ZESZYTY NAUKOWE POLITECHNIKI RZESZOWSKIEJ 296, Elektrotechnika 36
RUTJEE, z. 36 (1/2017), styczeń-czerwiec 2017, s. 41-51

Tomasz ŚLIWA[1]

PROTOTYPOWY TRÓJKOŁOWY MINIROBOT LABORATORYJNY

W artykule przedstawiono budowę kołowego minirobota laboratoryjnego mogącego służyć początkowo jako platforma do badania czujników, a w przyszłości do oceny inteligentnych algorytmów nawigacyjnych w warunkach rzeczywistego przemieszczania. Przy konstrukcji robota kierowano się dostępnością elementów, łatwością montażu oraz niską ceną. Robot został wyposażony w mikrokomputer Raspberry Pi 2 z systemem operacyjnym Linux Raspbian, który wraz z oprogramowaniem aplikacyjnym stanowi zarówno system sterujący jak i platformę rozwojową umożliwiającą tworzenie i uruchamianie programów bezpośrednio w minirobocie. Do komunikacji służą usługi terminalowe, dzięki którym możliwe jest wygodne programowanie robota z dowolnego miejsca z łączem internetowym. Opisana konstrukcja została zaprezentowana na rzeszowskich zawodach ROBO~motion 2016, na potrzeby których opracowano i zaprogramowano trzy tryby pracy.

Słowa kluczowe: Robot mobilny, Raspberry Pi 2, czujniki ultradźwiękowe, IMU

1. Wprowadzenie

Rozwój autonomicznych pojazdów pociąga za sobą konieczność stosowania znacznej liczby czujników, takich jak czujniki odległości, prędkości, przyspieszenia, kursu czy pozycji. Mały trójkołowy robot mobilny jako niedroga, łatwa do skonstruowania i dalszej rozbudowy platforma, może być użyteczny w procesie testowania czujników w warunkach ruchu zbliżonych do docelowego środowiska.

Od pewnego czasu powstaje wiele różnorodnych konstrukcji robotów mobilnych. Budowane są zarówno małe, hobbystyczne jednostki przeznaczone do udziału w zawodach [3, 5], nieco bardziej skomplikowane roboty edukacyjne m.in. z wizyjnym sprzężeniem zwrotnym zapewnianym przez zewnętrzną kamerę [2] jak i jeszcze bardziej skomplikowane, wielokołowe konstrukcje z manipulatorem [1]. Istnieją także gotowe zestawy edukacyjne takie jak robot Khepera [7, 12] czy Lego Mindstorms [6, 13]. Nawet małe konstrukcje bywają skompli-

[1] Tomasz Śliwa, Politechnika Rzeszowska, Wydział Elektrotechniki i Informatyki, Katedra Informatyki i Automatyki, al. Powstańców Warszawy 12 35-959 Rzeszów, 17 865 1490, tomeks@kia.prz.edu.pl

kowane i kosztowne, dlatego w proponowanym rozwiązaniu postawiono na prostotę i dostępność części.

W artykule opisano pierwszą fazę budowy minirobota konstruowanego w oparciu o uniwersalne, trójkołowe podwozie napędzane dwoma silnikami prądu stałego (DC). Moc oraz kierunek obrotów ustawiana jest indywidualnie dla każdego z silników, co zapewnia dobrą manewrowość. Jako modułu obliczeniowego użyto popularnego mikrokomputera Raspberry Pi 2 Model B [8] (w skrócie RPi2), którego zasoby sprzętowe i moc obliczeniowa zapewniają obsługę algorytmów sterujących, środowiska programowania oraz usług pomocniczych, np. zdalnego dostępu. Do wykrywania przeszkód w otoczeniu robota użyto czujników ultradźwiękowych. Całość oprogramowania niezbędnego do działania i obsługi zlokalizowano w samym robocie. Dostęp do zasobów robota możliwy jest ze zdalnego terminala, dzięki czemu nad rozwojem robota może pracować jednocześnie cały zespół współpracujących użytkowników. Po wstępnych testach sprzętu i oprogramowaniu minirobota, zaprezentowano go na rzeszowskich Międzynarodowych Zawodach Robotów ROBO~motion 2016.

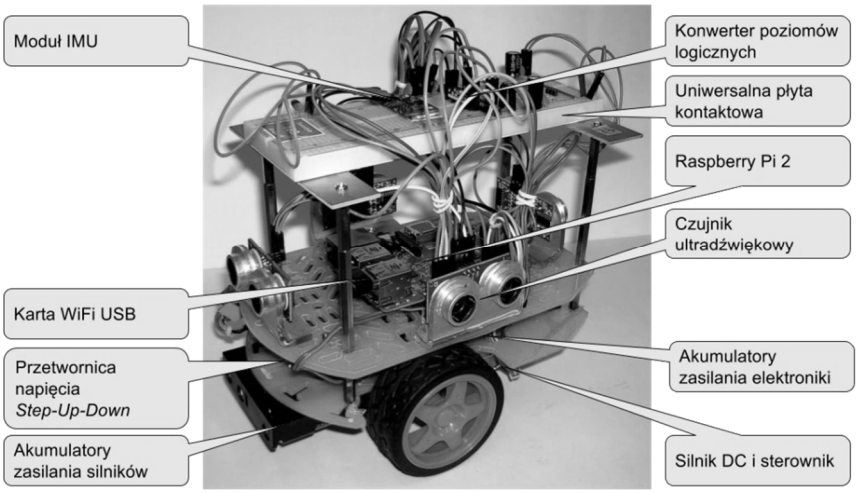
2. Budowa robota

Robot zbudowany został na bazie uniwersalnego, trójkołowego podwozia wyposażonego w dwa koła napędzane silnikami DC i jedno koło wsporcze. Komputer steruje kierunkiem obrotów oraz mocą silników DC [4] za pomocą sterownika - *drivera* kontrolowanego sygnałem PWM (*Pulse-Width Modulation*). Do pomiaru odległości od przeszkód na pokładzie zainstalowano cztery czujniki ultradźwiękowe. Znalazł się tam także układ IMU (*Inertial Measurement Unit*) zawierający żyroskop, magnetometr (kompas), akcelerometr, barometr i termometr. Ze względu na różnice w poziomach sygnałów cyfrowych poszczególnych modułów zastosowano konwerter poziomów logicznych. Do zasilania układów elektronicznych robota użyto 4 szeregowo połączone akumulatory NiMH, których napięcie stabilizuje przetwornica *Step-Up-Down*. W celu eliminacji zakłóceń wprowadzanych przez silniki DC sterowane sygnałem PWM, zastosowano osobne zasilanie silników z kolejnych 4 szeregowo połączonych akumulatorów NiMH. Zmontowany robot jest pokazany na rysunku 1.

Silniki DC DG01D 48:1 zamontowane w podwoziu zasilane są napięciem 5V, a do ich sterowania służy zintegrowany moduł drivera DRV8835. Umożliwia on za pomocą sygnałów PWM sterowanie mocą i kierunkiem obrotów. Wykorzystano także dodatkowe wejście zasilania *drivera*, aby doprowadzić napięcie do silników z osobnego zestawu akumulatorów.

Zastosowany mikrokomputer Raspberry Pi 2 posiada 40-pinowe złącze GPIO (*General Purpose Input/Output*), na które wyprowadzono standardowe sygnały magistral komunikacyjnych I2C, SPI, UART, linie zasilania +5V

i +3,3V oraz uniwersalne piny GPIO umożliwiające sterowanie urządzeniami peryferyjnymi. Wejścia/wyjścia GPIO pracują na poziomie logicznym 3,3V.



Rys. 1. Zmontowany robot mobilny

Fig. 1. Full-featured mobile robot

Zastosowany mikrokomputer Raspberry Pi 2 posiada 40-pinowe złącze GPIO (*General Purpose Input/Output*), na które wyprowadzono standardowe sygnały magistral komunikacyjnych I2C, SPI, UART, linie zasilania +5V i +3,3V oraz uniwersalne piny GPIO umożliwiające sterowanie urządzeniami peryferyjnymi. Wejścia/wyjścia GPIO pracują na poziomie logicznym 3,3V. Podanie 5V na wejście RPi2 może spowodować uszkodzenie. Jako układ IMU zastosowano moduł GY-80 z interfejsem I2C, zawierający układy:

- akcelerometr ADXL245B,
- żyroskop L3H4200D,
- magnetometr HMC5883L,
- barometr i termometr BMP085.

Do pomiaru odległości służą 4 czujniki ultradźwiękowe HC-SR04 o zasięgu 2m umieszczone z przodu, z tyłu i na bokach robota. Do komunikacji z czujnikiem HC-SR04 wymagane jest użycie dwóch linii, jednej do wyzwolenia pomiaru, drugiej do pomiaru czasu odpowiedzi, proporcjonalnego do odległości obiektu od czujnika. Ponieważ czujnik pracuje z zasilaniem i poziomami logicznymi 5V, dlatego pomiędzy wyjściami czujników a wejściami mikrokomputera RPi2 zainstalowano konwerter poziomów logicznych z tranzystorami BSS138. Do połączenia wszystkich modułów wykorzystano uniwersalną płytę kontaktową umieszczoną w górnej części robota.

Komunikacja robota z użytkownikiem-programistą odbywa się za pomocą małej karty sieciowej WiFi TL-WN725N podłączonej do jednego z czterech portów USB mikrokomputera RPi2.

3. Raspberry Pi 2 jako komputer pokładowy

W punkcie tym krótko przedstawiono relatywnie mocny mikrokomputer Raspberry Pi 2 (rys. 2) jako główny moduł obliczeniowy robota, pracujący pod kontrolą systemu operacyjnego Linux Raspbian, a także jego oprogramowanie oraz możliwości techniczne. Ten stosunkowo niewielki moduł elektroniczny jest wyposażony w:

- procesor ARM Cortex-A7 (900MHz, cztery rdzenie, 32bit),
- 1 GB pamięci RAM (współdzielona z GPU),

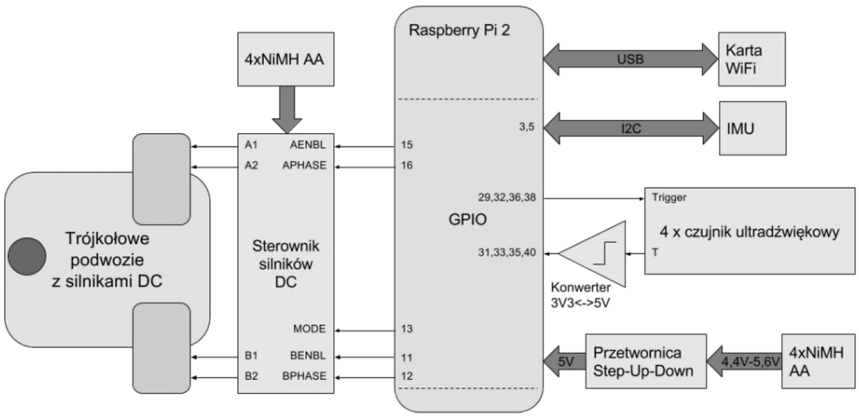
oraz SoC (*System on Chip*) i dedykowaną jednostkę GPU (*Graphics Processing Unit*). Komunikację z otoczeniem zapewnia szereg wejść i wyjść, spośród których w opisywanym zastosowaniu najważniejsze są:

- 40 pinowe złącze GPIO z interfejsami UART, I2C, SPI, PWM,
- cztery złącza USB 2.0,
- gniazdo microSD do podłączenia karty z systemem operacyjnym,
- Ethernet 10/100Mbps.

Ponadto na płytce znajduje się wyjście HDMI (*High-Definition Multimedia Interface*), wyjście analogowe audio/wideo (czteropolowy *jack* 3.5mm), interfejs kamery CSI (*Camera Serial Interface*), interfejs wyświetlacza dotykowego DSI (*Display Serial Interface*) oraz microUSB jako wejście zasilania. Na karcie pamięci SD (*Secure Digital*) zainstalowano system operacyjny Linux Raspbian oparty na popularnej dystrybucji Linux Debian. Dzięki 4 rdzeniom procesora oraz 1 GB pamięci RAM możliwe jest uruchomienie nie tylko procesów obsługujących układy peryferyjne robota, ale także bezpośrednie uruchomienie zaawansowanych środowisk programistycznych, tekstowych i graficznych usług terminalowych do bezpośredniego łączenia się z pulpitem robota, a ponadto usług plikowych i bazodanowych.

Dla modułu RPi2 jest dostępna biblioteka *wiringPi* [9] napisana w języku C zawierająca szereg funkcji umożliwiających m.in. konfigurację i bezpośredni dostęp do GPIO oraz magistral komunikacyjnych. Biblioteka wspiera:

- podstawowe operacje IO, t.j. tryb I/O pinu, rezystor „podciągający”, zapis i odczyt, zapis PWM,
- operacje czasowe (funkcje opóźniające, czas pracy procesu),
- obsługa magistral UART, I2C, SPI,
- programowy kontroler PWM,
- pseudo-przerwania, wątki, priorytet procesu,
- programowy generator prostych dźwięków.



Rys. 2. Schemat blokowy systemu robota mobilnego sterowanego przez Raspberry Pi 2

Fig. 2. System diagram of mobile robot controlled by Raspberry Pi 2

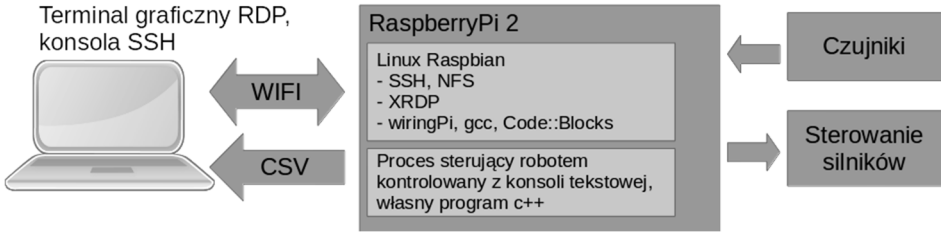
Wraz z bibliotekami programistycznymi dostarczane są narzędzia konsolowe umożliwiające dostęp do funkcji GPIO z poziomu powłoki systemu. Bibliotekę *wiringPi* przeportowano także dla innych języków programowania (Python, Ruby, PHP).

Instalacja systemu operacyjnego Raspbian polega na imporcie jego obrazu ze strony https://www.raspberrypi.org/downloads/raspbian/ i umieszczeniu go na karcie SD za pomocą zalecanego przez opiekunów systemu Raspbian oprogramowania *Win32DiskImager* [11] dla systemu Windows, *Etcher* [10] dla Mac OS lub *dd* dla systemu Linux. Podczas pierwszego uruchomienia modułu RPi2 konieczne jest podłączenie monitora i klawiatury. Dzięki wyprowadzonym złączom HDMI oraz USB nie jest to jednak trudnością. Wstępna konfiguracja polega m.in. na wyborze trybu pracy (GUI, linia poleceń), powiększeniu systemu plików do rozmiarów karty SD oraz konfiguracji połączeń internetowych. Następnie należy doinstalować pakiety wspomagające pracę (np.: *mc, htop, iftop*), kompilatory i środowiska programistyczne (*gcc, g++, Code::Blocks, Geany*), połączenie zdalne (*SSH, xrdp*), itd.

Po skonfigurowaniu i uruchomieniu połączeń zdalnych klawiatura i monitor nie są już potrzebne, ponieważ praca z modułem będzie możliwa z dowolnego komputera z zainstalowanym klientem SSH (np. *putty* dla Windows, *open-ssh* dla Linux) lub/i klientem usług terminalowych RDP (*Remote Desktop Protocol, mstsc „Podłączenie pulpitu zdalnego”* w systemie Windows, *rdesktop* lub *remmina* w systemie Linux).

4. Technika pracy z robotem

Poniżej przedstawiono technikę pracy z robotem, w tym sposób tworzenia oprogramowania i analizy wyników udostępnianych w formie pliku tekstowego CSV (*Comma-Separated Values*). Na rys. 3 pokazano ogólny schemat techniki pracy. Terminal graficzny umożliwia bezpośredni dostęp do pulpitu systemu operacyjnego zainstalowanego w RPi2 oraz uruchomienie środowiska programistycznego dla tworzenia oprogramowania przetwarzającego dane z czujników i sterującego silnikami zgodnie z określonym algorytmem. Oprogramowanie to uruchamia się bezpośrednio na RPi2, a wyniki pracy obserwuje w czasie rzeczywistym na konsoli tekstowej oraz zapisuje do tekstowego pliku CSV. Pliki CSV zawierające dane o stanie robota mogą być udostępnione do dalszej analizy za pomocą usług *nfs* (*Network File System*), *sshfs (SSH File System*, jako klienta pod systemem Windows można użyć programu *WinSCP),* FTP (*File Transfer Protocol*) lub HTTP (*Hypertext Transfer Protocol*).

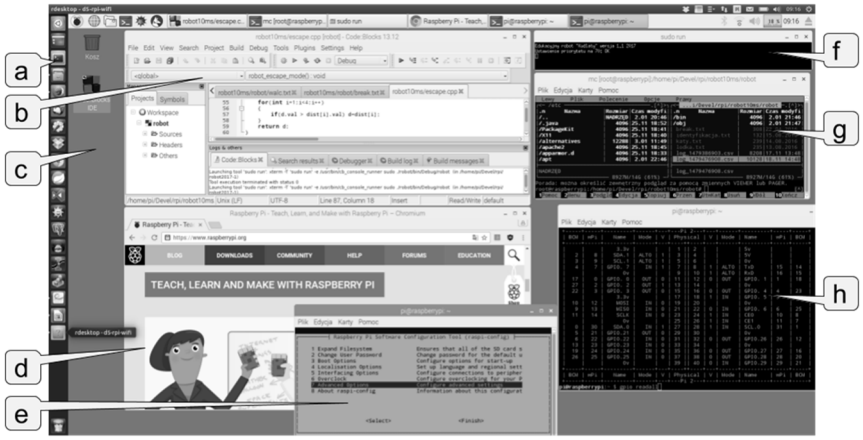


Rys. 3. Schemat blokowy techniki pracy z systemem robota

Fig. 3. Block diagram of working technique with robot system

Do tworzenia oprogramowania wykorzystano środowisko *Code::Blocks.* Jest to wieloplatformowe zintegrowane środowisko programistyczne napisane w C++ wspierające głównie języki C i C++. Funkcjonalność *Code::Blocks* opiera się na wtyczkach (*plugin*), dzięki czemu może być stosunkowo łatwo rozszerzane. *Code::Blocks* umożliwia tworzenie wieloplikowych projektów odciążając programistę od tworzenia własnych skryptów kompilacyjnych (*makefile*), posiada system podpowiedzi kodu (*intellisense*) oraz możliwość dodawania własnych poleceń. Przydaje się to np. w celu uruchomienia kompilowanego programu z wyższymi uprawnieniami wynikającymi z konieczności dostępu do urządzeń peryferyjnych, czy zmiany priorytetu wykonania procesu.

Na rysunku 4 pokazano przykładowy pulpit graficzny RPi2 z systemem Raspbian.



Rys. 4. Pulpit graficzny systemu Raspbian uruchomiony w systemie Ubuntu za pomocą terminala *rdesktop*

Fig. 4. Raspbian graphical desktop running on Ubuntu using the *rdesktop* terminal

Na rysunku 4 widoczne są następujące elementy:

a) pasek boczny systemu Ubuntu, tj. lokalnego pulpitu użytkownika,
b) zintegrowane środowisko programistyczne *Code::Blocks* uruchomione na RPi2,
c) pulpit zdalny procesora RPi2,
d) przeglądarka internetowa,
e) oprogramowanie *raspi-config* służące do podstawowej konfiguracji RPi2,
f) program sterujący robotem uruchomiony poprzez *sudo*,
g) menedżer plików *mc (Midnight Commander)*, umożliwiający m.in. szybki podgląd logów,
h) wynik polecenia *gpio readall* wyświetlający stan pinów GPIO.

Zastosowanie modelu pracy terminalowej umożliwia dostęp do robota z różnych komputerów i praktycznie z każdego miejsca, gdzie dostępne jest połączenie internetowe. Dzięki temu, że programista tworzy oprogramowanie bezpośrednio w systemie operacyjnym robota, jest ono dostępne w całości i w najnowszej wersji przy każdym połączeniu.

5. Zawody robotów 2016

Poniżej krótko scharakteryzowano pierwszą publiczną demonstrację opisywanego robota, która odbyła się na Międzynarodowych Zawodach Robotów

ROBO~motion w dniu 23 kwietnia 2016 roku na Politechnice Rzeszowskiej. Robot został zaprezentowany w konkurencji *Free Style* pod nazwą „Kudłaty". Na potrzeby zawodów zostały zaprogramowane trzy tryby działania:

1. Sterowanie ręczne, gdzie ruchem sterowano bezpośrednio za pomocą klawiatury komputera, na którym uruchomiony był terminal graficzny. Tryb ten umożliwiał sterowanie szybkością robota (klawisze +,-), kierunkiem ruchu przód/tył (W, S), wykonywaniem skrętów lewo/prawo (A, D) oraz szybkich obrotów lewo/prawo w miejscu (Z, X).
2. Ucieczka od przeszkody, w której robot wykorzystywał dane z ultradźwiękowych czujników odległości. Jeżeli poniżej danej odległości (30 cm) od przedniego lub tylnego czujnika pojawiła się przeszkoda, robot oddalał się od niej na bezpieczną odległość (50 cm) i zatrzymywał. Jeśli przeszkoda pojawiła się w zasięgu czujnika bocznego, wówczas robot wykonywał obrót w taki sposób, aby przeszkoda znalazła się w zasięgu czujnika tylnego i odjeżdżał na bezpieczną odległość. Okazało się, że mimo tak prostego algorytmu robot potrafi znaleźć bezpieczny punkt nawet w dość skomplikowanym otoczeniu (np. stoliki, inne roboty, bandy reklamowe), a szukanie bezpiecznego punktu wyglądało atrakcyjnie dla publiczności.
3. Odtwarzanie sekwencji ruchu zapisanej w postaci prostego programu w pliku tekstowym. W tym celu opracowano prosty język skryptowy obsługujący deklaracje zmiennych, proste bezparametrowe procedury i bezwarunkowe pętle. Dzięki wykorzystaniu stosu, pętle i procedury mogły być zagnieżdżone. Każda elementarna instrukcja znajdowała się w osobnej linii. Program wykonywany był z góry na dół, wielkość znaków słów kluczowych musiała zostać zachowana, możliwe były wcięcia spacjami i puste linie. Program sekwencji ruchu składał się z trzech części: deklaracja zmiennych (VARIABLES … VARIABLES_END), deklaracja procedur (PROCEDURES … PROCEDURES) oraz program główny (PROGRAM … PROGRAM_END). W tabeli 1 przedstawiono słowa kluczowe i polecenia języka skryptowego, gdzie w nawiasach kwadratowych ([ ]) wpisano parametry podawane przez użytkownika nie będące częścią języka.

Tabela 1. Słowa kluczowe języka skryptowego

Table 1. Script language keywords

Słowo kluczowe	Opis
VARIABLES	Początek bloku deklaracji zmiennych
VARIABLES_END	Koniec bloku deklaracji zmiennych
PROCEDURES	Początek bloku deklaracji procedur
PROCEDURES_END	Koniec bloku deklaracji procedur
PROGRAM	Początek bloku programu („punkt wejścia”)
PROGRAM_END	Koniec programu
$[*nazwa*]	Deklaracja i użycie zmiennej, np. $krok = 100
proc [*nazwa*]	Początek procedury *nazwa*, np.: proc circle
end_proc	Koniec definicji procedury
call [*nazwa*]	Wywołanie procedury, np.: call circle
loop [N]	Początek bloku pętli, gdzie N jest liczbą powtórzeń. Parametr N może być liczbą wpisaną bezpośrednio lub zmienną zadeklarowaną wcześniej, np.: loop 20, loop $circ_count
end_loop	Koniec bloku pętli
set [L] [R] [T]	Polecenie elementarne ustawiające procent mocy nominalnej na kołach robota przez podany czas, gdzie: • L - moc na kole lewym w zakresie -100 do 100 [%] • R - moc na kole prawym w takim samym zakresie • T - minimalny czas wykonania polecenia [ms]
cor [L] [R] [T]	Polecenie elementarne wykonujące korektę aktualnej mocy na kołach robota przez podany czas, gdzie: • L - korekta mocy na kole lewym w zakresie -100 do 100 [%] • R - korekta mocy na kole prawym w takim samym zakresie • T - minimalny czas wykonania polecenia [ms]

Krótki program rozpędzający robota od 0 do 100% w czasie 10 sekund (100 x 100 milisekund), a następnie wprowadzający go w ruch obrotowy trwający 5 sekund z 50% mocy na kołach ma postać:

```
VARIABLES
  $l_circ=50
  $r_circ=-50
VARIABLES_END
PROCEDURES
  proc circle
    set $l_circ $r_circ 5000
  end_proc
PROCEDURES_END
```

```
PROGRAM
  set 0 0 0
  loop 100
    cor 1 1 100
  end_loop

  call circle
  set 0 0 0
PROGRAM_END
```

Cały kod interpretera skryptu zajmuje około 300 linii tekstu. Podczas publicznej prezentacji w trakcie zawodów ROBO~motion 2016 korzystając z programu napisanego w tym języku skryptowym robot wykonał symulację ruchów walca wiedeńskiego w takt klasycznego utworu *Fale Dunaju* Josefa Ivanovicia.

6. Podsumowanie

W pierwszej fazie prac autora nad algorytmami nawigacji w czasie rzeczywistym udało się wykonać prototyp pełnosprawnego minirobota oraz przygotowano i przetestowano zestaw narzędzi oraz technik pracy nad rozwojem jego funkcjonalności. Dotychczasowe algorytmy były realizowane bez sprzężeń zwrotnych i czujników innych niż ultradźwiękowe.

W artykule zaprezentowano minirobot laboratoryjny skonstruowany dla przyszłych badań. Przedstawiono jego budowę mechaniczną, komputer sterujący wraz z zestawem oprogramowania użytkowego, charakterystykę i technikę zdalnej pracy z robotem. Zostały przeprowadzone pierwsze testy w ruchu, robot został także zaprezentowany na Międzynarodowych Zawodach Robotów ROBO~motion 2016. Zbudowany minirobot wydaje się spełniać oczekiwania oraz być solidną platformą rozwojową dla dalszych badań.

W ciągu następnych kilku miesięcy od zawodów ROBO~motion 2016 przygotowano kolejne elementy oprogramowania użytkowego, takie jak:

- kompensacja tarcia statycznego
- utrzymanie stałego kursu geograficznego
- utrzymanie stałej odległości od poruszającego się „przewodnika”
- obrót tworzący mapę odległości od przeszkód w otoczeniu robota

Elementy te wykorzystują dane z ultradźwiękowych czujników odległości oraz z magnetometru. Do sterowania zastosowano typowe mechanizmy ze sprzężeniem zwrotnym (regulatory PI) oraz filtry cyfrowe. Badania nawigacyjne o krótkim zasięgu w czasie rzeczywistym będą wykonywane na prostokątnej arenie. Planuje się także dodanie obsługi akcelerometru oraz kamery, jednej na robocie, a drugiej zewnętrznej. Celem następnego etapu prac jest uzupełnianie oprogramowania o algorytmy nawigacyjne, testowanie cyfrowej filtracji sygnałów, dobór parametrów algorytmów oraz zastosowanie metod inteligencji obliczeniowej (sieci neuronowe) do lokalizacji robota.

Literatura

[1] Dudek D., Kazała R., Strączyński P.: Mobilny robot manipulacyjny wykorzystujący technologie Internetu Rzeczy w systemie sterowania i monitorowania, Pomiary Automatyka Robotyka, nr 4/2016, s. 37-45.

[2] Figurowski D., Brasel M., Kubicki M.: Stanowisko laboratoryjne do badań algorytmów sterowania robotami mobilnymi z wizyjnym sprzężeniem zwrotnym, Pomiary Automatyka Robotyka, nr 3/2016, s. 71-76.

[3] Kalisch M., Panfil W.: System sterowania grupą robotów ligi Small Size Robot League, Pomiary Automatyka Robotyka, nr 11/2014, s. 90-95.
[4] Leniowski R.: Podstawy robotyki, Uniwersytet Rzeszowski, Rzeszów 2013
[5] Węgierek M., Świstak B., Winiarski T.: Modularne środowisko do rywalizacji robotów sportowych śledzących linię, Pomiary Automatyka Robotyka, nr 3/2015, s. 61-66.
[6] http://eti.pg.edu.pl/katedra-systemow-decyzyjnych-i-robotyki/aktualnosci/-/asset_publisher/Km3ylDPmIIDZ/content/roboty-mobilne
[7] http://rab.ict.pwr.wroc.pl/lab_010/stanowiska/khepera.php
[8] http://raspberrypi.org
[9] http://wiringpi.com
[10] https://etcher.io/
[11] https://sourceforge.net/projects/win32diskimager/
[12] https://www.k-team.com/khepera-iv
[13] https://www.lego.com/pl-pl/mindstorms

PROTOTYPE THREE-WHEEL LAB MINIROBOT

S u m m a r y

The paper presents construction of a three-wheel lab minirobot used initially as a sensor testing platform, and in future for evaluation of intelligent navigation algorithms in real motion environment. Availability of elements, easy assembling, and low price have been construction guidelines. The robot is equipped with Raspberry Pi 2 microcomputer and Linux Raspbian operating system. Together with application programs, it operates both as a control system and development platform allowing to prepare and run programs directly in the robot. Communication is provided by terminal services, so easy programming the robot from any place with internet access is possible. The construction described here was presented at Rzeszow Robo~motion 2016 competition. Three operating modes were available at that time.

Keywords: Mobile robot, Raspberry Pi 2, ultrasonic sensors, IMU

DOI: 10.7862/re.2017.4

Tekst złożono w redakcji: marzec 2017
Przyjęto do druku: maj 2017