
	<p align="center"> <b>Silesian University of Technology</b>  <b>Faculty of Automatic Control, Electronics and Computer Science</b>  <b>Department of Graphics, Computer Vision and Digital Systems</b> </p>			
Rok akademicki	Rodzaj studiów*	Przedmiot:	Grupa	Sekcja
<b>2024/2025</b>	<b>SSI</b>	<b>JA proj.</b>	<b>2</b>	<b>4</b>
Termin: (dzień, godzina)	środa, 14.30	Prowadzący:	<b>KH</b>	
Imię:	Kacper			
Nazwisko:	Baryłowicz			
Email:	kb305480@student.polsl.pl			
<p align="center"><b>Karta projektu</b></p>				
Temat projektu:				
<p align="center"><b>Symulowanie Daltonizmu (Deuteranopia)</b></p>				
Główne założenia projektu:				
<p><b>Opis ogólny:</b> Program symuluje ślepotę barw typu deuteranopia w obrazach graficznych z wykorzystaniem wielowątkowości. Program umożliwia przetwarzanie obrazów, symulując sposób, w jaki osoby z deuteranopią widzą kolory. Dodatkowo, przetwarzanie to zostało zaimplementowane zarówno w języku C#, jak i w asemblerze x64 (Assembly), co pozwala na porównanie efektywności obydwu podejść.</p> <p><b>Czym jest deuteranopia:</b> Deuteranopia to jeden z typów daltonizmu, który wpływa na percepcję zielonych barw. Osoby cierpiące na tę formę ślepoty barw mają problem z rozróżnianiem zieleni.</p> <p><b>Cel projektu:</b> Celem projektu jest stworzenie aplikacji, która pozwoli odpowiedzieć na pytanie, czy pisanie programu w niskopoziomowym języku (takim jak asembler) faktycznie przynosi korzyści pod względem szybkości działania w porównaniu do pisania w językach wysokopoziomowych (np. C#). Aplikacja umożliwi użytkownikowi wybór metody przetwarzania obrazu – w asemblerze (ASM) albo w C# – oraz pozwala określić liczbę wątków, które mają być użyte do wykonania obliczeń. Dzięki temu można bezpośrednio porównać wydajność obu metod i sprawdzić, która z nich lepiej radzi sobie z optymalizacją czasu przetwarzania w zależności od liczby użytych wątków i rodzaju kodu.</p> <p><b>Główne założenia projektu:</b></p> <ol style="list-style-type: none"> <li>1) <i>Wczytywanie obrazu:</i> Użytkownik wybiera plik graficzny (formaty obsługiwane: JPG) za pomocą interfejsu graficznego aplikacji. Obraz ten zostaje załadowany i przygotowany do przetwarzania.</li> <li>2) <i>Przetwarzanie obrazu:</i> Istnieją dwie opcje przetwarzania obrazu: <ul style="list-style-type: none"> <li>• Przetwarzanie w C#: Symulacja deuteranopii realizowana jest w języku C#, z wykorzystaniem mechanizmu wielowątkowości (Parallel.For). Każdy wątek odpowiada za przetwarzanie jednej części obrazu, co pozwala na przyspieszenie obliczeń.</li> <li>• Przetwarzanie w ASM: Funkcjonalność symulacji deuteranopii została także zaimplementowana w asemblerze, co pozwala na szybsze przetwarzanie obrazu. Funkcja przetwarza obraz na poziomie bitów, co pozwala na optymalizację i przyspieszenie operacji.</li> </ul> </li> <li>3) <i>Zapis przetworzonego obrazu:</i> Po przetworzeniu obrazu użytkownik ma możliwość zapisania wyniku w formacie JPG.</li> </ol> <p><b>Algorytm symulowania deuteranopii:</b> Algorytm symulacji deuteranopii opiera się na zmodyfikowaniu wartości poszczególnych kanałów kolorów:</p> <ul style="list-style-type: none"> <li>• Kanał czerwony (R) jest obliczany jako: <math>R = 0.625 * R + 0.375 * G</math></li> <li>• Kanał zielony (G) zostaje zredukowany do 70% swojej oryginalnej wartości: <math>G = 0.7 * G</math></li> <li>• Kanał niebieski (B) zostaje obniżony do 80%: <math>B = 0.8 * B</math></li> </ul> <p>W ten sposób kolory są przekształcane, aby zbliżyć się do widzenia przez osobę z deuteranopią. Proces ten jest realizowany dla każdego piksela obrazu.</p> <p><b>Proces działania aplikacji:</b> Wybór obrazu (użytkownik wybiera obraz z dysku do przetworzenia) → Wybór metody przetwarzania (użytkownik decyduje, czy obraz zostanie przetworzony w języku C# czy asemblerze (ASM)) → Ustawienie liczby wątków (użytkownik może ustawić liczbę wątków, które zostaną wykorzystane w trakcie przetwarzania) Przetwarzanie obrazu (przetwarzanie w C# albo w ASM, w zależności od wybranej metody) → Zapis obrazu (użytkownik wybiera miejsce, gdzie przetworzony obraz ma być zapisany na dysku).</p> <p><b>Interfejs graficzny (GUI)</b> Aplikacja posiada graficzny interfejs użytkownika (GUI) zaprojektowany z wykorzystaniem XAML (Extensible Application Markup Language). XAML pozwala na łatwe tworzenie przejrzystego i intuicyjnego interfejsu, który umożliwia użytkownikowi interakcję z programem w prosty sposób.</p>				