KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu
Zaawansowane technologie informatyczne [S2Elmob1-SPE>ZTI]

Przedmiot

Kierunek studiów
Elektromobilność

Rok/Semestr
2/3

Studia w zakresie (specjalność)
Systemy przetwarzania energii

Profil studiów
ogólnoakademicki

Poziom studiów
drugiego stopnia

Język oferowanego przedmiotu
polski

Forma studiów
stacjonarne

Wymagalność
obligatoryjny

Liczba godzin

Wykład
15

Laboratorium
30

Inne
0

Ćwiczenia
0

Projekty/seminaria
0

Liczba punktów ECTS

3,00

Koordynatorzy

dr inż. Konrad Górny
konrad.gorny@put.poznan.pl

dr hab. inż. Wojciech Pietrowski prof. PP
wojciech.pietrowski@put.poznan.pl

Wykładowcy

Wymagania wstępne

Student rozpoczynający ten przedmiot powinien posiadać podstawowe wiadomości z obsługi: komputera, systemu operacyjnego Windows oraz umiejętność programowania strukturalnego i obiektowego w języku Python, C++, Java lub C# . Student rozpoczynający ten przedmiot powinien posiadać podstawowe wiadomości w zakresie informatyki, systemów operacyjnych, algorytmów i struktur danych, języków programowania. Student rozpoczynający ten przedmiot powinien posiadać podstawowe wiadomości z geometrii analitycznej i różniczkowej, rachunku macierzowego.

Cel przedmiotu

Zapoznanie z podstawowymi zagadnieniami i pojęciami związanymi z językiem Python w zakresie umożliwiającym przetwarzanie sekwencji znaków i plików tekstowych. Zapoznanie z praktycznym zastosowaniem języków skryptowych do przetwarzania sygnałów. Zapoznanie z podstawowymi informacjami o platformie zaawansowanego środowiska programowania wizualnego .NET. Nabycie umiejętności projektowania oraz implementowania aplikacji okienkowych w języku C#. Testowanie tworzonych aplikacji. Zapoznanie ze współczesnymi metodami tworzenia trójwymiarowej grafiki komputerowej. Poznanie zasady działania omawianych algorytmów tworzenia grafiki. Zapoznanie z zaawansowanymi strukturami sieci neuronowych, elementami uczenia maszynowego.

Przedmiotowe efekty uczenia się

Wiedza:
Student ma poszerzoną i pogłębioną wiedzę w zakresie wybranych działów matematyki niezbędną do opisu elementów, układów i systemów stosowanych w elektromobilności.
Student ma poszerzoną wiedzę w zakresie technik programowania oraz stosowania nowoczesnych narzędzi informatycznych do analizy i syntezy układów elektrycznych pojazdów hybrydowych i elektrycznych w tym trakcyjnych.
Student ma podbudowaną teoretycznie wiedzę na temat nowoczesnych metod gromadzenia, przetwarzania i analizy danych, także w zakresie stosowania uczenia maszynowego.

Umiejętności:
Student potrafi pozyskać informacje (w języku polskim i angielskim) z różnych źródeł, dokonywać ich interpretacji, krytycznej oceny, analizy i syntezy, a także wyciągać wnioski oraz formułować i uzasadniać opinie.
Student potrafi przy gromadzeniu, przetwarzaniu i analizie danych stosować nowoczesne narzędzia informacyjno-komunikacyjne, zaawansowane techniki programowania oraz metody uczenia maszynowego.

Kompetencje społeczne:
Student ma świadomość znaczenia najnowszych osiągnięć naukowych i technicznych w rozwiązywaniu problemów badawczych i praktycznych oraz w razie potrzeby wspierania się opiniami ekspertów.

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:

Wykład: zaliczenie na podstawie kolokwium składającego się z pytań ogólnych i testowych. Skala ocen 51-60% pkt. dst, 61-70% pkt dst+, 71-80% pkt. db, 81-90% pkt. db+, 91-100% pkt. bdb.
Laboratorium: premiowanie praktycznej wiedzy zdobytej w trakcie poprzednich ćwiczeń laboratoryjnych, sprawdzenie praktycznych umiejętności programowania w języku Python (kolokwium zaliczeniowe), ocena wiedzy i umiejętności związanych z realizacją indywidualnych i grupowych projektów programistycznych.
Uzyskiwanie punktów dodatkowych za aktywność podczas zajęć, a szczególnie za: umiejętność współpracy w ramach zespołu praktycznie realizującego zadanie szczegółowe w laboratorium, wykorzystanie elementów i technik wykraczających poza materiał z zakresu prowadzonego wykładu i ćwiczeń laboratoryjnych, staranność estetyczną zrealizowanych projektów.

Treści programowe

Język C++, C#. Programowanie obiektowe. Wprowadzenie do platformy .NET. Przedstawienie struktury platformy .NET. Charakterystyka pakietu Microsoft Visual Studio. Podstawy programowania w języku C#. Zaawansowane elementy języka C#. Wykorzystanie zaawansowanych platform programistycznych w zagadnieniach związanych ze sztuczną inteligencją, uczeniu maszynowym, uczeniu sieci neuronowych. Wykorzystanie GPGPU, obliczeń równoległych oraz akceleracji sprzętowej w uczeniu sieci neuronowych, uczeniu maszynowym.

Tematyka zajęć

Wykład:
Język C++, C#. Programowanie obiektowe. Klasy, obiekty, dziedziczenie, hermetyzacja danych,

polimorfizm oraz metody wirtualne, klasy abstrakcyjne, szablony funkcji, szablony klas. Wprowadzenie do platformy .NET. Przedstawienie struktury platformy .NET. Rodzaje i typy platformy .NET. Zarys środowiska .NET Framework. Przegląd języków programowania dla platformy.NET. Charakterystyka pakietu Microsoft Visual Studio. Prezentacja środowiska programistycznego. Środowisko CLR (ang. Common Language Runtime). Podstawowe funkcje i usługi CLR. Zarządzanie pamięcią i zasobami. Podstawy programowania w języku C#. Omówienie składni języka: instrukcje, zmienne, operatory i typy danych. Dostęp i operacje na plikach. Zasady tworzenie klas, metod, konstruktorów oraz obiektów tych klas. Użycie tablic. Przegląd podstawowych narzędzi zawartych w SDK (ang. Software Development Kit). Zaawansowane elementy języka C#.
Procedury tworzenia trójwymiarowej grafiki komputerowej w języku wysokiego poziomu. Graficzna prezentacja wyników badań z wykorzystaniem OpenGL. Analiza fragmentu świata rzeczywistego w celu zbudowania własnej grafiki komputerowej. Przygotowanie scenariusza i realizacja animacji komputerowej.
Zapoznanie z zaawansowanymi strukturami sieci neuronowych, elementami uczenia maszynowego.
Laboratorium:
Tworzenie i przetwarzanie obiektów, obiekty jako argumenty funkcji. Komponenty, formularze, właściwości, zdarzenia, obsługa wyjątków. Edycja formularzy. Uruchamianie aplikacji. Tworzenie przykładowej aplikacji dla systemu Windows.Dyrektywy preprocesora. Obsługa zdarzeń. Obsługa błędów za pomocą wyjątków. Korzystanie z komponentów interfejsu Windows. Biblioteka Windows Forms. Wykorzystanie przestrzeni nazw, formularz początkowy i jego właściwości. Dodawanie elementów sterujących, określanie ich właściwości i definiowanie funkcji obsługi zdarzeń. Obsługa myszki i klawiatury, menu, pasek statusu, pasek narzędzi. Podstawowe elementy sterujące: przyciski, pola tekstowe, listy rozwijalne itp. Tworzenie wykresów. Okna dialogowe: dialogi standardowe i niestandardowe. Zakładki. Pozyskiwanie obiektu graphics. Narzędzia graficzne: czcionki, pióra pędzle. Funkcje rysujące i wypełniające. Przetwarzanie obrazów. Przetwarzanie pikseli, przetwarzanie obrazów. Rysowanie obiektów w trzech wymiarach. Przekształcenia geometryczne, obrót, przesunięcie, skalowanie. Rzutowanie perspektywiczne i prostopadłe. Kolorowanie i cieniowanie. Światło i cienie. Odwzorowanie tekstur. Mieszanie kolorów i przezroczystość. Antyaliasing. Krzywe i powierzchnie parametryczne.
Wykorzystanie zaawansowanych platform programistycznych w zagadnieniach związanych ze sztuczną inteligencją, uczeniu maszynowym, uczeniu sieci neuronowych. Wykorzystanie GPGPU, obliczeń równoległych oraz akceleracji sprzętowej w uczeniu sieci neuronowych, uczeniu maszynowym.

Metody dydaktyczne

Wykład: prezentacja zagadnień z wykorzystaniem środków multimedialnych, przykłady (np. obliczeniowe) podawane na tablicy, dyskusja nad zagadnieniami problemowymi.
Laboratorium: wykonywanie ćwiczeń laboratoryjnych w zespołach pod kontrolą prowadzącego.

Literatura

Podstawowa:
Stroustrup Bjarne, Programowanie : teoria i praktyka z wykorzystaniem C++, Helion 2020
Grębosz Jerzy, Opus Magnum C++11 : programowanie w języku C++, Helion 2020
Zieliński Józef, Podstawy programowania w języku C++, Oficyna Wydawnicza "Impuls", 2013
Graham Sellers, Richard S. Wright Jr., Nicholas Haemel, OpenGL Superbible: Comprehensive Tutorial and Reference (7th Edition), Helion 2016
Von Glitschka, Vector Basic Training: A Systematic Creative Process for Building Precision Vector Artwork (2nd Edition), Helion 2016
Dokumentacja elektroniczna systemu programowania Visual Studio.NET
Python 3 : kompletne wprowadzenie do programowania / Mark Summerfield ; [tł. Robert Górczyński]., Gliwice, Helion 2010.

Uzupełniająca:
Griffiths Ian, Adams Matthew, Liberty Jesse, C#. Programowanie, Helion, 2012
F. P. Preparata, M. I. Samos, Geometria obliczeniowa, Helion 2003
M. Jankowski, Elementy grafiki komputerowej, WNT 2006.
P. Kiciak, Podstawy modelowania krzywych i powierzchni. Zastosowania w grafice komputerowej, WNT 2005.
Graham Sellers, Richard S. Wright Jr., Nicholas Haemel, OpenGL Superbible: Comprehensive Tutorial and

Reference (7th Edition), Helion 2016

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	87	3,00
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	47	1,50
Praca własna studenta (studia literaturowe, przygotowanie do zajęć laboratoryjnych/ćwiczeń, przygotowanie do kolokwiów/egzaminu, wykonanie projektu)	40	1,50