Achizitii de Date
Laboratorul 2: Roboti cu *Drive* Diferential

Dan Novischi

1. Introducere

Scopul acestui laborator este de a implemnta modelul cinematic al unui robot diferential utilizand datele provinite de la senzori de tip *quadrature encoder* si regulatorul *PID* in vederea determinarii starii si controlului acestuia la orice moment de timp.

2. Model Diferential

Robotul cu *drive* diferential, prezentat schematic in Figura 1, este un robot mobil a carui miscare este bazata pe actuarea independenta a doua roti montate simetric de o parte si de alta a unui sasiu. Astfel, prin variatia independenta vitezelor de rotatie (la ax) a celor doua roti se poate genera miscarea dorita a robotului, nefiind necesara utilizarea unui mecanism aditional pentru controlul directiei. Uzual, implementarea fizica a acestui robot se realizeaza prin utlizarea a doua motoare electrice (de curent continuu) ce pot fi controlate independent prin tehnica de PWM studiata anterior.

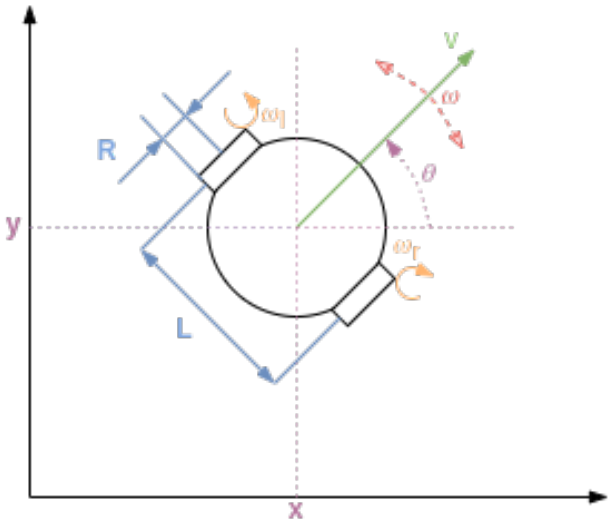

Figura 1: Modelul Diferential al Robotului

La nivelul inalt, miscarea robotului este descrisa de perechea de parametrii (v, ω) unde:

- v – reprezinta viteza liniara de deplasare a robotului.

- ω – reprezinta viteza angulara de rotatie a robotului.

in timp ce starea robotului este descrisa de perechea de parametrii (x, y, θ) denumita pose sau odometrie, unde:

- (x, y) – reprezinta coordonatele robotului fata de o referinta globala in 2D si sunt exprimate (de obicei) fata de punctul care reprezinta centrul de greutate al robotului.
- θ – reprezinta orientarea robotului fata de acelasi referinta globala. Prin conventie, acest unghi creste in sens trigonometric fata de axa X si scade in sensul acelor de cesornic fata de aceeasi axa.

Astfel, ecuatiile de miscare la acest nivel exprima felul in care se modifica starea robotului in timpul deplasarii si poarta numele de **model uniciclu**:

$$\hat{x} = v\cos(\theta)$$

$$\hat{y} = v\sin(\theta)$$

$$\hat{\theta} = \omega$$

Transfelul de la modelul unicilu la cel diferential (fizic) depinde de parametrii constructivi ai robotului, anume:

- R – raza rotiilor cuplate la axurile celor doua motoare (stang si drept).
- L – lungimea axei (imaginare) dintre roti a sasiului.

Prin intermediul acestor parametrii viteza de rotatie la ax a motorului drept ω_r si respectiv viteza la ax a celui stang ω_l se exrpima in functie de perechea (v, ω) astfel:

$$\omega_r = \frac{2v + \omega L}{2R} \ [rad/s]_{SI}$$

$$\omega_l = \frac{2v - \omega L}{2R} \ [rad/s]_{SI}$$

Tot aici, starea robotului la un anumit moment de timp se poate calcula pe baza deplasamentelor realizate de roata dreapta D_r si respectiv cea stanga D_l, anume:

$$x = x + \frac{D_r + D_l}{2}\cos(\theta) \ [m]_{SI}$$

$$y = y + \frac{D_r + D_l}{2}\sin(\theta) \ [m]_{SI}$$

$$\theta = \theta + \frac{D_r - D_l}{L} \ [rad]_{SI}$$

unde deplasamentul unei roti se determina pe baza rotatiei efectuate de acesta intre doua momente succesive de timp:

$$D = 2\pi R \cdot \Delta\texttt{rotatie}\,[m]_{SI}$$

3. Encoder in Cadratura

Un encoder in cadratura (en. quadrature encoder) este un senzor incremental cu ajutorul caruia se poate masura rotatia efectuata de axul unui motor intr-o anumita directie. Acesta are doua canale de iesire (A si B) care furnizeaza un numar de pulsuri egal distantate pe o rotatie a axului – vezi Figura 2. Pulsurile generate de cele doua canale sunt (constructiv) defazate intre ele la 90°. In functie de ordinea de detectie a pulsurilor – (A, B) sau (B, A) – se determina directia de rotatie a axului (in sensul acelor de ceas – clocwise CW sau in sens trigonometric – counter clock wise CCW).

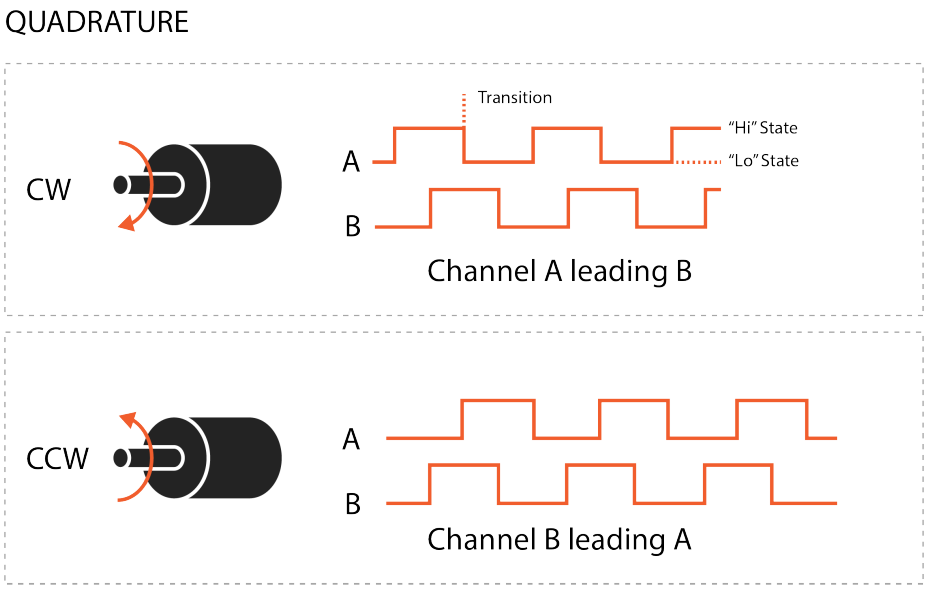

Figura 2: Encoderul in Cadratura.

Astfel, rotatia efectuata de axul unui motor, la care este cuplat un encoder in cadratura, se masoara prin incrementarea sau decrementarea numarului de pulsuri in functie de directia de rotatie. Mai exact, se masoara numarul de treceri aferente pulsurilor de la pozitiv la negativ si de la negativ la pozitiv aferent celor doua canale in functie de directia de rotatie. Acest numar de treceri poarta denumirea generica de *counts*, iar prin diferenta intre doua astfel de numere la momente succesive de timp obtinem deplasemntul (in counts) a axului intre aceste momente:

$$\Delta\texttt{counts} = \texttt{count}_k - \texttt{count}_{k-1}\ \ [\texttt{counts}]$$

Raportand aceasta diferenta la numarul total de *counts* N pe care encoderul il poate inregistra intr-o rotatie completa, obtinem numarul de rotatii (sau rotatia partiala) a axului intre cele doua momente succesive de timp:

$$\Delta\texttt{rotatie} = \frac{\Delta\texttt{counts}}{N} \ [\texttt{rotatii}]$$

Stiind numarul de rotatii la ax putem pe de-o parte calcula starea robotului, iar pe de alta putem raporta acest numar la unitatea de timp pentru a estima viteza de rotatie a motorului in $[rad/s]_{SI}$ sau in `rpm` (rotatii pe minut).

4. Cerinte

Sarciniile din cadrul acestui laborator umaresc cuplarea modelului uniciclu cu bucle de control aferente celor doua motoare astfel incat (vezi Figura 3): pe de-o parte robotul sa fie capabil sa execute orice comanda (v, ω), iar pe de alta parte starea (x, y, θ) acestuia sa fie determinata la orice moment de timp.

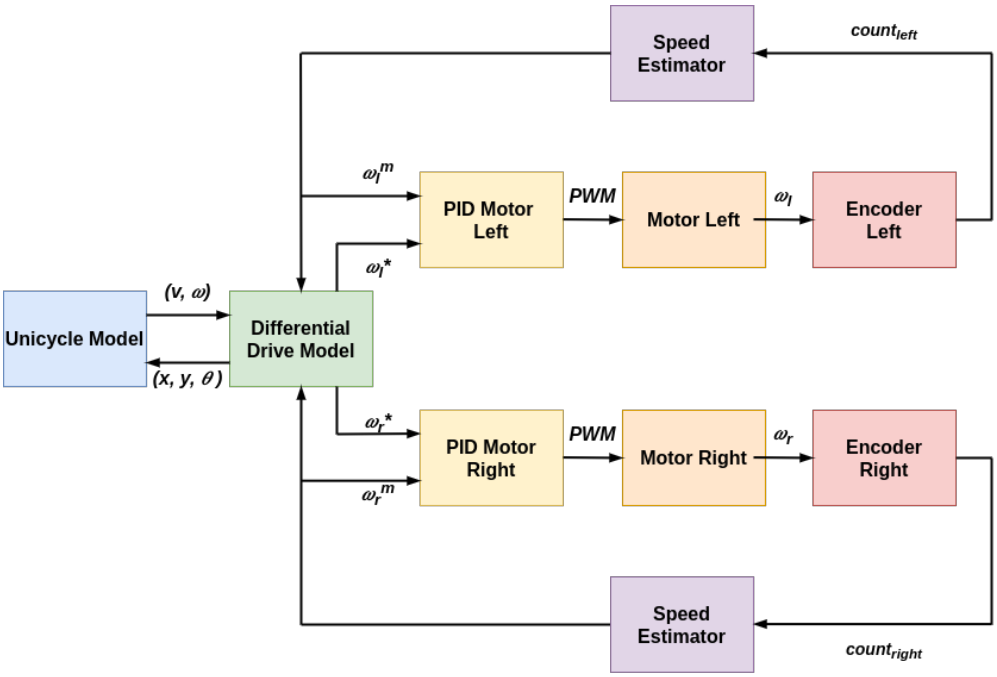

Figura 3: Bucla de control a unui Robot Diferential.

Aveti la dispozitie diagrama din TinkerCAD a carui circuit este deja legat pentru cele doua motoare. Scheletul de cod asociat este divizat in mai multe sectiuni (prin comentarii multi-line) dupa cum urmeaza:

- *Constant & Macro Definitions* – aici gasiti definite constantele/macro-urile utilizate in program; (ex: `#define ENC_MAX_COUNT 2064` care reprezinta numarul total N de counts furnizati de encoder intr-o rotatie).

- *Application Type Definitions* – aici gasiti toate definitiile de tip utilizate in aplicatie; (ex: `typedef struct Robot{...};`).
- *Application Function Definitions* – aici veti gasi toate definitiile de functii pe care le aveti de implementat si cele care au fost deja implementate.

  (ex: `void initRobot(struct Robot* robot, float r, float l){...}`)
- *Application Global Variables* – aici veti gasi toate declaratiile de variabile globale utilizate in program. (ex: `Robot robot;`)
- *Main application* – aplicatia principala (impartita intre `setup();` si `loop();`) care contine initializarea si aplicatia propriu-zisa ce trebuie implementata.

Cerinta 1 Utilizand TinckerCAD familirizati-va cu circuitul si scheletul de cod pentru acest laborator: https://www.tinkercad.com/things/7HMsEsmSYHw. Unde este cazul solicitati clarificari laborantului.

Cerinta 2 Implementati functiile asociate tipului `Robot` dupa cum urmeaza:

a) `void initRobot(struct Robot* robot, float r, float l)` intializeaza un obiect de tip `Robot`.

b) `void setRobotCommand(struct Robot* robot, float v, float omega)` calculeaza vitezele de rotatie pentru motorul stang si drept al robotului pe baza unei comenzi (v, ω) primite ca parametru.

c) `void updateRobotPose(struct Robot* robot, ...)` actualizeaza starea robotului pe baza rotatiilor $\Delta\texttt{rotatie}_{\texttt{left}}$ si $\Delta\texttt{rotatie}_{\texttt{right}}$ efectuate de cele doua motoare intre doua momente de timp succesive.

Cerinta 3 Implementati functiile asociate tipului `SpeedEstimator` dupa cum urmeaza:

a) `void initSpeedEstimator(struct SpeedEstimator* est, struct Encoder *enc)` intializeaza un obiect de tip `SpeedEstimator`.

b) `float estimateSpeed(struct SpeedEstimator* est, int direction)` estimeaza numarul de rotatii (Δ`rotatii`) efectuat intre doua monete succesive de timp.

c) `float get_rpm(float delta_rotation, long update_freq)` calculeaza viteza la axul unui motor in rotatii pe minut – RPM.

d) `float get_rad_per_second(float delta_rotation, long update_freq)` calculeaza viteza la axul unui motor in rad/s.

Cerinta 4 Implementatii aplicatia principala in functia `loop` urmarind TODO-urile din schelet si cerintele de mai jos:

a) Setati comanda ($v = 0.5\ m/s,\ \omega = 0\ rad/s$) si determinati directia de rotatie a fiecarui motor pe baza unei comenzi.

b) In bucla de control a motoarelor initializati referintele aferente celor doua regulatoare PID.

c) Calculati Δ`rotatii` pentru motorul stang si drept aferent unei iteratii a buclei de control.

d) Setati intrarea pentru regulatoarele PID stang si drept, apoi obtineti iesirele celor doua regulatoare.

e) Utilizati iesirile regulatoarelor PID pentru a genera semnalele PWM aferente celor doua motoare.

f) Actualizati poese-ul robotului.

Nota: Ce comanda se poate furniza robotului astfel incat traiectoria acestuia sa descrie un cerc cu raza de 1 m ?