KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu
Podstawowe Narzędzia i Metody Programowania Robotów Autonomicznych

Przedmiot

Kierunek studiów
Automatyka i Robotyka
Studia w zakresie (specjalność)
Robotyka i Systemy Autonomiczne
Poziom studiów
drugiego stopnia
Forma studiów
stacjonarne

Rok/semestr
1/1
Profil studiów
ogólnoakademicki
Język oferowanego przedmiotu
polski
Wymagalność
obligatoryjny

Liczba godzin

Wykład
30
Laboratoria
30
Inne (np. online)
0
Ćwiczenia
0
Projekty/seminaria
0

Liczba punktów ECTS

4

Wykładowcy

Odpowiedzialny za przedmiot/wykładowca:
dr hab. inż. Dominik Belter

Odpowiedzialny za przedmiot/wykładowca:

Wymagania wstępne

Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę z podstaw robotyki i programowania. Powinien również posiadać umiejętność pozyskiwania informacji ze wskazanych źródeł oraz mieć gotowość do podjęcia współpracy w ramach zespołu

Cel przedmiotu

Przekazanie studentom wiedzy z zakresu narzędzi używanych do programowania robotów autonomicznych, poprawnego wykorzystania tych narzędzi i integracji systemów sterowania i dobierania narzędzi do rzeczywistych problemów.

Przedmiotowe efekty uczenia się

Wiedza

1. Ma zaawansowaną i pogłębioną wiedzę w zakresie metod analizy i projektowania systemów sterowania.

2. Ma uporządkowaną i pogłębioną wiedzę związaną z systemami sterowania i układami kontrolno-pomiarowymi.

3. Ma podstawową wiedzę o cyklu życia systemów automatyki i robotyki oraz układów kontrolno-pomiarowych.

Umiejętności

1. Potrafi posługiwać się technikami informacyjno-komunikacyjnymi

2. Potrafi zintegrować i zaprogramować specjalizowane systemy zrobotyzowane

3. Potrafi dokonać krytycznej analizy sposobu funkcjonowania systemów sterowania i systemów robotyki; posiada także umiejętność doboru systemów automatyki z wykorzystaniem sterowników mikroprocesorowych

4. Potrafi skonstruować algorytm rozwiązania złożonego i nietypowego zadania inżynierskiego i prostego problemu badawczego oraz zaimplementować, przetestować i uruchomić go w wybranym środowisku programistycznym dla wybranych systemów operacyjnych

Kompetencje społeczne

1. Posiada świadomość konieczności profesjonalnego podejścia do zagadnień technicznych, skrupulatnego zapoznania się z dokumentacją oraz warunkami środowiskowymi, w których urządzenia i ich elementy mogą funkcjonować

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:

Wiedza nabyta w ramach wykładu jest weryfikowana przez jeden 45-minutowy egzamin realizowany w sesji egzaminacyjnej. Egzamin składa się z 20-30 pytań (testowych) i do 5 pytań otwartych, różnie punktowanych. Próg zaliczeniowy: 50% punktów. Zagadnienia do egzaminu, na podstawie których opracowywane są pytania są udostępniane na wykładzie.

Umiejętności nabyte w ramach zajęć laboratoryjnych weryfikowane są podstawie kolokwium zaliczeniowego, składającego się z 20 pytań oraz sprawdzeniu realizacji zadania praktycznego realizacji problemu planowania ruchu. Próg zaliczeniowy: 50% punktów.

Treści programowe

Wykład:

1. Skrypty systemowe w bashu/python, cron, bashrc, services

2. Programowanie układu Discovery z poziomu Linuxa

3. udev rules, stałe nazwy dla urządzeń USB, low latency dla komunikacji USB

4. węzeł w ROSie do komunikacji w USB i publikujący dane

5. Przetwarzanie współbieżne w C++ (wątki, procesy)

6. CUDA (wykonywanie operacji na karcie graficznej)

7. Tensorflow + ROS (uruchomienie sieci wykrywającej obiekty w ROS)

8. Remote Master (ROS na wielu komputerach)

9 Wykrywanie obiektów ArUco do kalibracji

10 ROS bags (zebranie danych z kamery do kalibracji)

11 TFy w ROSie (odczytywanie przekształceń z wcześniej zapisanych ROS-bagów)

12 Kalibracja kamer easyHandEye (na ROS-bagach)

Laboratorium:

1. Skrypty systemowe w bashu/python, cron, bashrc, services

2. Programowanie układu Discovery z poziomu Linuxa

3. udev rules, stałe nazwy dla urządzeń USB, low latency dla komunikacji USB

4. węzeł w ROSie do komunikacji w USB i publikujący dane

5. Przetwarzanie współbieżne w C++ (wątki, procesy)

6. CUDA (wykonywanie operacji na karcie graficznej)

7. Tensorflow + ROS (uruchomienie sieci wykrywającej obiekty w ROS)

8. Remote Master (ROS na wielu komputerach)

9 Wykrywanie obiektów ArUco do kalibracji

10 ROS bags (zebranie danych z kamery do kalibracji)

11 TFy w ROSie (odczytywanie przekształceń z wcześniej zapisanych ROS-bagów)

12 Kalibracja kamer easyHandEye (na ROS-bagach)

Metody dydaktyczne

1. Wykład: prezentacja multimedialna, ilustrowana przykładami podawanymi na tablicy.

2. Ćwiczenia laboratoryjne: instrukcje realizowane na komputerach i robotach dostępnych w laboratorium

Literatura

Podstawowa

Mark Mitchell, Jeffrey Oldham, Alex Samuel, Advanced Linux Programming, New Riders Publishing

Robot Operating System (ROS), Springer 2016

Uzupełniająca

M. Galewski, STM32. Aplikacje i ćwiczenia w języku C, Wydawnictwo BTC, Legionowo 2011

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	100	4
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	60	2,5
Praca własna studenta (studia literaturowe, przygotowanie do zajęć laboratoryjnych/ćwiczeń, przygotowanie do kolokwiów/egzaminu, wykonanie projektu)[1]	40	1,5

1 niepotrzebne skreślić lub dopisać inne czynności