

POLITECHNIKA BIAŁOSTOCKA

WYDZIAŁ INFORMATYKI

PRACA DYPLOMOWA INŻYNIERSKA

TEMAT: EDYTOR MODELI 3D OPARTYCH  
O WOKSELE

WYKONAWCA: PAWEŁ ALEKSIEJUK

.....  
podpis

PROMOTOR: DR INŻ. ŁUKASZ GADOMER

.....  
podpis

BIAŁYSTOK 2022 r.



## Karta dyplomowa

|  |  |   |
|--|--|---|
| Politechnika Białostocka<br><br>Wydział Informatyki<br><br><br>Katedra Nazwa katedry   | Studia stacjonarne<br><br>studia I stopnia | Numer albumu studenta:<br>105527  |
|  |  | Rok akademicki 2021/2022  |
|  |  | Kierunek studiów:<br>Informatyka<br>Specjalność: Inżynieria<br>Oprogramowania |
| <b>Paweł Aleksiejuk</b><br><br><b>TEMAT PRACY DYPLOMOWEJ:</b><br><b>Edytor modeli 3D opartych o woksele</b><br><br>Zakres pracy: <ol style="list-style-type: none"> <li>1. Przegląd podobnych rozwiązań dostępnych na rynku.</li> <li>2. Zdefiniowanie wymagań stawianych wobec rozwiązania.</li> <li>3. Opracowanie prostego silnika 3D.</li> <li>4. Stworzenie narzędzia do edycji modelu 3D</li> <li>5. Testowanie stworzonego rozwiązania.</li> </ol> <b>Słowa kluczowe (max 5):</b> Woksel, 3D, Edytor Modeli, C++  |  |   |
| <div style="display: flex; justify-content: space-between;"> <div style="width: 45%; text-align: center;">                 .....<br/>                 Imię i nazwisko promotora - podpis             </div> <div style="width: 45%; text-align: center;">                 .....<br/>                 Imię i nazwisko kierownika katedry - podpis             </div> </div>   |  |   |
| <div style="display: flex; justify-content: space-between;"> <div style="width: 30%; text-align: center;">                 .....<br/>                 Data wydania tematu pracy dyplomowej<br/>                 - podpis promotora             </div> <div style="width: 30%; text-align: center;">                 .....<br/>                 Regulaminowy termin złożenia<br/>                 pracy dyplomowej             </div> <div style="width: 30%; text-align: center;">                 .....<br/>                 Data złożenia pracy dyplomowej<br/>                 - potwierdzenie dziekanatu             </div> </div> |  |   |
| <div style="display: flex; justify-content: space-between;"> <div style="width: 45%; text-align: center;">                 .....<br/>                 Ocena promotora             </div> <div style="width: 45%; text-align: center;">                 .....<br/>                 Podpis promotora             </div> </div>   |  |   |
| <div style="display: flex; justify-content: space-between;"> <div style="width: 30%; text-align: center;">                 .....<br/>                 Imię i nazwisko recenzenta             </div> <div style="width: 30%; text-align: center;">                 .....<br/>                 Ocena recenzenta             </div> <div style="width: 30%; text-align: center;">                 .....<br/>                 Podpis recenzenta             </div> </div>  |  |   |



Subject of diploma thesis

3D model editor based on voxels.

## **Summary**

Streszczenie pracy po angielsku.



Załącznik nr 4 do „Zasad postępowania przy przygotowaniu i obronie  
pracy dyplomowej na PB”  
Białystok, dnia 05.01.2022 r.

**Paweł Aleksiejuk**

Imiona i nazwisko studenta

**105527**

Nr albumu

**informatyka, stacjonarne**

Kierunek i forma studiów

**dr inż. Łukasz Gadomer**

Promotor pracy dyplomowej

### OŚWIADCZENIE

Przedkładając w roku akademickim 2021/2022 Promotorowi **dr inż. Łukasz Gadomer** pracę dyplomową pt.: **Edytor modeli 3D opartych o woksele**, dalej zwaną pracą dyplomową, **oświadczam, że:**

- 1) praca dyplomowa stanowi wynik samodzielnej pracy twórczej;
- 2) wykorzystując w pracy dyplomowej materiały źródłowe, w tym w szczególności: monografie, artykuły naukowe, zestawienia zawierające wyniki badań (opublikowane, jak i nieopublikowane), materiały ze stron internetowych, w przypisach wskazywałem/am ich autora, tytuł, miejsce i rok publikacji oraz stronę, z której pochodzą powoływane fragmenty, ponadto w pracy dyplomowej zamieściłem/am bibliografię;
- 3) praca dyplomowa nie zawiera żadnych danych, informacji i materiałów, których publikacja nie jest prawnie dozwolona;
- 4) praca dyplomowa dotychczas nie stanowiła podstawy nadania tytułu zawodowego, stopnia naukowego, tytułu naukowego oraz uzyskania innych kwalifikacji;
- 5) treść pracy dyplomowej przekazanej do dziekanatu Wydziału Informatyki jest jednakowa w wersji drukowanej oraz w formie elektronicznej;
- 6) jestem świadomy/a, że naruszenie praw autorskich podlega odpowiedzialności na podstawie przepisów ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (Dz. U. z 2019 r. poz. 1231, późn. zm.), jednocześnie na podstawie przepisów ustawy z dnia 20 lipca 2018 roku Prawo o szkolnictwie wyższym i nauce (Dz. U. poz. 1668, z późn. zm.) stanowi przesłankę wszczęcia postępowania dyscyplinarnego oraz stwierdzenia nieważności postępowania w sprawie nadania tytułu zawodowego;
- 7) udzielam Politechnice Białostockiej nieodpłatnej, nieograniczonej terytorialnie i czasowo licencji wyłącznej na umieszczenie i przechowywanie elektronicznej wersji pracy dyplomowej w zbiorach systemu Archiwum Prac Dyplomowych Politechniki Białostockiej oraz jej zwielokrotniania i udostępniania w formie elektronicznej w zakresie koniecznym do weryfikacji autorstwa tej pracy i ochrony przed przywłaszczeniem jej autorstwa.

..... czytelny podpis studenta





# Spis treści

|   |           |
|---|-----------|
| <b>Streszczenie</b>                             | <b>5</b>  |
| <b>Wstęp</b>                                    | <b>11</b> |
| <b>1 Przegląd istniejących rozwiązań</b>        | <b>13</b> |
| 1.1 MagicaVoxel . . . . .                       | 13        |
| 1.2 Mega Voxels Play . . . . .                  | 14        |
| 1.3 Qubicle . . . . .                           | 15        |
| 1.4 Goxel . . . . .                             | 16        |
| 1.5 VoxEdit Beta . . . . .                      | 17        |
| <b>2 Projekt systemu</b>                        | <b>19</b> |
| 2.1 Wymagania . . . . .                         | 19        |
| 2.2 Diagramy stanów . . . . .                   | 20        |
| 2.3 Diagram przypadków użycia i opisy . . . . . | 21        |
| <b>3 Zastosowane technologie i rozwiązania</b>  | <b>27</b> |
| 3.1 Języki programowania . . . . .              | 27        |
| 3.2 Środowisko programistyczne . . . . .        | 27        |
| 3.3 Biblioteki . . . . .                        | 28        |
| <b>4 Implementacja</b>                          | <b>29</b> |
| <b>5 Prezentacja aplikacji</b>                  | <b>31</b> |
| <b>6 Rozdział 6</b>                             | <b>35</b> |
| <b>Podsumowanie</b>                             | <b>37</b> |
| <b>Bibliografia</b>                             | <b>40</b> |

|                        |           |
|------------------------|-----------|
| <b>Spis tabel</b>      | <b>41</b> |
| <b>Spis rysunków</b>   | <b>43</b> |
| <b>Spis listingów</b>  | <b>45</b> |
| <b>Spis algorytmów</b> | <b>47</b> |

# Wstęp

Grafika we współczesnym użyciu kojarzy się głównie z komputerowym przedstawieniem danych. Dane te tworzą medium wizualne, które mogą być wyświetlane między innymi na ekranach naszych monitorów komputerowych. Najczęściej rozróżniamy dwie rodzaje grafik: 2D (ang. *two-dimensional*) i 3D (ang. *three-dimensional*). Grafika 2D polega na przedstawieniu medium wizualnego opartego na obiekcie o dwóch wymiarach, zaś grafika 3D, analogicznie na obiekcie o trzech wymiarach.

Wokselem (ang. *voxel*) [24] nazywamy przedstawienie punktu w trójwymiarze. Nazwa woksół jest połączeniem angielskich słów *volume* oraz *element* i jest to analogiczne połączenie do *picture* i *element* w przypadku piksela (ang. *pixel*). Najczęstszym sposobem

Zainteresowany tymi tematami, autor postanowił prześledzić drogę tworzenia grafiki 3D od strony edytora modeli, tworząc go od podstaw w ramach tej pracy. Edytor ten ma pozwolić użytkownikowi na kreację modelu 3D opartego na woksłach, wykorzystując wbudowane mechanizmy edycji.

Motywacją do napisania tej pracy było chęć stworzenia prostego funkcjonalnego silnika graficznego wraz z narzędziem do tworzenia modeli obsługiwanych przez ten silnik. W późniejszym czasie, planuję rozszerzyć ten projekt, tworząc w pełni funkcjonalną grę 3D.

Zakres pracy obejmował:

- Przegląd podobnych rozwiązań dostępnych na rynku.
- Zdefiniowanie wymagań stawianych wobec rozwiązania.
- Opracowanie prostego silnika 3D.
- Stworzenie narzędzia do edycji modelu 3D.
- Testowanie stworzonego rozwiązania.

Rozdział 1 przedstawia 5 istniejących już na rynku edytorów graficznych opartych o woksół, w celu zaznajomienia się z podstawowymi funkcjonalnościami postawionymi przez ich autorów.

Rozdział 2 skupia się na przedstawieniu dogłębnie projektu systemu, w celu zapoznania się z wymaganiami wobec aplikacji, jak i z wizją systemu w formie graficznej.

Rozdział 3 opisuje zastosowane technologie i rozwiązania, tłumacząc logikę i motywację za wyborem każdego z nich.

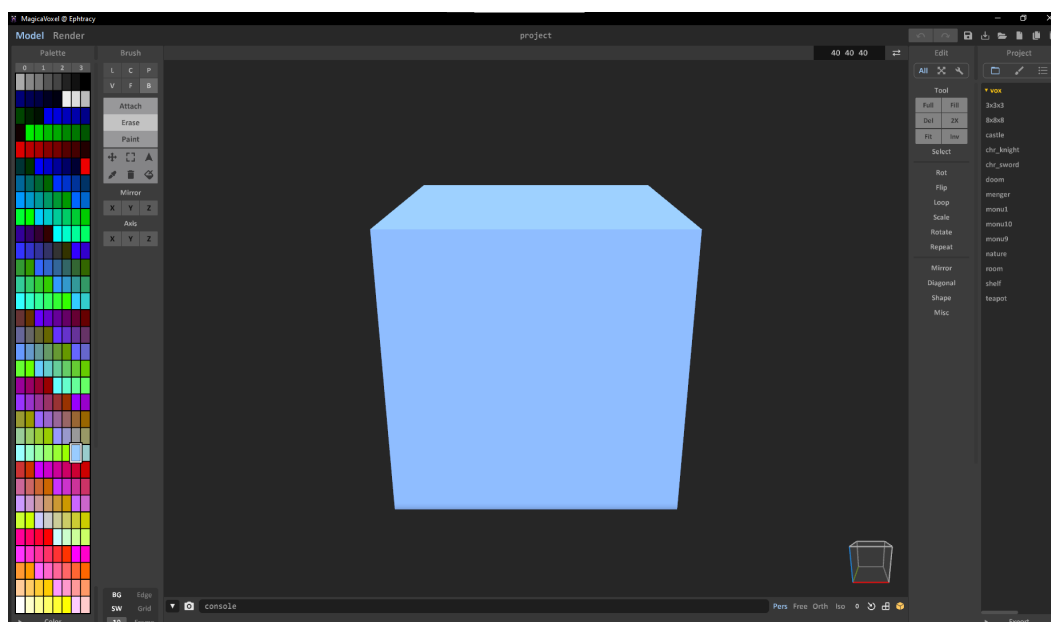
Rozdział 4 opisuje realizację aplikacji. Zawiera w sobie informacje na temat technologii, środowiska programistycznego i implementacji.

# 1. Przegląd istniejących rozwiązań

Z uwagi na specjalistyczne zastosowanie stworzonego edytora graficznego, a mianowicie tworzenie specjalnych obiektów obsługiwanych przez wbudowany silnik graficzny, istniejące rozwiązania w głównej mierze mają służyć jako wykaz podstawowych, jak i dodatkowych funkcjonalności do możliwej implementacji w ostatecznym rozwiązaniu.

## 1.1 MagicaVoxel

MagicaVoxel [7] jest najpopularniejszym darmowym desktopowym edytorem wokseli dostępnym aktualnie na rynku. Stworzony i na bieżąco aktualizowany przez użytkownika o pseudonimie @ephtracy pozwala na nie tylko tworzenie modeli, ale też zdjęć do późniejszego udostępniania. Taka funkcjonalność pozwala na przetestowanie modelu w różnych warunkach, które są edytowalne poprzez parametry w wewnętrznym silniku renderującym. Interfejs rozwiązania został przedstawiony na rysunku 1.1



Rysunek 1.1: Ekran startowy programu MagicaVoxel (Windows), źródło: [7]

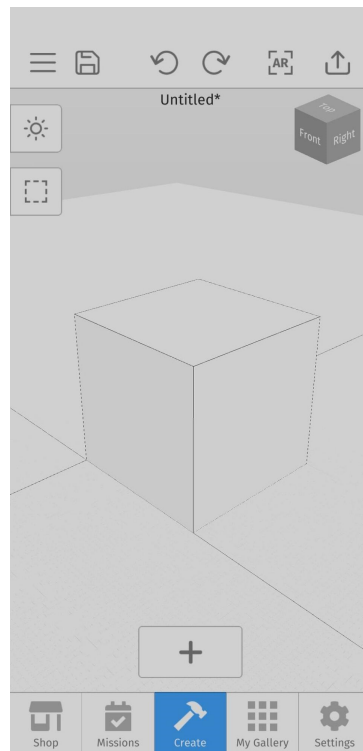
Główne atuty oprogramowania według producenta:

- Zaawansowany wewnętrzny silnik renderujący.
- Całkowicie darmowe oprogramowanie, nawet w przypadku użycia komercyjnego.

MagicaVoxel jest dostępny za darmo na platformach Windows i macOS.

## 1.2 Mega Voxels Play

Mega Voxels Play [10] to darmowy mobilny edytor stworzony przez Go Real Games. Tak jak większość edytorów wokselowych, pozwala na podstawowe operacje takie jak dodawanie, usuwanie i malowanie. Aplikacja posiada wbudowany sklep, który pozwala na pobranie gotowych modeli, w celu późniejszego wykorzystania. Interfejs rozwiązania został przedstawiony na rysunku 1.2



Rysunek 1.2: Ekran startowy programu Mega Voxels Play (Android), źródło: [10]

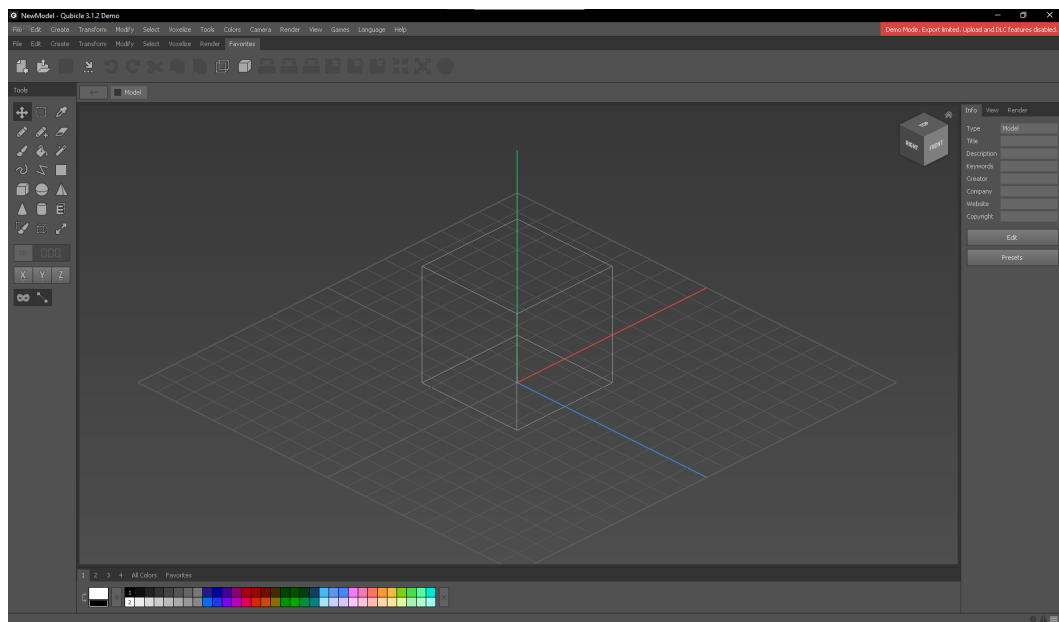
Główne atuty oprogramowania według producenta:

- Duża ilość bazowych modeli do pobrania.
- Prostość w obsłudze.
- Wsparcie dla AR (Rozszerzonej rzeczywistości).
- Różne efekty przetwarzania końcowego.

Mega Voxels Play jest dostępny za darmo na platformach mobilnych (Android i iOS).

### 1.3 Qubicle

Qubicle [9] jest zaawansowanym desktopowym narzędziem stworzonym przez Mind-desk, przeznaczonym do tworzenia wokselowych modeli. Z porównaniem do poprzedników, aplikacja nie posiada limitu wielkości modeli, co pozwala użytkownikom na swobodne tworzenie wielkich modeli, jak i całych terenów. Dodatkowo oprócz standardowego w edytorach formatu .obj (Wavefront File), wspierane są też takie formaty jak .fbx (Autodesk), .dae (Collada). Interfejs rozwiązania został przedstawiony na rysunku 1.3



Rysunek 1.3: Ekran startowy programu Qubicle (Windows, Steam), źródło: [9]

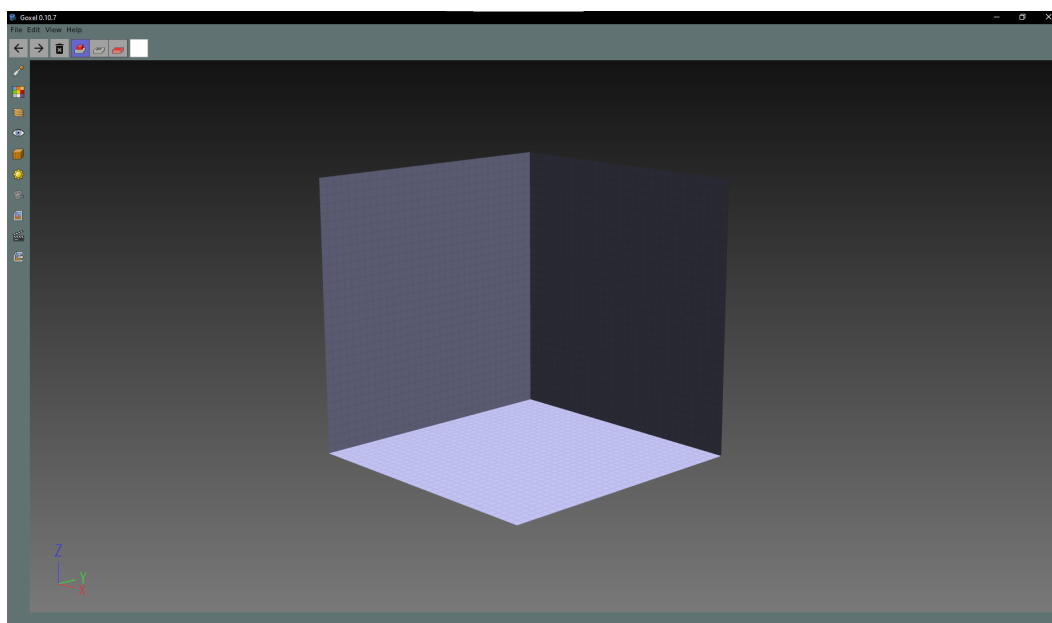
Główne atuty oprogramowania według producenta:

- Bardzo dużo narzędzi do edycji.
- Proste w obsłudze.
- Wbudowane narzędzie do konwersji z modelu siatkowego na model wokselowy.
- Wiele formatów do eksportu modeli.

Qubicle jest dostępny w czterech wersjach na platformach Windows i macOS, wersja okrojona (demo) za darmo, wersja podstawowa (bazowa) za 53.99 PLN, wersja rozszerzona (indie) za 89.99 PLN i pełna opcja (pro) za 410.56 PLN.

## 1.4 Goxel

Goxel [13] jest otwartym oprogramowaniem do edycji modeli wokselowych na komputery osobiste i urządzenia mobilne stworzone przez użytkownika o pseudonimie @guillaumechereau (GitHub). Główną funkcjonalnością Goxel, jest możliwość tworzenia warstw, w taki sam sposób jak w popularnych aplikacjach do manipulacji obrazami, między innymi takim jaki jest Adobe Photoshop. Interfejs rozwiązania został przedstawiony na rysunku 1.4



Rysunek 1.4: Ekran Startowy programu Goxel (Windows), źródło: [13]

Główne atuty oprogramowania według producenta:

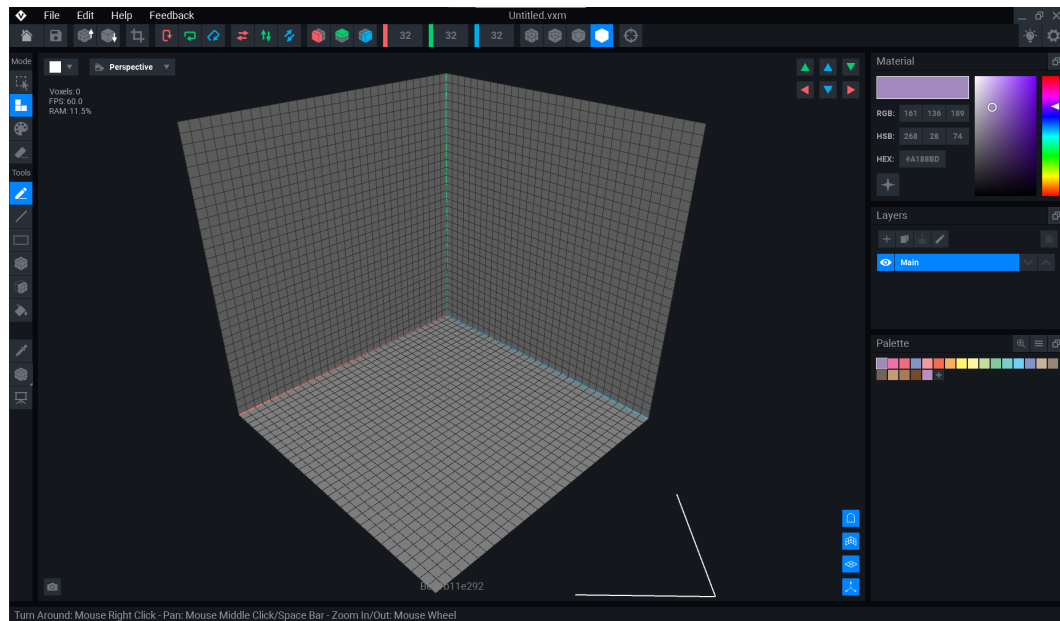
- Nieskończona wielkość sceny.
- Możliwość tworzenia obiektów na różnych warstwach.
- Wieloplatformowość.
- Wiele formatów do eksportu modeli.

Goxel jest dostępny za darmo na platformach Windows, Linux, iOS i macOS, a w przypadku platformy Android za opłatą 25.99 PLN.



## 1.5 VoxEdit Beta

VoxEdit Beta [20] jest darmowym oprogramowaniem stworzonym przez Pixowl do gry The Sandbox Game. Unikalną funkcjonalnością na tle innych aplikacji do edycji wokseli, jest możliwość montowania szkieletu i jego późniejszej animacji. Interfejs rozwiązania został przedstawiony na rysunku 1.5



Rysunek 1.5: Ekran startowy programu VoxEdit Beta (Windows), źródło: [20]

Główne atuty oprogramowania według producenta:

- Możliwość tworzenia animacji.
- Specjalny tryb edycji bloków.
- Przyjazny interfejs dla użytkownika.

VoxEdit Beta jest dostępny za darmo na platformach Windows i macOS.



## **2. Projekt systemu**

W tym rozdziale przedstawiono główne rozwiązania z zakresu inżynierii oprogramowania i projektowania, takie jak wymagania funkcjonalne i нефункциональные co do niniejszej pracy oraz diagramy, na których zobrazowano działanie systemu oraz zamieszczono ich opis. Tworząc taki projekt systemu, umożliwi on zaznajomienie się z podstawowym działaniem aplikacji.

### **2.1 Wymagania**

Na podstawie informacji zebranych z rozdziału 1 „Przegląd istniejących rozwiązań”, jak i wiedzy autora na temat programów graficznych, określone zostały podstawowe wymagania dotyczące aplikacji. Podzielono je na wymagania funkcjonalne oraz нефункциональные.

#### **2.1.1 Wymagania funkcjonalne**

Aplikacja stworzona w ramach niniejszej pracy powinna spełniać następujące funkcjonalności:

- Tworzenie modeli 3D.
- Prosty interfejs użytkownika.
- Zmiana pozycji i wielkości okienek.
- Edycja modeli w czasie rzeczywistym.
- Tworzenie własnych opisów materiałów
- Zapis i odczyt modelu.
- Zmiana właściwości oświetlenia.

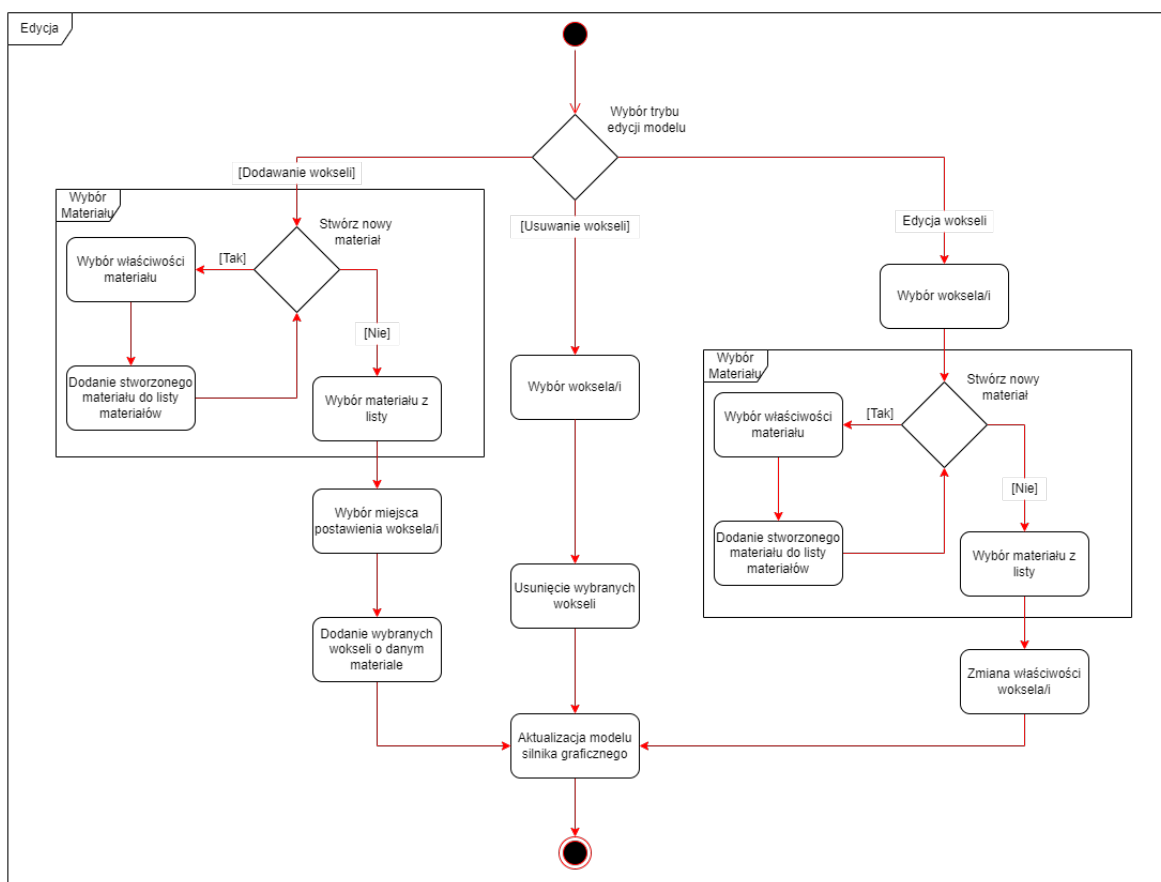
#### **2.1.2 Wymagania нефункциональные**

Prócz wymagań funkcjonalnych zdefiniowano szereg wymagań нефункциональных, które aplikacja powinna spełniać. Należą do nich:

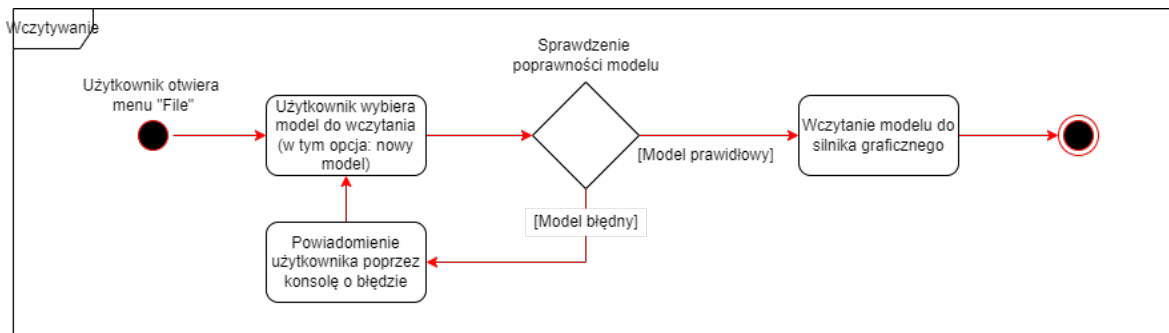
- Możliwość ponownego użycia silnika 3D w innych projektach.
- Wysoka responsywność na zmiany w modelu.
- Działanie na wielu platformach desktopowych (Windows, Linux, macOS).
- Konsola debugująca w czasie rzeczywistym.

## 2.2 Diagramy stanów

W aplikacji rozróżniamy dwa główne stany, stan gotowości do edycji albo stan wczytywania. Pierwszy z nich, przedstawiony na rysunku 2.1 jest odpowiedzialny za wprowadzanie zmian na obiekcie 3D. Z uwagi na to, że silnik renderuje klatki w czasie rzeczywistym, stan ten jest w ciągłej gotowości, czekający na ingerencje użytkownika. Drugi stan (rysunek 2.2) jest wywoływany w momencie wybrania nazwy pliku do wczytania.



Rysunek 2.1: Diagram stanów dla edycji, źródło: opracowanie własne

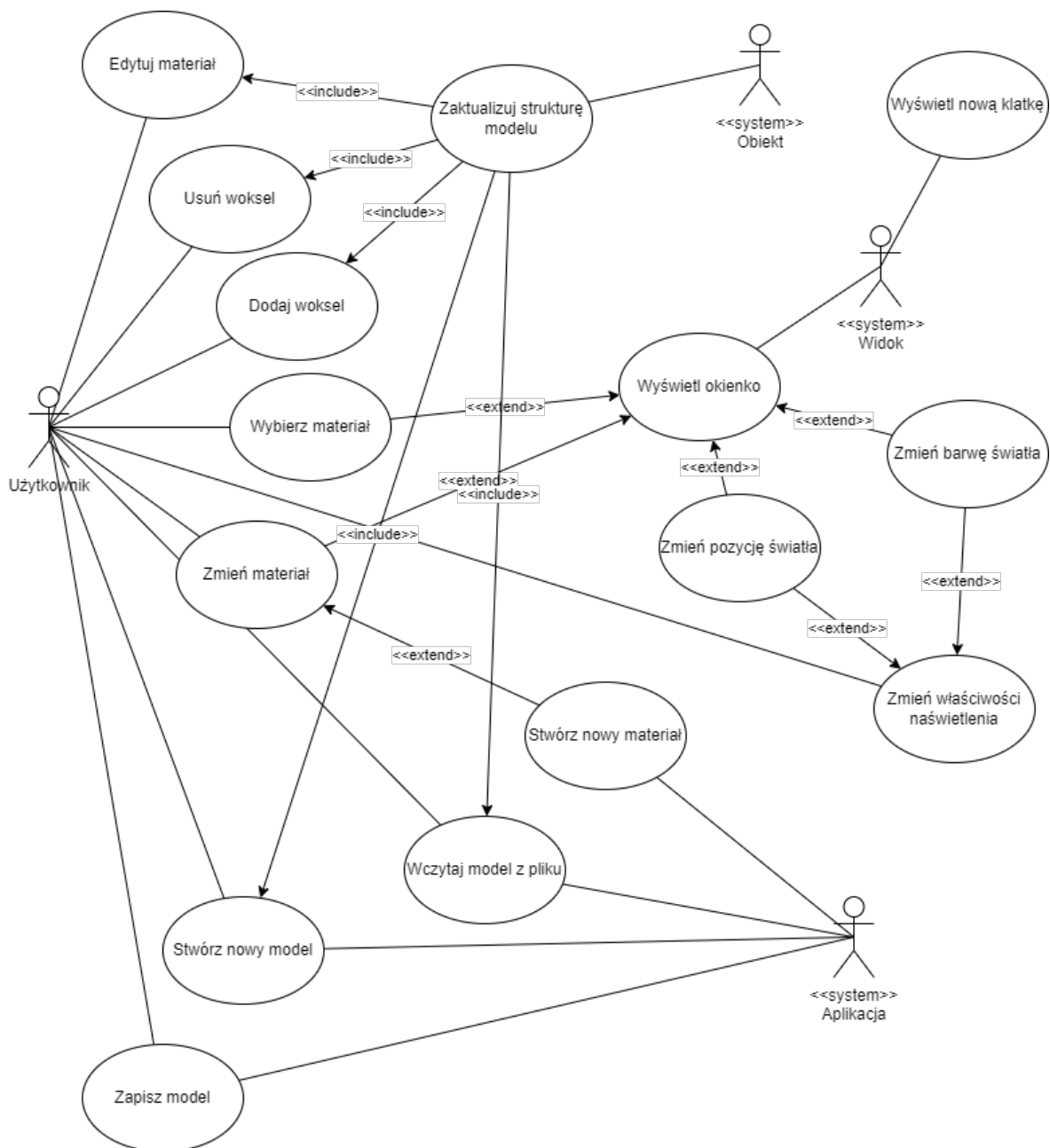


Rysunek 2.2: Diagram stanów dla wczytywania, źródło: opracowanie własne

## 2.3 Diagram przypadków użycia i opisy

Na rysunku 2.3 przedstawiono diagram przypadków użycia, w którym zawarte są wszystkie najważniejsze wymagania funkcjonalne w formie graficznej.

Głównymi przypadkami użycia tego systemu są metody manipulacji modelu 3D. To one pozwalają nam na dodawanie wokseli, usuwanie ich oraz edytowanie materiału. Przypadki te są odpowiednio opisane w tabeli 2.1, tabeli 2.2 oraz tabeli 2.3.



Rysunek 2.3: Diagram przypadków użycia, źródło: opracowanie własne

Tabela 2.1: Opis przypadku użycia „Dodaj wksel”

| Sekcja  | Treść   |
|---|---|
| <b>Uczestniczący aktorzy</b>  | Użytkownik, Widok, Aplikacja, Obiekt  |
| <b>Warunki wstępne</b>  | W okienku „Edit Mode” zaznaczony tryb „Add”   |
| <b>Warunki końcowe</b>  | Dodanie wksela do obiektu   |
| <b>Rezultat</b>   | Pojawienie się wksela w miejscu wskazanym przez użytkownika   |
| <b>Scenariusz główny</b>  | <ol style="list-style-type: none"> <li>1. Użytkownik wybiera materiał z listy, bądź dodaje swój własny i go zatwierdza.</li> <li>2. Użytkownik nacelowuje na interesującą go ściankę wksela, w celu postawienia na obok niej nowego wksela, po czym zatwierdza prawym przyciskiem myszy.</li> <li>3. Aplikacja przekazuje do obiektu dane kliknięcia.</li> <li>4. Obiekt aktualizuje strukturę danych.</li> <li>5. Widok zostaje odświeżony w następnej klatce.</li> <li>6. Użytkownik widzi efekt swojego działania na modelu 3D.</li> </ol> |
| <b>Scenariusz wyjątku</b>   | <p>Zdarzenie: Użytkownik nie kliknął na ściankę istniejącego wksela</p> <p>Wynik: Brak dodania wksela do modelu 3D</p>  |
| <b>Zależności czasowe</b>   | <ol style="list-style-type: none"> <li>1. Częstotliwość wykonania: 0 lub więcej na sesję.</li> <li>2. Typowy czas realizacji: 8,9 ms.</li> <li>3. Maksymalny czas realizacji: 33,2 ms.</li> </ol>   |
| <b>Wartości uzyskane przez aktorów po zakończeniu przypadków użycia</b> | <ol style="list-style-type: none"> <li>1. Pojawienie się wksela w miejscu i o materiale wybranym przez użytkownika.</li> <li>2. Obiekt posiada zaktualizowaną strukturę o wksela.</li> </ol>  |

Tabela 2.2: Opis przypadku użycia „Usuń woxsel”

| Sekcja  | Treść  |
|---|--|
| <b>Uczestniczący aktorzy</b>  | Użytkownik, Widok, Aplikacja, Obiekt   |
| <b>Warunki wstępne</b>  | W okienku „Edit Mode” zaznaczony tryb „Remove”   |
| <b>Warunki końcowe</b>  | Usunięcie wskazanego woksela z obiektu   |
| <b>Rezultat</b>   | Zniknięcie woksela w miejscu wskazanym przez użytkownika   |
| <b>Scenariusz główny</b>  | <ol style="list-style-type: none"> <li>1. Użytkownik nacelowuje na interesujący go woxsel, po czym zatwierdza prawym przyciskiem myszy.</li> <li>2. Aplikacja przekazuje do obiektu dane kliknięcia.</li> <li>3. Obiekt zwraca woxsel zainteresowania.</li> <li>4. Aplikacja usuwa zwrócony woxsel.</li> <li>5. Widok zostaje odświeżony w następnej klatce.</li> <li>6. Użytkownik widzi efekt swojego działania na modelu 3D.</li> </ol> |
| <b>Scenariusz wyjątku</b>   | <p>Zdarzenie: Użytkownik nie kliknął w istniejącego woksela</p> <p>Wynik: Brak usunięcia woksela z modelu 3D</p>   |
| <b>Zależności czasowe</b>   | <ol style="list-style-type: none"> <li>1. Częstotliwość wykonania: 0 lub więcej na sesję.</li> <li>2. Typowy czas realizacji: 5.9 ms.</li> <li>3. Maksymalny czas realizacji: 16,6 ms.</li> </ol>  |
| <b>Wartości uzyskane przez aktorów po zakończeniu przypadków użycia</b> | <ol style="list-style-type: none"> <li>1. Zniknięcie woksela w miejscu wybranym przez użytkownika.</li> <li>2. Obiekt posiada zaktualizowaną strukturę bez klikniętego woksela.</li> </ol>   |



Tabela 2.3: Opis przypadku użycia „Edytuj materiał”

| Sekcja  | Treść   |
|---|---|
| <b>Uczestniczący aktorzy</b>  | Użytkownik, Widok, Aplikacja, Obiekt  |
| <b>Warunki wstępne</b>  | W okienku „Edit Mode” zaznaczony tryb „Color”   |
| <b>Warunki końcowe</b>  | Zmiana materiału we wskazanym miejscu w obiekcie  |
| <b>Rezultat</b>   | Zmiana materiału woksela w miejscu wskazanym przez użytkownika  |
| <b>Scenariusz główny</b>  | <ol style="list-style-type: none"> <li>1. Użytkownik nacelowuje na interesujący go wksel, po czym zatwierdza prawym przyciskiem myszy.</li> <li>2. Aplikacja przekazuje do obiektu dane kliknięcia.</li> <li>3. Obiekt zwraca wksel zainteresowania.</li> <li>4. Aplikacja zmienia kolor zwróconego woksela na ostatni wybrany materiał.</li> <li>5. Widok zostaje odświeżony w następnej klatce.</li> <li>6. Użytkownik widzi efekt swojego działania na modelu 3D.</li> </ol> |
| <b>Scenariusz wyjątku</b>   | <p>Zdarzenie: Użytkownik nie kliknął w istniejącego woksela</p> <p>Wynik: Brak zmiany koloru woksela w modelu 3D</p>  |
| <b>Zależności czasowe</b>   | <ol style="list-style-type: none"> <li>1. Częstotliwość wykonania: 0 lub więcej na sesję.</li> <li>2. Typowy czas realizacji: 3.2 ms.</li> <li>3. Maksymalny czas realizacji: 16,6 ms.</li> </ol>   |
| <b>Wartości uzyskane przez aktorów po zakończeniu przypadków użycia</b> | <ol style="list-style-type: none"> <li>1. Zmiana właściwości materiału woksela w miejscu wybranym przez użytkownika.</li> <li>2. Obiekt posiada zmienioną specyfikację materiału w klikniętym wokselu.</li> </ol>   |



### 3. Zastosowane technologie i rozwiązania

Aby stworzyć aplikację, wpierv trzeba podjąć decyzję dotyczącą technologii w jakiej ma ona powstać. Programy działające w czasie rzeczywistym, wymagają dużej odpowiedzialności ze strony programisty, by zapewnić użytkownikowi jak najpłynniejsze doświadczenie podczas użytkowania. W tym celu, wybrano rozwiązania pozwalające programiście na jak największą ingerencję w sposób działania, jednocześnie pozwalając na wykorzystanie gotowych rozwiązań.

#### 3.1 Języki programowania

Aplikacja została napisana w języku C++ w wersji ISO/IEC 14882:2011 [8] (C++11), z wyjątkiem plików cieniowania barw (ang. *shader files*), które zostały napisane w języku GLSL (OpenGL Shading Language) w wersji „330 core” [12]. Są wysokopoziomowymi językami, ze składnią pochodzącą z języka C, pozwalającymi na bezpośredni dostęp do zasobów sprzętowych i funkcji systemowych.

#### 3.2 Środowisko programistyczne

Z uwagi na wymaganie multiplatformowości postawione w etapie projektowania systemu, wszystkie narzędzia programistyczne do tworzenia oprogramowania, powinny te zapotrzebowanie spełniać. Visual Studio Code [18] jest idealnym przykładem narzędzia, będącego multiplatformowe, bezpłatne oraz elastyczne. Głównym atutem tego IDE (ang. *Integrated Development Environment*), jest ogromna biblioteka rozszerzeń. Do stworzenie aplikacji, użyte zostały następujące pozycje:

- C/C++ [16] - wsparcie dla języków C i C++.
- CMake [25] - wsparcie dla języka CMake.
- CMake Tools [17] - integracja programu CMake.
- VSCode Icons [22] - zestaw ikon.
- Path Intellisense [15] - automatyczne kończenie nazw plików.

### 3.3 Biblioteki

Jedną z największych przewag języka C++ jest łatwość korzystania z bibliotek napisanych w C, C++ lub innych, niezależnie od platformy systemowej. Kontynuując temat multiplatformowości, systemy obsługi okien nie jest wspólny dla każdego z systemów operacyjnych. W celu rozwiązania tego problemu, użyta została biblioteka GLFW [6], pełniąca rolę interfejsu programistycznego aplikacji (ang. *Application Programming Interface*), wspierająca systemy okien takie jak Windows, macOS, X11 oraz Wayland.

Silniki graficzne są zaawansowanymi modułami, które wykonują wiele obliczeń matematycznych. W celu zapewnienia najwyższej wydajności działania tych obliczeń, jak i zapewnieniu zgodności struktur matematycznych pomiędzy kodem pisanym w C++, a kodem pisanym w GLSL, wybrano bibliotekę glm [3].

Do komunikacji pomiędzy użytkownikiem a silnikiem 3D, wykorzystano bibliotekę ImGui [19], pełniącą rolę graficznego interfejsu użytkownika (ang. *GUI Graphical User Interface*). Została stworzona specjalnie na potrzeby szybkich iteracji, potrzebnych między innymi w silnikach gier, aplikacjach 3D czasu rzeczywistego oraz aplikacjach pełnoekranowych.

Zastosowanie OpenGL w aplikacji wymaga użycia biblioteki ładującej, odpowiedzialnej za ładowanie wskaźników, funkcji w czasie rzeczywistym, jąder, jak i rozszerzeń. [11] Jedną z takich bibliotek jest Glad [4], która jest generowana przez użytkownika na podstawie jego potrzeb.

## **4. Implementacja**



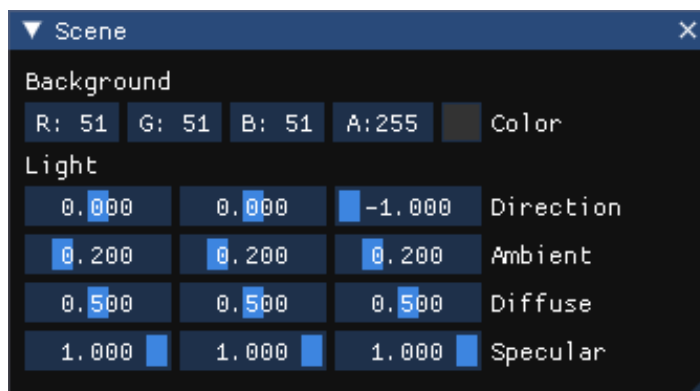
## 5. Prezentacja aplikacji

Głównym założeniem interfejsu była prostota i customizowalność. Osiągnięte to zostało poprzez wyeksponowanie modelu, który jest renderowany w czasie rzeczywistym przez silnik 3D edytora.



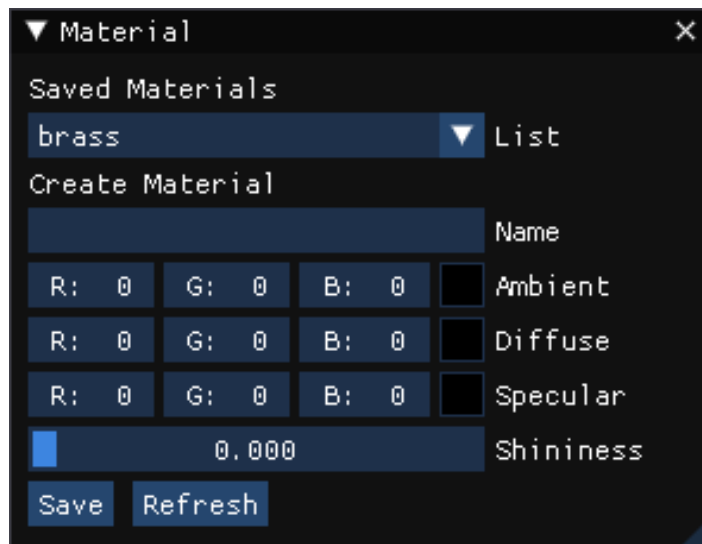
Rysunek 5.1: Ekran startowy programu, źródło: opracowanie własne

By użytkownikowi dać więcej możliwości ustawienia wyglądu wyjściowego, dodano opcję zmiany wartości oświetlenia, jak i tła aplikacji. Wygląd okna „Scene” został ukazany na rysunku 5.2.



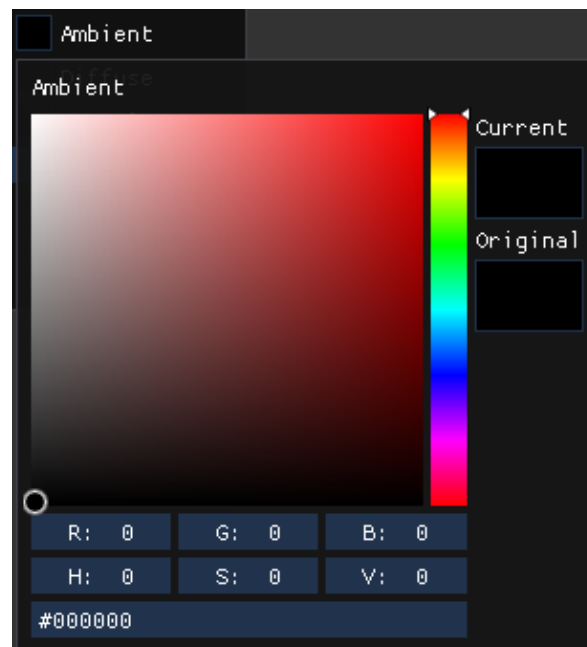
Rysunek 5.2: Okno zmiany ustawień sceny, źródło: opracowanie własne

Zmiana aktywnego materiału odbywa się poprzez wybór z listy dostępnych, jak i stworzonych materiałów w oknie „Material” (rysunek 5.3).



Rysunek 5.3: Okno zmiany bieżącego materiału, jak i jego edycji, źródło: opracowanie własne

W celu ułatwienia wyboru koloru podczas tworzenia materiału oraz wyboru tła sceny, parametry „Color” w przypadku okna „Scene”, jak i „Ambient”, „Diffuse” i „Specular” w przypadku okna „Material” posiadają opcję wyboru ręcznego z kwadratu kolorów (rysunek 5.4).

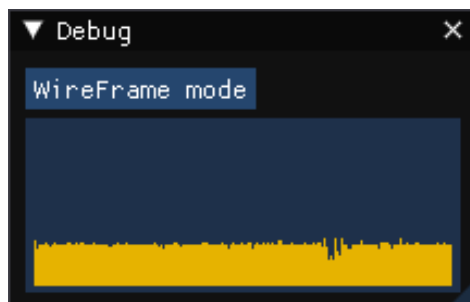


Rysunek 5.4: Okno wyboru wartości koloru wraz z podglądem po prawej stronie, źródło: opracowanie własne

Okno „Debug” (rysunek 5.5) powstało specjalnie z uwagą na wymagania czasowe dotyczące głównych funkcjonalności. Przycisk „WireFrame mode” przełącza w silniku 3D

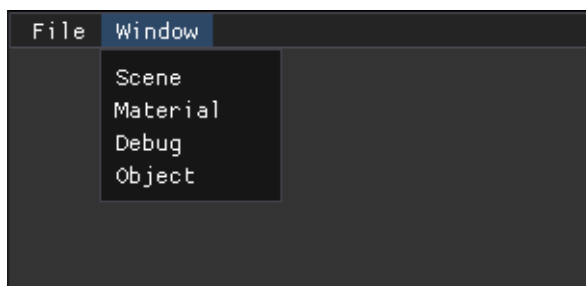


sposób renderowania obiektów na tylko krawędzie. Pod przyciskiem w czasie rzeczywistym kreślony jest wykres czasu renderowania pojedynczej klatki (ang. *frame time*).

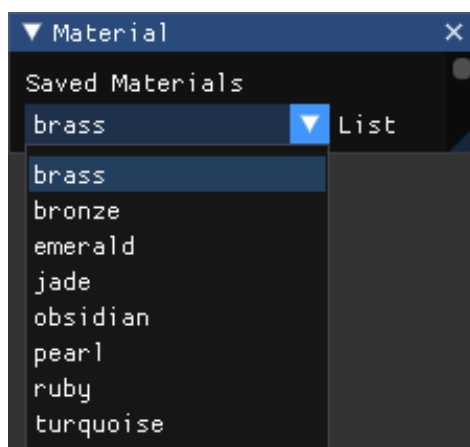


Rysunek 5.5: Okno z informacjami o, źródło: opracowanie własne

Przedstawione powyżej okna aplikacji, są dostępne z poziomu paska nawigacji (rysunek 5.6) za pomocą którego można je pokazywać lub chować. Dla każdego z tych okienek, użytkownik może zmienić ich pozycję, wielkość, jak i zminimalizować, według własnego uznania. Na rysunku 5.7 przedstawione zostało zmniejszone okno „Material” z rysunku 5.3.



Rysunek 5.6: Pasek nawigacyjny z wybraną opcją „Window”, źródło: opracowanie własne



Rysunek 5.7: Okno „Material” ze zmienionym rozmiarem przez użytkownika, źródło: opracowanie własne



## **6. Rozdział 6**



## **Podsumowanie**

Tutaj będzie podsumowanie.



## Bibliografia

- [1] J. Autor. Nazwa strony internetowej. <http://www.dlugi.adres.url.zlamie.sie.gdzies.w.srodku.com>, stan z 01.01.2010 r.
- [2] U. Autor and W. Kolejny. Tytuł publikacji. *Nazwa czasopisma*, 12(2):132–145, May 2012.
- [3] G-Truc Creation. OpenGL Mathematics (GLM). <https://glm.g-truc.net/0.9.9/index.html>, stan z 18.02.2021 r.
- [4] Davldde. glad. <https://glad.davld.de/>, stan z 18.02.2021 r.
- [5] Joey de Vries. *Learn OpenGL: Learn modern OpenGL graphics programming in a step-by-step fashion*. KW Publishers, s. 69-, 2020.
- [6] elmindreda. GLFW. <https://www.glfw.org/>, stan z 18.02.2021 r.
- [7] ephtracy. MagicaVoxel. <https://ephtracy.github.io>, stan z 04.02.2021 r.
- [8] International Organization for Standardization. ISO/IEC 14882:2011 standard. <https://www.iso.org/standard/50372.html>, stan z 18.02.2021 r.
- [9] Minddesk Software GmbH. Qubicle. <https://www.minddesk.com>, stan z 04.02.2021 r.
- [10] LLC Go Real Games. Mega Voxels Play. <https://www.megavoxels.com>, stan z 04.02.2021 r.
- [11] The Khronos Group. OpenGL Loading Library. [https://www.khronos.org/opengl/wiki/OpenGL\\_Loading\\_Library](https://www.khronos.org/opengl/wiki/OpenGL_Loading_Library), stan z 18.02.2021 r.
- [12] The Khronos Group. The OpenGL® Shading Language specification. <https://www.khronos.org/registry/OpenGL/specs/gl/GLSLangSpec.3.30.pdf>, stan z 18.02.2021 r.
- [13] guillaumechereau. Goxel. <https://goxel.xyz>, stan z 04.02.2021 r.
- [14] B. Inny. Tytuł publikacji. In *Tytuł książki*, pages 5–32, Feb 2011.

- [15] Christian Kohler. Path Intellisense extension for Visual Studio Code. <https://marketplace.visualstudio.com/items?itemName=christian-kohler.path-intellisense>, stan z 18.02.2021 r.
- [16] Microsoft. C/C++ extension for Visual Studio Code. <https://marketplace.visualstudio.com/items?itemName=ms-vscode.cpptools>, stan z 18.02.2021 r.
- [17] Microsoft. CMake Tools extension for Visual Studio Code. <https://marketplace.visualstudio.com/items?itemName=ms-vscode.cmake-tools>, stan z 18.02.2021 r.
- [18] Microsoft. Visual Studio Code. <https://code.visualstudio.com>, stan z 18.02.2021 r.
- [19] ocornut. Dear ImGui. <https://github.com/ocornut/imgui>, stan z 18.02.2021 r.
- [20] Pixowl. VoxEdit Beta. <https://www.voxedit.io>, stan z 04.02.2021 r.
- [21] Samuel R. Buss Sam Buss. *3D Computer Graphics: A Mathematical Introduction with OpenGL*. Cambridge University Press, s. 69-87, 2003.
- [22] VSCode Icons Team. vscode-icons extension for Visual Studio Code. <https://marketplace.visualstudio.com/items?itemName=vscode-icons-team.vscode-icons>, stan z 18.02.2021 r.
- [23] Z. Test and K. Następny. *Tytuł książki*. Wydawca, Adres, 1995.
- [24] Naty Hoffman Tomas Akenine-Moller, Eric Haines. *Real-Time Rendering, Fourth Edition*. CRC Press, s. 578, 2018.
- [25] twxs. CMake extension for Visual Studio Code. <https://marketplace.visualstudio.com/items?itemName=twxs.cmake>, stan z 18.02.2021 r.



## Spis tabel

|             |   |    |
|-------------|---|----|
| Tablica 2.1 | Opis przypadku użycia „Dodaj woksel” . . . . .    | 23 |
| Tablica 2.2 | Opis przypadku użycia „Usuń woksel” . . . . .     | 24 |
| Tablica 2.3 | Opis przypadku użycia „Edytuj materiał” . . . . . | 25 |



## Spis rysunków

|             |   |    |
|-------------|---|----|
| Rysunek 1.1 | Ekran startowy programu MagicaVoxel (Windows), źródło: [7] . . .  | 13 |
| Rysunek 1.2 | Ekran startowy programu Mega Voxels Play (Android), źródło: [10]  | 14 |
| Rysunek 1.3 | Ekran startowy programu Qubicle (Windows, Steam), źródło: [9] .   | 15 |
| Rysunek 1.4 | Ekran Startowy programu Goxel (Windows), źródło: [13] . . . . .   | 16 |
| Rysunek 1.5 | Ekran startowy programu VoxEdit Beta (Windows), źródło: [20] . .  | 17 |
| Rysunek 2.1 | Diagram stanów dla edycji, źródło: opracowanie własne . . . . .   | 20 |
| Rysunek 2.2 | Diagram stanów dla wczytywania, źródło: opracowanie własne . . .  | 21 |
| Rysunek 2.3 | Diagram przypadków użycia, źródło: opracowanie własne . . . . .   | 22 |
| Rysunek 5.1 | Ekran startowy programu, źródło: opracowanie własne . . . . .   | 31 |
| Rysunek 5.2 | Okno zmiany ustawień sceny, źródło: opracowanie własne . . . . .  | 31 |
| Rysunek 5.3 | Okno zmiany bieżącego materiału, jak i jego edycji, źródło:<br>opracowanie własne . . . . .             | 32 |
| Rysunek 5.4 | Okno wyboru wartości koloru wraz z podglądem po prawej stronie,<br>źródło: opracowanie własne . . . . . | 32 |
| Rysunek 5.5 | Okno z informacjami o, źródło: opracowanie własne . . . . .   | 33 |
| Rysunek 5.6 | Pasek nawigacyjny z wybraną opcją „Window”, źródło: opracowa-<br>nie własne . . . . .                   | 33 |
| Rysunek 5.7 | Okno „Material” ze zmienionym rozmiarem przez użytkownika,<br>źródło: opracowanie własne . . . . .      | 33 |



## **Spis listingów**



## **Spis algorytmów**