kluczowych i wykrytych pozycji kamery wraz. Nazwa pozycji odpowiada nazwie pliku zdjęcia.
- Oznaczone wspólne punkty kluczowe dla wybranych zdjęć. Na rekonstrukcji sceny punkty są oznaczone gwiazdką i promieniem ciągnącym się od miejsca kamery wraz z przydzieloną literą. Na zdjęciu punkty są oznaczone fioletowym kółkiem i odpowiednią literą.
- Wykres zależności czasu wykonywania, procentu wykrytych ujęć i powodzenia na etapie rekonstrukcji sceny od ustawień parametru transformacji SIFT.
  - Na czerwono: czas w sekundach
  - Na niebiesko: wykryte ujęcia w procentach
  - Na zielono: procent udanych prób rekonstrukcji sceny, może przyjmować wartość 0 (próba nieudana) lub 100 (rekonstrukcja powiodła się), wartość pomiędzy 0 a 100 oznaczała błąd podczas wykonywania pomiarów.

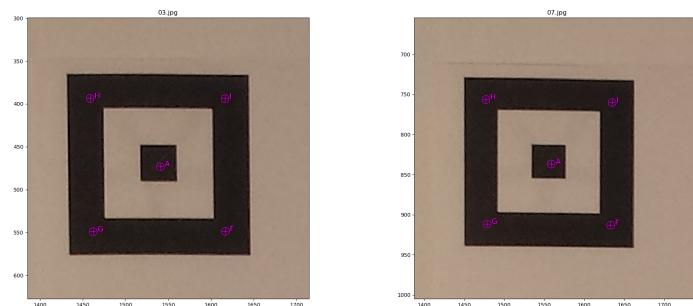
## 4.1 Odtwarzanie sceny z różną zawartością. Wykrycie i dopasowanie cech

Dla dwóch zestawów "A1" i "mark" przeprowadzono odtwarzanie sceny przy takich samych parametrach.

Scena "mark" przedstawia sobą cztery czarne znaki charakterystyczne na białym tle. Scena była specjalnie zaprojektowana, aby sprawdzić zachowanie algorytmu **SIFT**.



Rysunek 4.5 Budynek A1 PWr, scena "A1", 19 zdjęć



Rysunek 4.6 Wybrane punkty charakterystyczne, zdjęcie 03 i 07, scena "mark"

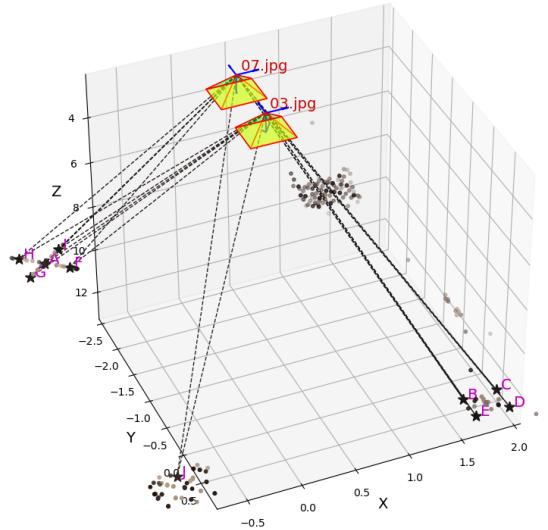
Punkty charakterystyczne na rysunku 4.6 są umieszczone na wierzchołkach znaku, co zgadza się z algorytmem, gdyż w tych miejscach występują ekstrema funkcji **DoG**. Brak punktów na krawędziach spowodowany jest jednym z kroków filtracji **SIFT** punktów na krawędziach.

Charakterystyczną różnicą odtwarzanych scen "A1" (Rysunek 4.8) i "mark" (Rysunek 4.7) pod względem wykrytych punktów kluczowych jest znacznie mniejsza ilość odnalezionych cech dla sceny "mark". Taka różnica jest skutkiem ograniczonej ilości elementów, które mogłyby spełnić kryteria transformaty **SIFT**.

Dodatkowo można odznać bardzo dobry wynik jak zestawienia punktów kluczowych dla sceny "A1" (Rysunek 4.9 - 4.11) przy tak dużej liczbie elementów kluczowych, jak i dokładność wyznaczonych kształtów sceny (Rysunek 4.7, 4.15 - 4.17) — zdjęcia sceny "mark" rzeczywiście były robione przemieszczaniem kamery wzduż prostej.

## 4.2 Wpływ parametrów rozpoznawania cech na wynik

Pierwszy krok procesu fotogrametrycznego - rozpoznawanie cech można dokonać z różnym parametrem ustawień transformaty SIFT. Taki parametr, nazywany "preset" może przyjąć jedną z wartości ["low", "medium", "normal", "high", "ultra"]. Rysunki 4.22 - 4.26 posiadają wyniki wykonywań procesu fotogrametrycznego dla różnych ustawień. Dla przeciętnej sceny stosowanie ustawień "low" i "medium" zwykle było niewystarczające aby wykryć jakiekolwiek pozycje kamery. Ustawienie "normal" okazało się wystarczającym aby wykryć 80% pozycji. Ustawienia "high" i "ultra" pozwalają uzyskać najlepszy



Rysunek 4.7 Wybrane punkty charakterystyczne, scena "mark"

wynik, jednak stosowanie takich ustawień skutkuje znacznym wydłużeniem czasu obliczenia.

Dogłębne wyszukiwanie cech może wykryć więcej punktów kluczowych w scenach z wysoką entropią, np. dla sceny "A1", "c2", rekonstruowane sceny 3D (Rysunek 4.12 - 4.14, 4.18 - 4.21) których posiadają charakterystyczny wzrost wykrytych cech wraz z wzrostem parametrem "preset".

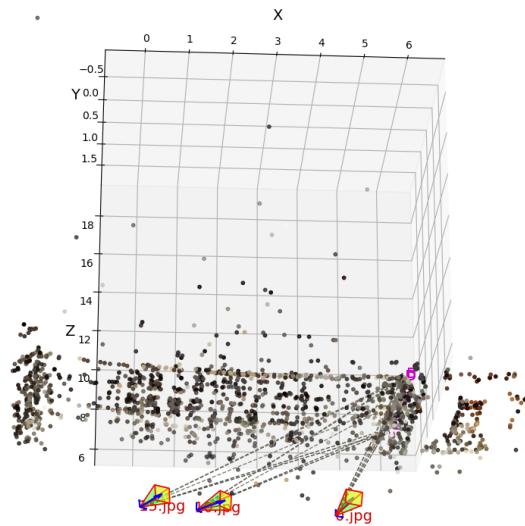
Swoją drogą dla sceny "mark" — sceny o niskiej entropii zbiór cech pozostaje praktycznie niezmienny (Rysunek 4.15 - 4.17)

Skutkiem ubocznym stosowania wyższych ustawień można uważać zwiększenie szumu (błędnie wykrytych i źle zlokalizowanych cech wspólnych) w scenach z dużą ilością obiektów (Rysunek 4.19, 4.20, 4.21).

Zestaw "c1" posiada zdjęcia przy różnym oświetleniu — część zdjęć była zrobiona z dodatkowym źródłem światła. Można zauważyć, że rekonstrukcja sceny powiodła się dopiero przy większej liczbie cech, przy większym ustawieniu "high" (w porównaniu do innych zestawów odtwarzanie sceny następowało przy mniejszym nastawieniu), przy czym nawet przy ustawieniu "ultra", część pozycji wciąż nie jest odtwarzana. Jednym z warunków dobrego wyniku fotogrametrycznego jest niezmiennosć sceny w trakcie robienia zdjęć, w tym oświetlenia.

### 4.3 Czas obliczenia

Rysunki 4.22 - 4.26 przedstawiają zależności czasu od ustawień transformaty **SIFT**. Zmiana parametrów transformaty wykrywającej cechy ma bezpośredni wpływ na czas potrzebny do wykrycia cech i pośredni dla parowania cech i odtwarzania sceny, gdyż dokładniejsze wyszukiwanie cech zwiększa liczbę znalezionych punktów kluczowych. Wyjatek stanowią sceny z ograniczoną liczbą elementów. Na przykład dla sceny "mark" liczba



Rysunek 4.8 Wybrane punkty charakterystyczne, scena "A1"

wykrytych elementów przy ustawieniach "normal" i "high" nie zmienia się. Wzrost czasu przy ustawieniu "ultra" jest powiązany wzrostem szumu (błędnie wykrytych punktów charakterystycznych).

Bezpośredni wpływ na czas obliczenia również stanowi liczba zdjęć. Czas potrzebny na przetworzenie zestawu zdjęć wzrasta wraz z liczbą ujęć.



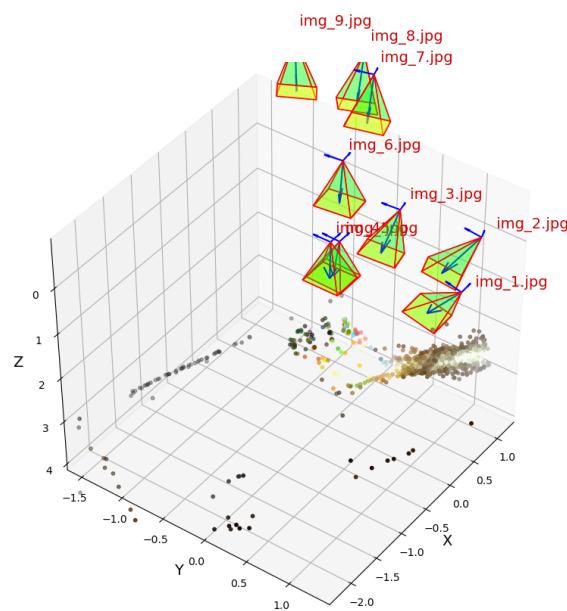
Rysunek 4.9 Wybrane punkty charakterystyczne, zdjęcie 0, scena "A1"



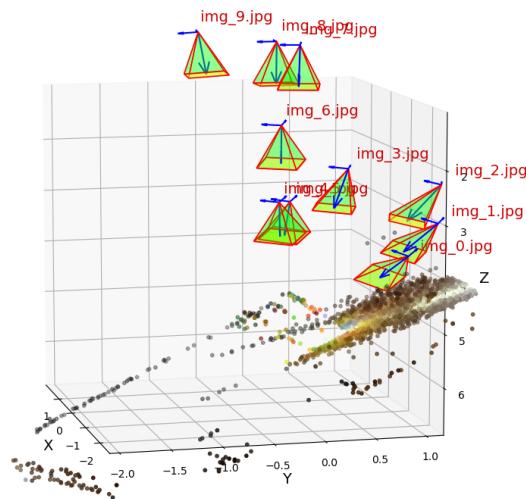
Rysunek 4.10 Wybrane punkty charakterystyczne, zdjęcie 10, scena "A1"



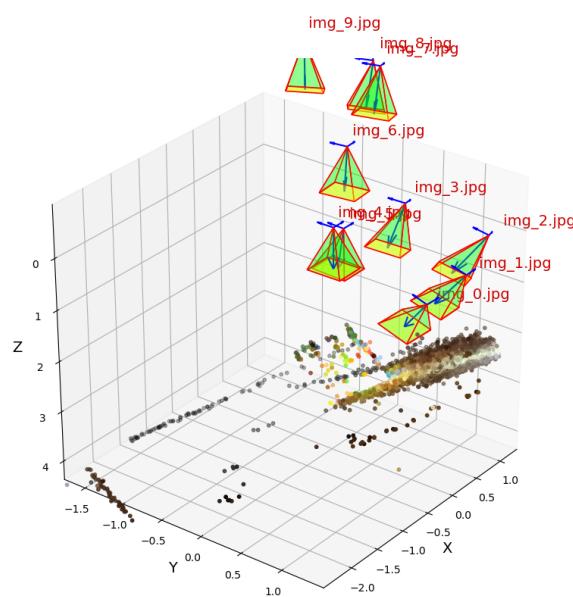
Rysunek 4.11 Wybrane punkty charakterystyczne, zdjęcie 15, scena "A1"



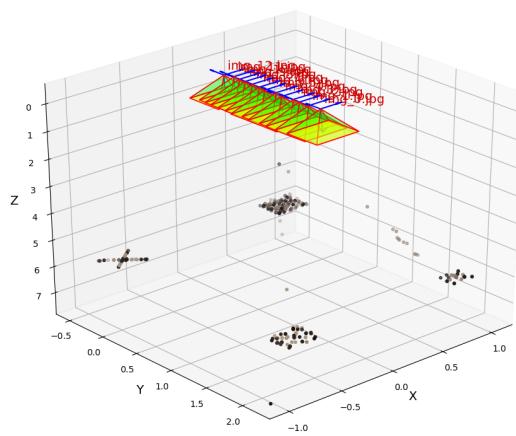
Rysunek 4.12 Odtwarzana scena "c2", parametr: normal



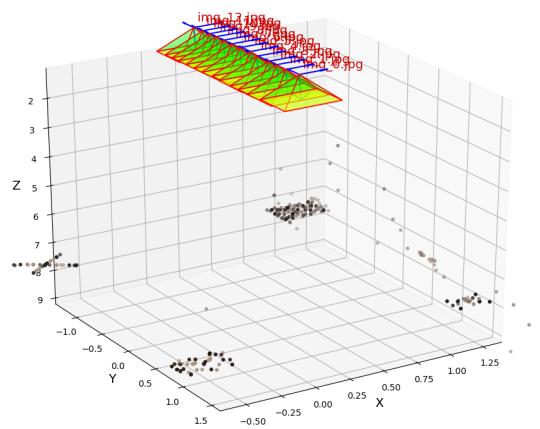
Rysunek 4.13 Odtwarzana scena "c2", parametr: high



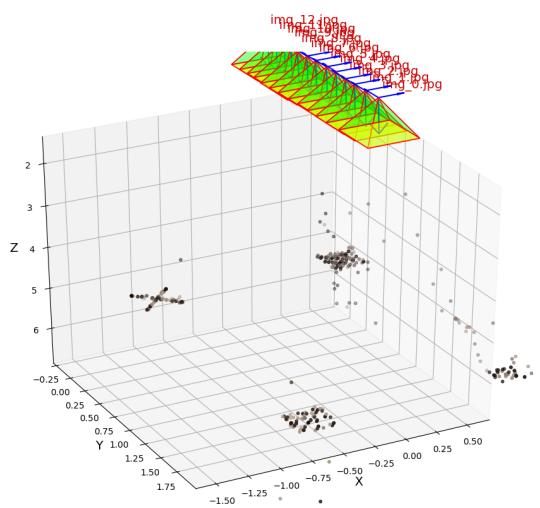
Rysunek 4.14 Odtwarzana scena "c2", parametr: ultra



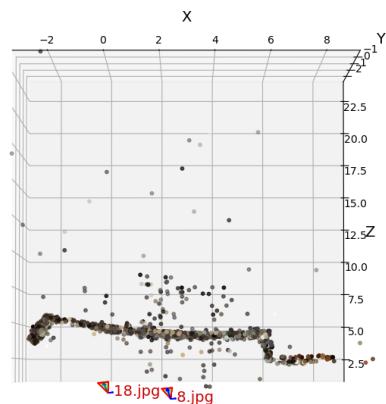
Rysunek 4.15 Odtwarzana scena "mark", parametr: normal



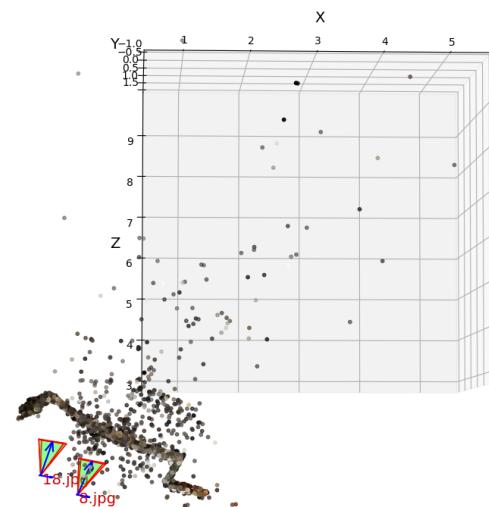
Rysunek 4.16 Odtwarzana scena "mark", parametr: high



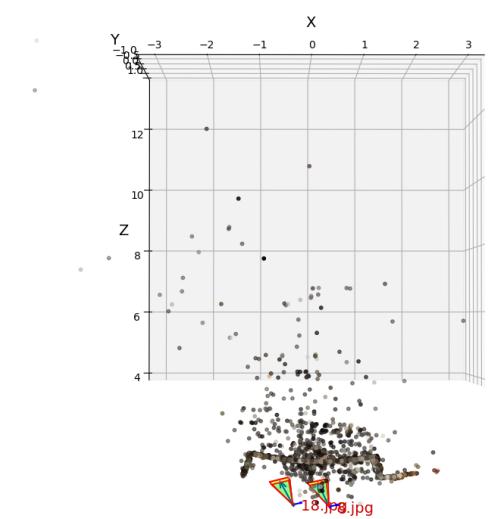
Rysunek 4.17 Odtwarzana scena "mark", parametr: ultra



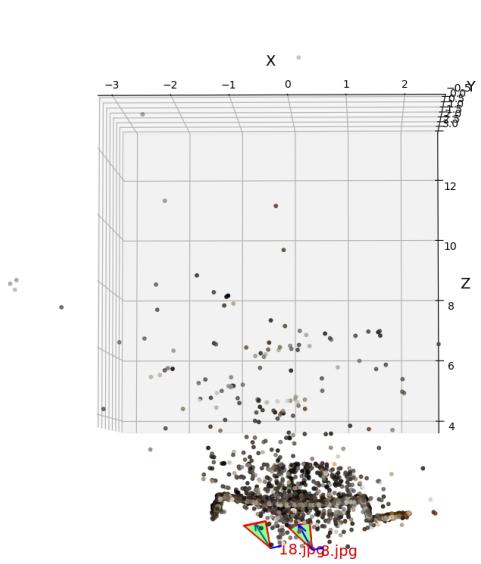
Rysunek 4.18 Odtwarzana scena "A1", parametr: medium



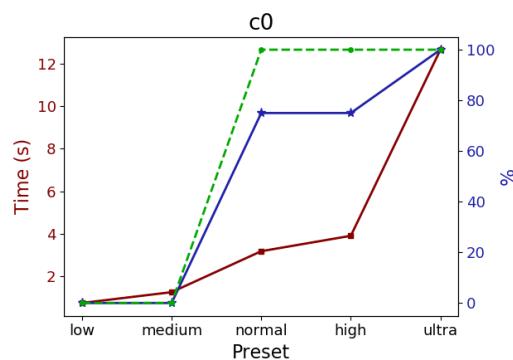
Rysunek 4.19 Odtwarzana scena "A1", parametr: normal



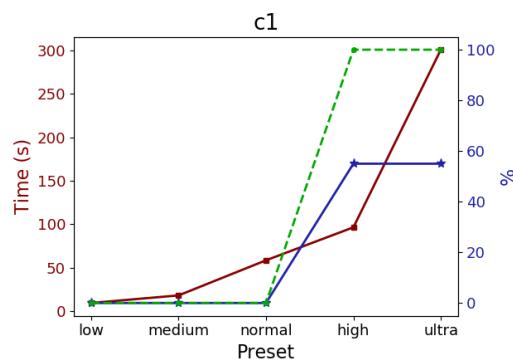
Rysunek 4.20 Odtwarzana scena "A1", parametr: high



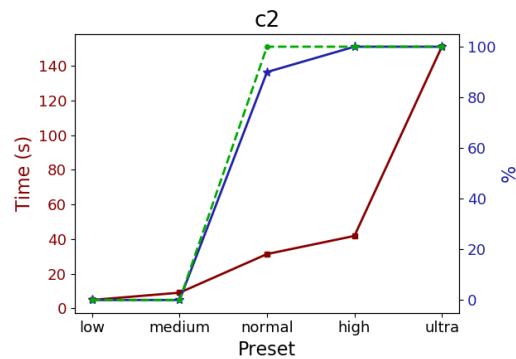
Rysunek 4.21 Odtwarzana scena "A1", parametr: ultra



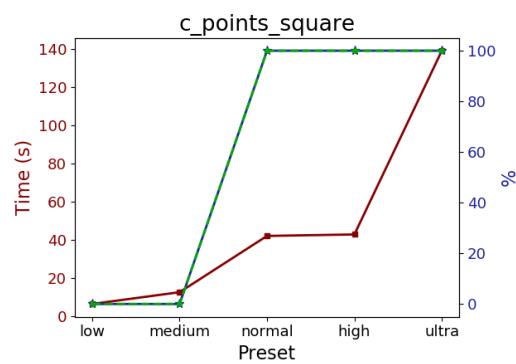
Rysunek 4.22 Czas wykonania, scena "c0"



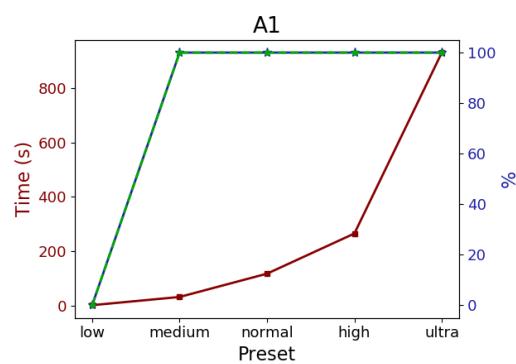
Rysunek 4.23 Czas wykonania, scena "c1"



Rysunek 4.24 Czas wykonania, scena "c2"



Rysunek 4.25 Czas wykonania, scena "mark"



Rysunek 4.26 Czas wykonania, scena "A1"



# Rozdział 5

## Podsumowanie

W trakcie wykonywania pracy inżynierskiej udało się zbudować zestaw narzędzi (śródowisko), które umożliwia uruchomienie procesu lokalizacji fotogrametrycznej, wyświetlenie wyników lokalizacji, odtwarzanie sceny w przestrzeni trójwymiarowej. Dodatkowo środowisko posiada narzędzie dla przeprowadzania testów wydajności dla różnych zestawów danych i różnych parametrów przetwarzania.

Przedstawione środowisko wykorzystuje platformę AliceVision dla implementacji procesu fotogrametrycznego. Platforma ta została wybrana dla potrzeb projektu, ponieważ jest dobrze opracowanym i potężnym frameworkiem z ciągle podtrzymywany i otwartym kodem źródłowym. Do wad frameworku można odnieść częściowy brak dokumentacji i dość skomplikowany proces komplikacji frameworku w środowisku własnym.

Dla praktycznego sprawdzenia zaproponowanej teorii lokalizacji zostały dobrane pięć zestawów zdjęć zawierające różne sceny. Sceny czterech z pięciu zestawów zawierały zestaw obiektów ( kostka Rubika, arkusz zawierający znaki specjalne ), jedna scena zawierała otoczenie (duży budynek).

Można powiedzieć, że proces przetwarzania zdjęć jest dość wymagający pod względem zapotrzebowania zasobów obliczeniowych. Obliczenia były przeprowadzane na przeciętnym sprzęcie obliczeniowym dla roku 2020 — procesor Intel i7-2620m.

Dla zestawu o rozmiarze 10 zdjęć potrzebne jest około 40 sekund na uzyskanie zadowalającego wyniku. Dla zestawu o rozmiarze 4 zdjęć uzyskanie zadowalającego wyniku zajęło 4 sekundy.

Ponieważ czas obliczenia wzrasta wraz ze wzrostem nastawień parametrów przetwarzania i wraz ze wzrostem ilości danych, dla dużych zestawów danych proponowana jest próba stosowania niższych parametrów na początek, a następnie stopniowe zwiększenie ustawień aż do momentu uzyskania zadowalających wyników. Dodatkowo można wprowadzić filtrację wstępną obrazów złej jakości i zbędnej zawartości.

Uzyskanie dobrego wyniku lokalizacji (stosunek wykrytych pozycji do liczby zdjęć) wprost zależy od liczby cech charakterystycznych na zdjęciach. Przykładem dobrego wyniku można uważać przetwarzanie zestawu "A1", zdjęcia którego mają wysoki poziom entropii.

Dla 4 z 5 zestawów proces lokalizacji zakończył się powodzeniem, pozostały zestaw posiadał zdjęcia przy różnym oświetleniu i część ujęć nie została wykryta. Aby uzyskać najlepszy wynik scena fotogrametryczna powinna być niezmienna w trakcie robienia zdjęć.

Zastosowany algorytm pozwala na względную lokalizację, oznacza to, że rzeczywiste położenie i odległości pozostają nieznane. Aby móc na podstawie opisanego algorytmu uzyskać lokalizację absolutną należy wprowadzić skalę (na przykład przez zmierzenie jednego z obiektów sceny) i wprowadzić układ odniesienia do sceny.



# Literatura

- [1] AliceVision. Photogrammetry Pipeline  
<https://alicevision.org/#photogrammetry>
- [2] AliceVision. Source code  
<https://github.com/alicevision/AliceVision>
- [3] AliceVision. Node Reference  
<https://github.com/alicevision/meshroom-manual>
- [4] Ives Rey-Otero<sup>1</sup>, Mauricio Delbracio. *Anatomy of the SIFT Method*  
<http://www.ipol.im/pub/art/2014/82/>
- [5] David G. Lowe. *Distinctive Image Features from Scale-Invariant Keypoints*  
<https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf>
- [6] Pierre Moulon<sup>1,2</sup>, Pascal Monasse<sup>1</sup>, Renaud Marlet *Adaptive Structure from Motion with a contrariomodel estimation*  
<http://imagine.enpc.fr/~marletr/publi/ACCV-2012-Moulon-et-al.pdf>
- [7] John Hunter, Darren Dale, Eric Firing, Michael Droettboom. *Matplotlib. User's Guide*  
<https://matplotlib.org/Matplotlib.pdf>
- [8] *argparse — Parser for command-line options, arguments and sub-commands*  
<https://docs.python.org/3/library/argparse.html>
- [9] *json — JSON encoder and decoder*  
<https://docs.python.org/3/library/json.html>