Laboratorium Podstaw Robotyki

Politechnika Poznańska
Katedra Sterowania i Inżynierii Systemów

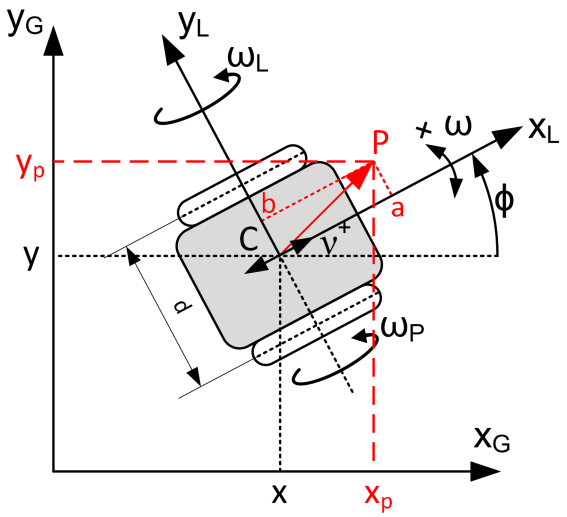
ĆWICZENIE 3

Kinematyka i lokalizacja dwukołowego robota mobilnego

Celem ćwiczenia jest wyprowadzenie modelu kinematyki dwukołowego robota mobilnego oraz poznanie budowy i zasady działania systemu pomiarowego odometrii inkrementalnej służącego do określania bieżącej pozycji i orientacji platformy robota w globalnym układzie współrzędnych.

1 Kinematyka dwukołowego robota mobilnego

Model kinematyki dowolnego systemu mechanicznego wynika z matematycznych relacji wiążących wszystkie prędkości występujące w rozważanym systemie. Uwaga nasza skupiona zostanie na prostym modelu robota dwukołowego przedstawionego na rys. 1. Na rysunku tym zaznaczono lokalny $\{x_L, y_L\}$ oraz globalny $\{x_G, y_G\}$ układ współrzędnych. Środek układu lokalnego przywiązano na osi kół i w połowie odległości pomiędzy kołami. Pozycję oraz orientację platformy wyrażoną w układzie globalnym oznaczono odpowiednio jako x, y oraz φ (zwrot strzałki oznacza dodatni przyrost kąta orientacji). Zaznaczono także prędkości kątowe kół: prawego ω_P i lewego ω_L oraz prędkości związane z całą platformą: prędkość postępową v początku układu lokalnego (znak + oznacza dodatni zwrot prędkości) oraz prędkość kątową ω (znak + oznacza dodatni kierunek obrotu platformy). Przy założeniu braku poślizgu kół (zarówno poślizgu wzdłużnego



Rysunek 1: Dwukołowy robot mobilny w globalnym układzie współrzędnych.

jak i poprzecznego), relacje wiążące prędkości kół z prędkościami postępową v i kątową ω całej platformy są następujące [1]:

$$v(t) = \frac{v_P(t) + v_L(t)}{2} = \frac{[\omega_P(t) + \omega_L(t)]R}{2}, \tag{1}$$

$$\omega(t) = \frac{v_P(t) - v_L(t)}{d} = \frac{[\omega_P(t) - \omega_L(t)]R}{d}, \tag{2}$$

gdzie v_L i v_P oznaczają liniowe prędkości związane z obwodem odpowiednio lewego i prawego koła, R jest długością promienia koła, a wartość d oznacza rozstaw między kołami (por. rys. 1). W oparciu o prostą interpretację geometryczną i korzystając z rysunku 1 można zapisać następujące relacje różniczkowe [2]:

$$\dot{\varphi} = \omega, \tag{3}$$

$$\dot{x} = v\cos\varphi, \tag{4}$$

$$\dot{y} = v\sin\varphi, \tag{5}$$

gdzie $\dot{x} \equiv v_x$ i $\dot{y} \equiv v_y$ oznaczają składowe prędkości postępowej platformy wyznaczone jako ortogonalne rzuty na poszczególne osie układu bazowego $\{x_G, y_G\}$.

1.1 Korzystając z zależności (3)-(5) wyznaczyć macierzowe równanie stanu robota dwukołowego przyjmując wektor stanu $\mathbf{q} \triangleq [\varphi\, x\, y]^T$ oraz sterowanie $\mathbf{u} \triangleq [\omega\ v]^T$.

1.2 Korzystając z zależności (1)-(5) wyznaczyć macierzowe równanie stanu robota dwukołowego przyjmując wektor stanu $\mathbf{q} \triangleq [\varphi\ x\ y]^T$ oraz sterowanie $\mathbf{u} \triangleq [\omega_P\ \omega_L]^T$.

Punkt P stanowi odsunięcie wskaźnika (punktu referencyjnego) od środka łączącego osie robota. W układzie lokalnym współrzędne punktu P wynoszą $P = [a, b]$. zaś w układzie zewnętrznym:

$$P = \begin{bmatrix} x \\ y \end{bmatrix} + R(\varphi) \begin{bmatrix} a \\ b \end{bmatrix}$$

gdzie: $R(\varphi) = \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix}$.

2 Lokalizacja – obliczanie pozycji i orientacji robota

Aby robot mógł bezpiecznie poruszać się w danym otoczeniu, musi on mieć możliwość określenia swojego położenia względem otaczających go przeszkód. Zwykle zadanie to realizowane jest poprzez budowę mapy otoczenia i okresowe wyznaczanie współrzędnych pozycji oraz orientacji robota w globalnym układzie współrzędnych (wyznaczanie lokalizacji robota). Do wyznaczenia położenia i orientacji konieczna jest znajomość modelu kinematyki platformy.

Powszechnie stosowaną techniką służącą do określania lokalizacji kołowych robotów mobilnych jest odometria, która polega na wyznaczaniu położenia i orientacji robota na podstawie pomiarów obrotu kół za pomocą przetworników obrotowo-impulsowych (enkoderów) sprzężonych mechanicznie z kołami robota [3]. Na podstawie ciągu impulsów wyznaczana jest zmiana pozycji kątowej każdego z kół w przyjętej jednostce czasu, pozwalając na obliczenie prędkości kół: ω_P, ω_L. Prędkości te z kolei pozwalają na obliczenie prędkości platformy robota zgodnie z zależnościami (1) oraz (2). Układ odometrii inkrementalnej oblicza bieżące wartości pozycji i orientacji platformy w następujący sposób:

$$\begin{aligned}
\varphi(t) &= \varphi(0) + \int_0^t \omega(\tau)d\tau = \varphi(0) + \int_0^t \frac{[\omega_P(\tau) - \omega_L(\tau)]\,R}{d}d\tau, \\
x(t) &= x(0) + \int_0^t v_x(\tau)d\tau = x(0) + \int_0^t \frac{[\omega_P(\tau) + \omega_L(\tau)]\,R}{2}\cos\varphi(\tau)d\tau, \\
y(t) &= y(0) + \int_0^t v_y(\tau)d\tau = y(0) + \int_0^t \frac{[\omega_P(\tau) + \omega_L(\tau)]\,R}{2}\sin\varphi(\tau)d\tau,
\end{aligned} \tag{6}$$

gdzie $x(0), y(0)$ i $\varphi(0)$ oznaczają wartości początkowe współrzędnych pozycji i orientacji platformy.

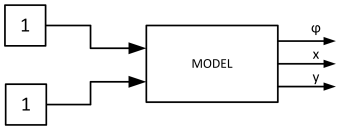
Należy pamiętać, iż odometria najczęściej jest implementowana w postaci dyskretnej i realizowana za pomocą układów cyfrowych. W związku z tym równania (3)-(5) należy przedstawić w postaci równań różnicowych stosując jedną ze znanych metod dyskretyzacji. Dla przykładu, równania (6) wynikające z dyskretyzacji modelu (3)-(5) metodą *Eulera wstecz* mają następującą postać [3]:

$$\begin{aligned} \varphi(n) &= \varphi(n-1) + T_p\,\omega(n), \\ x(n) &= x(n-1) + T_p\,v_x(n), \\ y(n) &= y(n-1) + T_p\,v_y(n), \end{aligned} \tag{7}$$

gdzie: $\omega(n) = \frac{[\omega_P(n)-\omega_L(n)]R}{d}$, $v_x(n) = \frac{[\omega_P(n)+\omega_L(n)]R}{2}\cos\varphi(n)$ oraz $v_y(n) = \frac{[\omega_P(n)+\omega_L(n)]R}{2}\sin\varphi(n)$, a $T_p > 0$ jest przyjętym okresem próbkowania.

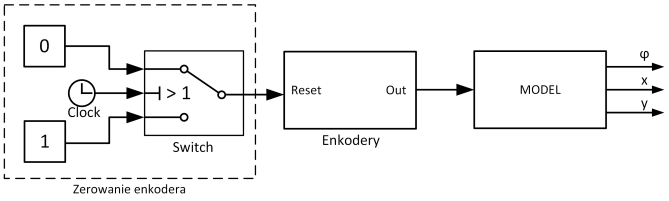
2.1 Konfiguracja środowiska (Windows XP)

2.1 W środowisku Matlab-Simulink utworzyć schemat blokowy realizujący obliczenia lokalizacji platformy w oparciu o sygnały prędkości kątowych ω_P oraz ω_L.
Obliczyć i uwzględnić w programie współrzędne punktu P, przyjmując $b = 0$
Sprawdzenie algorytmu należy wykonać dla stałych i sinusoidalnie zmiennych prędkości ω_P, ω_L (rys. 2).
Przed realizacją ćwiczenia należy utworzyć własny podkatalog w katalogu D:\Laboratorium\Robotyka\cw3, w którym należy zapisywać wyniki.



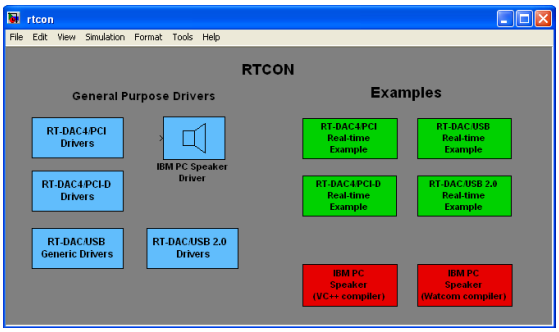
Rysunek 2: Schemat realizujący obliczenia lokalizacji platformy robota w oparciu o sygnały zadane przez użytkownika

2.2 Powyższy schemat blokowy zmodyfikować tak, by sygnały prędkości kół robota ω_P oraz ω_L były rzeczywistymi sygnałami wózka dwukołowego połączonego z kartą RT-DAC4/PCI D (rys. 3).



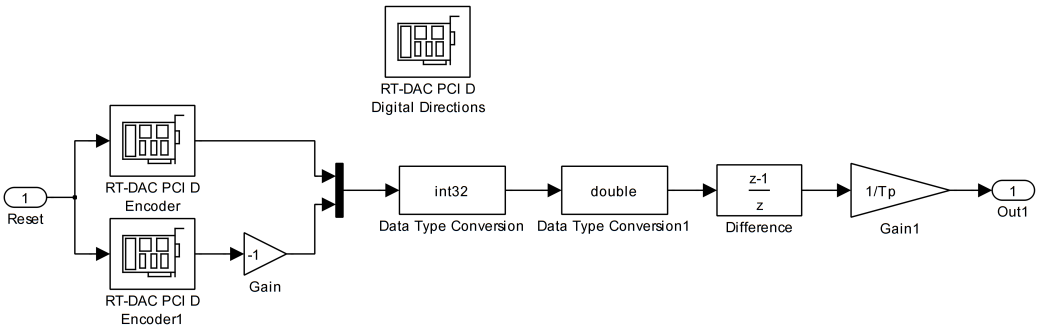
Rysunek 3: Schemat realizujący obliczenia lokalizacji platformy robota w oparciu o sygnały z enkoderów

Aby powyższy schemat blokowy przygotować do współpracy z kartą RT, należy skorzystać z biblioteki Simulink'a *RTCON* (rys. 4).
Uruchomienie biblioteki następuje poprzez wpisanie w *Command Window* programu *MATLAB* polecenia **rtcon**.
Wybór odpowiednich funkcji następuje po dwukrotnym kliknięciu i otwarciu bloku sterowników z grupy *General Purpose Drivers*.



Rysunek 4: Widok głównego okna pakietu RT-CON

Sposób realizacji bloku *Enkodery* z rysunku 3 przedstawia rysunek 5 .



Rysunek 5: Schemat realizacji odczytu z enkoderów w czasie rzeczywistym

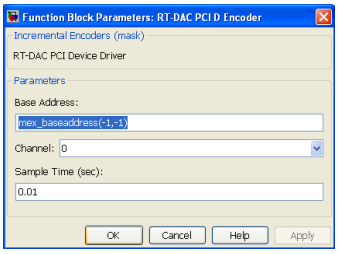
W zmodyfikowanym schemacie blokowym sygnały prędkości kątowych ω_P, ω_L należy skojarzyć z kartą RT poprzez blok *RT-DAC PCI D Encoder*, który jest programowym licznikiem impulsów zbierającym impulsy z dwóch linii przetworników dla koła prawego i lewego. Na wyjściach tego bloku otrzymuje się zarejestrowaną w jednostce czasu $T_p = const$ liczbę impulsów enkoderów obu kół platformy. Wartość tego czasu oraz liczba impulsów stanowią podstawę do wyznaczenia chwilowych prędkości kół robota. Do obliczenia prędkości ω_L i ω_P konieczne jest przeskalowanie otrzymanych wartości wyjściowych bloku *RT-DAC PCI D Encoder* stosując następujący współczynnik skalujący (na wyjściu z bloku *Enkodery*):

$$\mathbf{K} = \frac{2\pi}{N}, \tag{8}$$

gdzie $\mathbf{N} = 648$ oznacza liczbę impulsów przetwornika przypadającą na jeden obrót koła platformy mobilnej, a $\mathbf{T_p} = 0.01[s]$ jest przyjętym (stałym) okresem próbkowania. Wartości parametrów R i d występujące w równaniach (6) muszą odpowiadać geometrycznym wymiarom rzeczywistej platformy mobilnej:

$$\mathbf{R} = 0.021[m], \qquad \mathbf{d} = 0.073[m]. \tag{9}$$

2.3 W celu poprawnego odczytu impulsów z enkoderów należy określić w bloku *RT-DAC PCI D Encoder* jego adres oraz kanał. Odczyt następuje na kanale '0' dla lewego koła, oraz '1' dla koła prawego.
Konfigurację enkodera dla kanałów '0' oraz '1' przedstawia rysunek 6, gdzie w miejsce *Base Address* należy wpisać wartość **57088** będącą adresem konkretnego wejścia z enkodera na karcie RT.



Rysunek 6: Konfiguracja enkodera

2.4 W oknie, w którym tworzymy schemat blokowy dokonać ustawień następujących parametrów symulacji:
Simulation → Configuration Parameters → Solver:

- Stop time→ 9999999,
- Type-FixedStep→ ode1,
- FixedStepSize→ 0.01,
- Tasking Mode→ SingleTasking.

Simulation → Configuration Parameters → Code Generation:

- System target file→ rtcon.tlc (Browse...),
- Language→ C,
- Description→ RT-CON (Visual C/C++).

Simulation → Configuration Parameters → Code Generation → Interface:

- Data exchange → Interface: External Mode,
- Host/Target interface → Transport layer: RT-CON tcpip.

Simulation → Configuration Parameters → Code Generation → RT-CON options:

- Display start-up message → TAK,
- Append RT-DAC/USB drivers → NIE,
- Append RT-DAC/PCI drivers → TAK.

2.5 Po ustawieniu parametrów symulacji uruchomić kompilację modelu :
Simulation → Configuration Parameters → Code Generation → BUILD:

2.6 W wyniku poprawnego procesu kompilacji w oknie poleceń MATLABa zostaną wyświetlone następujące komunikaty:
`### Created RT-CON executable: NAZWA_PLIKU.dll`
`### Successful completion of build procedure for model: NAZWA_MODELU`
Aby uruchomić tryb pracy *real-time* należy ustawić `w głównym oknie symulacji` tryb pracy `External` (zamiast *Normal*), a nastepnie uruchomić *Simulation → Connect to target*.
Zatwierdzić *OK* w oknie informacyjnym kompilacji oraz uruchomić tryb *Start real-time code*.

3 Przebieg ćwiczenia

3.1 Przyjąć punkt centralny kątomierza jako punkt P (rys. 7), o następujących parametrach, $P = (a, b) = (0, 35)[mm]$.

3.2 Mając do dyspozycji linijkę/papier milimetrowy, zmierzyć przebytą przez robota odległość, tzn. dokonać przejazdu wzdłuż jednej z krawędzi stołu na odległość 30 cm. W trakcie wykonywania pomiarów nie podnosić robota ze stołu.

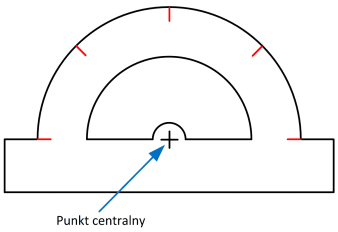
3.3 Porównać powyższą odległość z odległością wynikającą z pomiarów za pomocą odometrii (wyniki w programie *Simulink*).

3.4 Wykonać kilkukrotnie przejazd na odległość 30 cm (w przód, następnie w tył). Po powrocie do punktu początkowego odczytać wartość położenia w programie *Simulink*. W trakcie wykonywania pomiarów nie podnosić robota ze stołu.

3.5 Ustawić robota na środku planszy.

3.6 Obrócić robota o podany przez prowadzącego kąt (np. 47°).

3.7 Porównać wartość wskazaną na kątomierzu z wartością odczytaną z pomiarów (*Simulink*).



Rysunek 7: Punkt centralny kątomierza

Literatura

[1] K. Kozłowski and J. Majchrzak, "Nowe algorytmy sterowania nieholonomicznym robotem mobilnym," in XIV Krajowa Konferencja Automatyki, vol. 2, (Zielona Góra), pp. 625–632, 2002.

[2] K. Tchoń, A. Mazur, I. Dulęba, R. Hossa, and R. Muszyński, Manipulatory i roboty mobilne. Modele, planowanie ruchu, sterowanie. Warszawa: Akademicka Oficyna Wydawnicza PLJ, 2000.

[3] T. Jedwabny, M. Kowalski, J. Majchrzak, and G. Wiczyński, "Przykład wielosensorycznego systemu pozycjonowania nieholonomicznego robota mobilnego," in XIV Krajowa Konferencja Automatyki, vol. 2, (Zielona Góra), pp. 641–646, 2002.