УПРАВЛЕНИЕ НА МАНИПУЛАТОР С 3 СТЕПЕНИ НА СВОБОДА С RASPBERRY PI И CODESYS

Станимир Йорданов[1], Веселин Митев [2] , Теодор Ковачев[3]
[123]*Технически Университет Габрово*

CONTROL OF A MANIPULATOR WITH 3 DEGREES OF FREEDOM WITH RASPBERRY PI AND CODESYS

Stanimir Yordanov[1], Veselin Mitev[2] , Teodor Kovachev[3]
[1 2 3] *Technical University of Gabrovo*

Abstract

The given article covers the general formulations forward and inverse kinematic problems for robot motion control systems. We have discussed the difficulties how to solve such problems using analytical and numerical methods. The developed algorithms have been tested in the management of a laboratory model of a robotic arm with 3 axles in freedom.

Keywords: realtime control, raspberry pi, CoDeSys, robot arm.

ВЪВЕДЕНИЕ

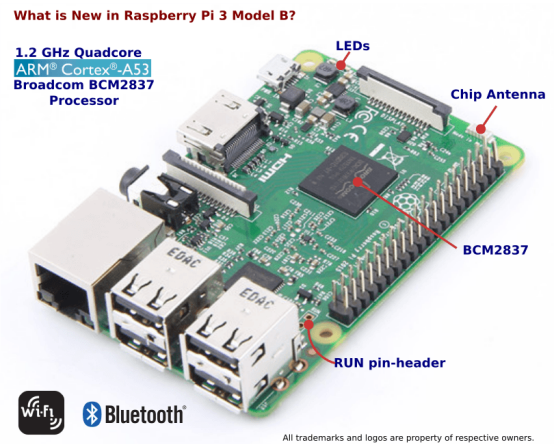
Роботизираните системи намират все по-голямо приложение в машиностроенето, автомобилостроенето и други отрасли на индустрията. Те заместват човека в все по-тежки, опасни или повторяеми операции. Намират приложение в дейности като заваряване, асемблиране, боядисване, обслужване на машини и др.

Работата с по-висока скорост и по-високите съотношения между полезния товар и теглото изискваха по-добро разбиране на сложната, взаимосвързана нелинейна динамика на роботите. Тази динамика в развитието на робототехниката изисква търсенето на нови методи и подходи от теорията на нелинейните и адаптивни системи за управление на сервозадвижванията в роботите. Един от тези подходи е управлението на базата на модели, включително компенсацията на кинематичната грешка, оптималното управление на сервосистемите и захранването. В настоящата статия се разглежда един подход за управление на робот-манипулатор с помощта на PLC изградено на базата на вградена система Raspberry Pi и програмирано с CoDeSys.

ИЗЛОЖЕНИЕ

Raspberry Pi [4] е едночипов компютър разработен във Великобритания за популяризиране на обучението по компютърни науки. Той се състои от процесор, графичен процесор и RAM памет. Тя е различна за различните модели като започва от 256MB за първите версии и достига до 1GB за последния модел Raspberry Pi 3 Model B. Последната версия е изградена с процесор Broadcom BCM2837 с четири ядрено ARM Cortex A53 (ARMv8) ядро. Тактовата честота е 1.2 GHz и RAM памет от 1GB. Мрежовият интерфейс се състои от 10/100 MBit/s адаптер както и с вграден 802.11n wireless модул.

Чрез софтуерният продукт CODESYS Control for Raspberry Pi SL‘, RaspberryPi се превръща в програмируем логически контролер (PLC).



Фиг. 1. *Едноплатков компютър Raspberry Pi 3*

Controller Development System – CoDeSys

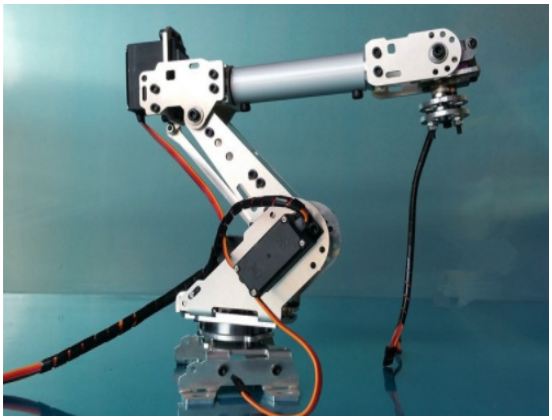
CoDeSys е софтуерна платформа за разработване на приложения за програмируеми логически контролери (PLC). Системата е базирана на международен индустриален стандарт IEC 61131-3. Поддържат се всичките езици за програмиране дефинирани в стандарта.

За програмиране на Raspberry Pi чрез CoDeSys е необходимо инсталиране на софтуерен пакет CODESYS Control for Raspberry Pi SL [5].

Обект за управление

Като обект за управление е избрана роботизирана ръка с 3 степени на свобода. Тя се задвижва от 4 серво двигателя MG996R за управление на основа, рамо. Серводвигателите, използвани в макета представляват постояннотокови двигатели със свързан потенциометър в изходния вал. Така изходната позиция постоянно се сравнява със зададената от контролера за управление. Всяка разлика между двете стойности поражда грешка в съответната посока, която кара серводвигателя да коригира позицията си. MG996R е стандартно сервозадвижване с висок въртящ момент и с ъгъл на завъртане от 120° (по 60° във всяка посока). Захранващото напрежение на двигателя е от 4,8-7,2 VDC. Управлението на позицията се извършва чрез широчинно-импулсна модулация.

Към двигателя се подават сигнали с честота 50Hz (20ms период). В зависимост от ширината на импулса (Duty cycle) двигателя променя ъгъла си на завъртане.



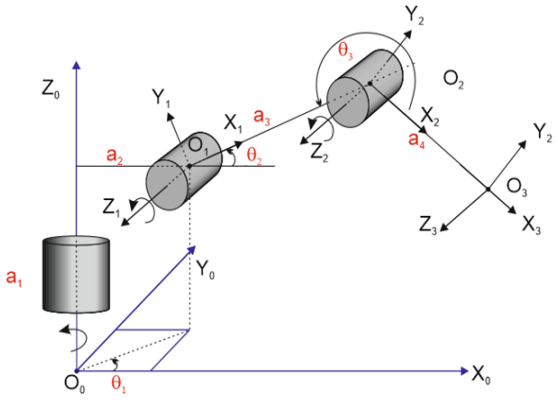
Фиг. 2. *Манипулатор с три степени на свобода*

Пълният обхват на завъртане от -60° до 60° се получава при ширина на импулса от 1ms до 2ms. Така при 1ms има завъртане на -60°, при 1,5ms оста се завърта на 0°, а при 2ms на 60°.

Фиг. 3. *серво двигател MG996R*

Изработването на импулсите за управление на всеки двигател се осъществява от драйвер на фирмата Adafruit. Той представлява 16-канален 12-битов I2C – PWM контролер, базиран на LED Controller PCA9685.

Кинематика на обекта



Фиг. 4. *Кинематична схема на манипулатора*

На фигура 4 е показана кинематичната схема на манипулатор с три степени на сво-

бода. На схемата с 0, 1 и 2 са обозначени базовата, раменната и лакетната става. Инструментът се намира на фрейм с номер 3. Трите звена имат ъгъл на завъртане съответно θ_1, θ_2 , θ_3. За управлението на една роботизирана ръка, от гледна точка на кинематиката, трябва да се решат две основни задачи, описващи движението:

- При зададени ъгли на завъртане на звената на манипулатора (основа, рамо, лакът), да се намери положението на инструмента в пространството – *Кинематика в права посока*.

- При зададена позиция на инструмента (X,Y,Z), да се намерят съответните ъгли на завъртане на звената на манипулатора – *Кинематика в обратна посока*.

Кинематика в права посока

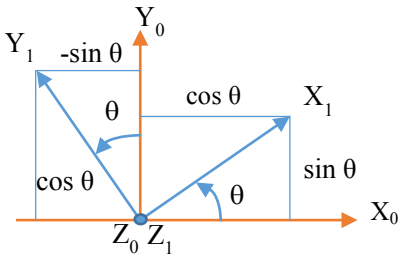
За да се намери позицията на инструмента в пространството е необходимо съставяне на математически модел описващ движението [1,2,3]. Той се представя с матрица, описваща ротацията и преместването на звената, наречена **хомогенна трансформираща матрица.**

$$H_n^m = \begin{bmatrix} R_n^m & d_n^m \\ 0 & 1 \end{bmatrix} \qquad (1)$$

Където: R_n^m е ротационна матрица - даваща информация как едното звено е завъртяно спрямо предходното, а d_n^m е вектор на отместване – показващ, отместването на звено спрямо предходното звено.

Ротационна матрица

В роботиката ротационната матрица дава представата как една координатна система (фрейм) е завъртяна спрямо друга. Тя представлява квадратна матрица с 3 реда и три колони (3x3). Всяка колона е проекцията на съответната ѝ ос, спрямо предходния фрейм.



***Фиг. 5**. Графично разположение на два фрейма*

$$R_1^0 = \begin{matrix} \\ x_0 \\ y_0 \\ z_0 \end{matrix} \begin{matrix} x_1 & y_1 & z_1 \\ \lceil\blacksquare & \blacksquare & \blacksquare\rceil \\ \lvert\blacksquare & \blacksquare & \blacksquare\rvert \\ \lfloor\blacksquare & \blacksquare & \blacksquare\rfloor \end{matrix} \qquad (2)$$

На фигура 5 са представени два фрейма, завъртени един спрямо друг на ъгъл θ. За да се попълни матрицата се приема, че дължината на всяка ос е равна на 1. Така всяка проекция може да се определи с тригонометрични функции. Проекцията на X_1 спрямо X_0 е cos θ. Проекцията на X_1 спрямо Y_0 е sin θ. X_1 е перпендикулярна на Z_0. Това означава, че проекцията и е равна на нула.

$$R_1^0 = \begin{matrix} \\ x_0 \\ y_0 \\ z_0 \end{matrix} \begin{matrix} x_1 & y_1 & z_1 \\ \lceil\cos\theta & \blacksquare & \blacksquare\rceil \\ \lvert\sin\theta & \blacksquare & \blacksquare\rvert \\ \lfloor\ 0 & \blacksquare & \blacksquare\rfloor \end{matrix} \qquad (3)$$

Y_1 също е завъртян с ъгъл θ спрямо Y_0. Така проекцията на Y_1 спрямо X_0 е $\sin\theta$, но тъй като посоката на завъртане е отрицателна, се получава $-\sin\theta$. Проекцията на Y_1 спрямо Y0 е cos θ. Y_1 е перпендикулярна на Z0, проекцията ѝ е 0.

$$\begin{matrix} \\ x_0 \\ y_0 \\ z_0 \end{matrix} \begin{matrix} x_1 & y_1 & z_1 \\ \lceil\cos\theta & -\sin\theta & \blacksquare\rceil \\ \lvert\sin\theta & \cos\theta & \blacksquare\rvert \\ \lfloor\ 0 & 0 & \blacksquare\rfloor \end{matrix} \qquad (4)$$

За последната колона се спазва същото правило. Z_1 е перпендикулярна на X_0 и на Y_0, за това проекцията и е нула. Z_1 е успоредна на Z_0, нейната проекция е пълната дължина на оста Z_1 т.е. 1.

$$\begin{matrix} \\ x_0 \\ y_0 \\ z_0 \end{matrix} \begin{matrix} x_1 & y_1 & z_1 \\ \lceil\cos\theta & -\sin\theta & 0\rceil \\ \lvert\sin\theta & \cos\theta & 0\rvert \\ \lfloor\ 0 & 0 & 1\rfloor \end{matrix} \qquad (5)$$

Получената матрица R_1^0 показва, че всички оси се завъртат на ъгъл θ, спрямо оста Z. Така могат да се съставят три елементарни ротационни матрици, при ротация само по една ос:

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix},$$

$$R_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix},$$

$$R_z = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

При ротация на повече от една ос, трансформацията се опростява до завъртане по основните оси. Съставят се съответните елементарни ротационни матрици. Общата ротационната матрица се получава, като се умножат съответните матрици.

$$R_1^0 = R_x * R_y * R_z \quad (6)$$

За да се намери ротационната матрица на цялата система т.е. да се намери матрицата от фрейма на инструмента към базовият фрейм е необходимо да се намерят ротационните матрици на всички звена. Общата ротационна матрица, показваща как последното звено е завъртяно спрямо базовото, се намира като се умножат ротационните матрици на всички звена, започващи от базовото и завършващи с инструмента.

$$R_3^0 = R_1^0 R_2^1 R_3^2 \quad (7)$$

Важен момент при определяне на матриците е определяне ориентацията на всички оси. Има 4 правила, по които се извършва това:

Правило 1: Оста Z е оста на въртенето във въртящо се звено или посоката на движението в движещо се звено.

Правило 2: Оста X трябва да е перпендикулярна на оста Z на текущото звено и перпендикулярна на Z оста на предходното звено.

Правило 3: Всеки фрейм трябва да спазва правилото на дясната ръка както следва: с изпъната дясна ръка и пръсти, сочещи по оста X, палец, сочещ по оста Z, а дланите сочат оста Y.

Правило 4: Всички X оси трябва да пресичат Z осите на предходния фрейм.

Общата ротационната матрица на звеното е съставена от две части. Първата част показва каква е позицията на звеното, спрямо предходното при ъгъл на завъртане 0°. Втората част е съответната ротационна матрица Rx, Ry или Rz.

За трите звена се получава:

$$R_1^0 = R_{Z_0} * R_0^1 0 = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} \cos\theta_1 & 0 & \sin\theta_1 \\ \sin\theta_1 & 0 & -\cos\theta_1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$R_1^2 = R_{Z_1} * R_1^2 0 = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 \\ \sin\theta_2 & \cos\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 \\ \sin\theta_2 & \cos\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (8)$$

$$R_2^3 = R_{Z_2} * R_2^3 0 = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 \\ \sin\theta_3 & \cos\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 \\ \sin\theta_3 & \cos\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Вектор на изместване (Displacement Vector)

Ротационната матрица показва как е завъртян инструмента на манипулатора, спрямо координатната система на базовото звено. Но в движението на инструмента освен ротацията, той променя и своята позиция. Ротационната матрица не дава информация как се променя тя. Тази задача се изпълнява от вектора на изместване d и представлява матрица с 3 реда и една колона.

$$\mathrm{d}_\mathrm{n}^\mathrm{m} = \begin{bmatrix} \mathrm{X}_\mathrm{n}^\mathrm{m} \\ \mathrm{Y}_\mathrm{n}^\mathrm{m} \\ \mathrm{Z}_\mathrm{n}^\mathrm{m} \end{bmatrix} \quad (9)$$

Първият ред на матрицата показва каква е относителната позиция на звено m към звено n по оста X, втория ред – по оста Y и третия ре – по оста Z. Векторите на изместване на изследвания обект са:

$$\mathrm{d}_1^0 = \begin{bmatrix} \mathrm{a}_2 * \cos\theta_1 \\ \mathrm{a}_2 * \sin\theta_1 \\ \mathrm{a}_1 \end{bmatrix} \quad (10)$$

$$\mathrm{d}_2^1 = \begin{bmatrix} \mathrm{a}_3 * \cos\theta_2 \\ \mathrm{a}_3 * \sin\theta_2 \\ 0 \end{bmatrix} \quad (11)$$

$$d_3^2 = \begin{bmatrix} a_4 * \cos\theta_3 \\ a_4 * \sin\theta_3 \\ 0 \end{bmatrix} \quad (12)$$

Homogeneous Transformation Matrix

За всяко звено, след базовото, се съставя хомогенна трансформираща матрица (1) даваща информация за положението на звеното спрямо предходното. Така за манипулатор с три оси на свобода (три звена) се получават три отделни матрици.

$$H_1^0 = \begin{bmatrix} R_1^0 & d_1^0 \\ 0 & 1 \end{bmatrix}, H_2^1 = \begin{bmatrix} R_2^1 & d_2^1 \\ 0 & 1 \end{bmatrix}, H_3^2 = \begin{bmatrix} R_3^2 & d_3^2 \\ 0 & 1 \end{bmatrix}$$

За възел 1 матрицата се получава:

Ротационна матрица Вектор на преместване

$$H_1^0 = \begin{bmatrix} R_1^0 & d_1^0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_1 & 0 & \sin\theta_1 & a_2 * \cos\alpha\theta_1 \\ \sin\theta_1 & 0 & -\cos\theta_1 & a_2 * \sin\theta_1 \\ 0 & 1 & 0 & a_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (13)$$

За възел 2

$$H_2^1 = \begin{bmatrix} R_2^1 & d_2^1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & a_3 * \cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & a_3 * \sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (14)$$

За възел 3

$$H_3^2 = \begin{bmatrix} R_3^2 & d_3^2 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & a_4 * \cos\theta_3 \\ \sin\theta_3 & \cos\theta_3 & 0 & a_4 * \sin\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (15)$$

Общата матрица се получава при умножение на матриците на всички звена. Получава се матрица с четири реда и четири стълба, даваща информация за положението на инструмента (последното звено) спрямо базовата координатна система.

$$H_3^0 = H_1^0 * H_2^1 * H_3^2 \quad (16)$$

След умножението на трите матрици се получава квадратна матрица с 4 реда и 4 колони. Позицията на инструмента се намира на 4 колона, както следва:

$$X=H_0^3\ [1,4],\ Y=H_0^3\ [2,4],\ Z=H_0^3\ [3,4] \quad (17)$$

Програмна реализация на правата задача

Поради повишения брой математически уравнения с помощта, на които се управляват серво двигателите, задвижващи звената на роботизираната ръка, е избран език за програмиране Structured text. Той позволява лесна работа със сложни математически функции.

Кинематиката в права посока е решена по два метода. За намиране на позицията на инструмента е необходимо съставянето на хомогенни трансформиращи матрици за трите звена.

Функцията *ForwardKinematic_V1* приема като параметър структура от моментното състояние на ъглите, сключени между звената на манипулатора и връща координатите на инструмента в координатната система на базата. Като променливи за функцията се декларират трите ротационни матрици (R01, R02, R03), трита вектора на изместване (d01, d02, d03) и трите хомогенни трансформиращи матрици (H01, H02, H03). Ротационните матрици се декларират като двумерен масив с размер 3x3. Вектора на изместване се декларира като двумерен масив от 3 реда и 1 колона - 3x1. Кодът реализиращ правата киниматична задача в среда на CodeSys е:

```
FUNCTION ForwardKinematic_V1:
                        Coordinates
VAR_INPUT
    inputs  :   ArmAngles;
END_VAR
VAR
    inputsRadians  : ArmAngles;
    R01: ARRAY[1..3,1..3] OF REAL;
    R02: ARRAY[1..3,1..3] OF REAL;
    R03: ARRAY[1..3,1..3] OF REAL;
    d01: ARRAY[1..1,1..3] OF REAL;
    d02: ARRAY[1..1,1..3] OF REAL;
    d03: ARRAY[1..1,1..3] OF REAL;
    H01: ARRAY[1..4,1..4] OF REAL;
    H02: ARRAY[1..4,1..4] OF REAL;
    H03: ARRAY[1..4,1..4] OF REAL;
    H12: ARRAY[1..4,1..4] OF REAL;
    H  : ARRAY[1..4,1..4] OF REAL;
END_VAR
inputsRadians.Base:=
     DegToRad(Val:=inputs.Base);
inputsRadians.Elbow:=DegToRad(
     Val:=-180.0+inputs.Elbow);
inputsRadians.Shoulder:=
   DegToRad(Val:=inputs.Shoulder);
// Ротационна матрица на основата
R01[1,1]:=COS(inputsRadians.Base);
R01[1,2]:=0;
R01[1,3]:=SIN(inputsRadians.Base);
R01[2,1]:=SIN(inputsRadians.Base);
R01[2,2]:=0;
R01[2,3]:=-COS(inputsRadians.Base);
R01[3,1]:=0; R01[3,2]:=1;R01[3,3]:=0;
// Ротационна матрица на рамото
R02[1,1]:=COS(inputsRadians.Shoulder);
R02[1,2]:=-SIN(inputsRadians.Shoulder);
R02[1,3]:=0;
R02[2,1]:=SIN(inputsRadians.Shoulder);
R02[2,2]:=COS(inputsRadians.Shoulder);
R02[2,3]:=0; R02[3,1]:=0;
R02[3,2]:=0; R02[3,3]:=1;
// Ротационна матрица на китката
R03[1,1]:=COS(inputsRadians.Elbow);
R03[1,2]:=-SIN(inputsRadians.Elbow);
R03[1,3]:=0;
R03[2,1]:=SIN(inputsRadians.Elbow);
R03[2,2]:=COS(inputsRadians.Elbow);
R03[2,3]:=0; R03[3,1]:=0;
R03[3,2]:=0; R03[3,3]:=1;
// Вектор на отместване на основата
d01[1,1]:= GVL.a2*
COS(inputsRadians.Base);
d01[2,1]:= GVL.a2*
SIN(inputsRadians.Base);
d01[3,1]:= GVL.a1;
// Вектор на отместване на рамото
d02[1,1]:= GVL.a3*
COS(inputsRadians.Shoulder);
d02[2,1]:= GVL.a3*
SIN(inputsRadians.Shoulder);
d02[3,1]:= 0;
// Вектор на отместване на китката
d03[1,1]:= GVL.a4*
COS(inputsRadians.Elbow);
d03[2,1]:= GVL.a4*
SIN(inputsRadians.Elbow);
d03[3,1]:= 0;
// Съставяне на хомогенна трансфор-
// мираща матрица за всяко звено
H01:=FillTransformationMatrix(
                    Ro:=R01,d:=d01);
H02:=FillTransformationMatrix(
                    Ro:=R02,d:=d02);
H03:=FillTransformationMatrix(
                    Ro:=R03,d:=d03);
//Умножение на матриците
MatrixMUL( 4, 4, ADR(H01),
           ADR(H02), ADR(H12));
MatrixMUL( 4, 4, ADR(H12),
           ADR(H03),ADR(H));
//Резултат
ForwardKinematic_V1.X:=H[1,4];
ForwardKinematic_V1.Y:=H[2,4];
ForwardKinematic_V1.Z:=H[3,4];
```

Всяка трансформираща матрица се състои от две части: Ротационна матрица и вектор на изместване. Попълването на трансформиращите матрици се извършва от функцията *FillTransformationMatrix*. Тя приема като параметри масив с размер 3x3 (три реда, три стълба) за ротационната матрица и масив с размер 3x1 за вектора на изместване. Върнатия резултат от функцията е масив с размер 4x4, представляващ съответната хомогенна трансформираща матрица. След като се умножат трите хомогенни трансформиращи матрици се получава общата трансформираща матрица за модела. В последната колона редове 1, 2 и 3 на тази матрица се намират стойностите, показващи позицията на инструмента в координатната система на базата съответно X, Y, Z. Функцията връща структура Coordinates, попълнена с координатите на инструмента.

```
ForwardKinematic_V1.X:=H[1,4];
ForwardKinematic_V1.Y:=H[2,4];
ForwardKinematic_V1.Z:=H[3,4];
```

Кинематика в обратна посока

Кинематиката в обратна посока [1,2,3] решава следния проблем: При известна позицията на инструмента (Координатите му X,Y,Z), методът връща ъглите на всички въртящи се звена или дължината на рамото при призматични звена.

В текущата точка е представен графичен метод за намиране на неизвестните промен-

ливи (ъглите на звената на манипулатора). За да се намерят те се използват три основни закона в геометрията:

- *Питагорова теорема*

$$\boldsymbol{a^2 + b^2 = c^2\ ;}$$

- *Косинусова теорема.*

$$a^2 = c^2 + b^2 - 2 * \cos \propto * b * c$$

- *Тригонометричните функции:*

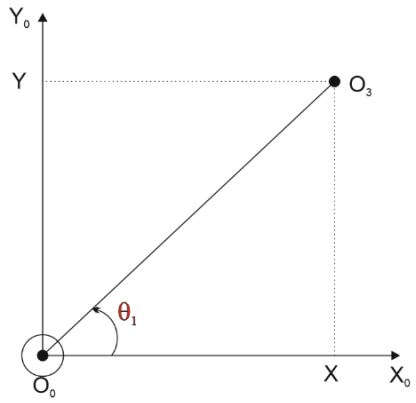
$$\sin \propto, \cos \propto, \tan \propto, \tan^{-1} \propto, \cos^{-1} \propto$$

За даденият модел известните величини са позицията на инструмента в координатното пространство: X, Y, Z ориентирана спрямо основата на манипулатора. Търсените величини са: ъгълът на завъртане на базата θ_1, ъгълът на рамото θ_2 и ъгълът на лакътя θ_3. Уравненията за трите ъгъла се намират като се представи модела в два изгледа: поглед от горе(фиг. 6) и поглед от страни (фиг. 7).

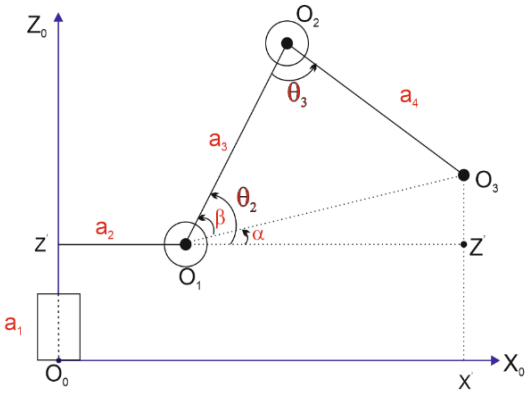
Важен момент при графичният метод е ъглите на манипулатора да не са 0°, 90°, 180°, 270°. Това води до по-ясен изглед и до съставяне на верни уравнения.

$$O_0O_3 = \sqrt{X^2 + Y^2} \qquad (18)$$

$$\theta_1 = \tan^{-1}\frac{Y}{X} \qquad (19)$$



Фиг. 6. *Манипулатор поглед отгоре*



Фиг. 7. *Манипулатор поглед отстрани*

От първият изглед, който показва манипулатора, погледнат отгоре, се съставят две уравнения: за θ_1 и за отсечка O_0O_1, която е проекцията на манипулатора в равнината X_0Y_0.

От вторият изглед се съставят и останалите уравнения от модела.

Отсечката $O_0X' \equiv O_0O_3$ (12) от предишният изглед $\Rightarrow O_1Z' = O_0X' - a_2, O_3Z' = Z - a_1$

$$O_1O_3 = \sqrt{(O_3Z')^2 + (O_1Z')^2} \qquad (20)$$

Ъгъл θ2 представлява сбор от ъглите α и β.

$$\theta_2 = \propto + \beta \qquad (21)$$

Ъгъл α може да се намери от правоъгълния триъгълник O_1O_3Z', като съотношение между срещулежащия катет към прилежащия катет:

$$\frac{O_3Z'}{O_1Z'} = \tan \propto \ \Rightarrow \propto = \tan^{-1}\frac{O_3Z'}{O_1Z'}$$

$$\propto = \tan^{-1}\frac{O_3Z'}{O_1Z'} \qquad (22)$$

Ъгъл β се намира по Косинусова теорема от триъгълник: $O_1O_2O_3$:

$$a_4^2 = a_3^2 + (O_1O_3)^2 - * a_3 * O_1O_3 * \cos\beta \Rightarrow$$

$$\beta = \cos^{-1}\frac{a_3^2 + (O_1O_3)^2 - a_4^2}{2 * a_3 * O_1O_3} \qquad (23)$$

$$\theta_2 = \propto + \beta = \tan^{-1}\frac{O_3Z'}{O_1Z'} + \cos^{-1}\frac{a_3^2 + (O_1O_3)^2 - a_4^2}{2 * a_3 * O_1O_3} \qquad (24)$$

$$\theta_3 = \cos^{-1}\frac{a_3^2 + a_4^2 - (O_1O_3)^2}{2 * a_3 * a_4} \qquad (25)$$

Уравнения (19), (24) и (25) дават ъглите на манипулатора при съответната позиция на инструмента.

Програмна реализация на правата задача

В програмния код функцията, намираща ъглите на манипулатора отговарящи за позицията е *InverseKinematik_V1*. Тя приема като входящ параметър структура с позицията на инструмента и връща структура с ъглите на манипулатора.

Изчисляване на проекцията на манипулатора O_0O_3 в равнината XY на базовия фрейм се определя чрез Питагорова теорема. Ъглите θ_2 и θ_3 се изчисляват по косинусова теорема.

Кодът реализиращ обратната кинематична задача в среда на CodeSys е:

```
FUNCTION InverseKinematik_V1:ArmAngles
VAR_INPUT
    Coordinate      : Coordinates;
END_VAR
VAR
    T1,T2,T3 :  LREAL;
    C, tmp   :  LREAL;
    Xp, Zp,Cp:  LREAL;
END_VAR
// Намиране на ъгълът на завъртане
// на базата
IF Coordinate.X=0 THEN
     T1:=DegToRad(90);
ELSE
   T1:=ATAN2LREAL(x:=Coordinate.X,
                  y:=Coordinate.Y);
END_IF
// Проекция на манипулатора по XY на
// базовия фрейм
C:=SQRT(EXPT(Coordinate.X,2)+
      + EXPT(Coordinate.Y,2));
// Изчисляване на ъглите
// на рамото и лакътя
    Xp:=C-GVL.a2;
// изчислява дължината на отсечка O1Z'
    Zp:=Coordinate.Z-GVL.a1;
// изчислява дължината на отсечка O3Z'
    Cp:=SQRT(Zp*Zp+Xp*Xp);
// изчислява дължината на отсечка O1O3
// изчислява ъгъл θ2
    T2:=ATAN2LREAL(x:=Xp,y:=zp) +
       ACOS((GVL.a3*GVL.a3 + Cp*Cp -
       GVL.a4*GVL.a4)/(2*GVL.a3*Cp));
    // изчислява ъгъл θ3
    tmp:=(2*GVL.a3*GVL.a4);
T3 :=ACOS((GVL.a3*GVL.a3 +
          GVL.a4*GVL.a4-Cp*Cp)/tmp);
InverseKinematik_V1.Base :=
    LREAL_TO_REAL(Rad_To_Deg(T1));
InverseKinematik_V1.Shoulder :=
    LREAL_TO_REAL(Rad_To_Deg(T2));
InverseKinematik_V1.Elbow :=
    LREAL_TO_REAL(Rad_To_Deg(T3));
InverseKinematik_V1.GripperUpDown:=
  (180-(InverseKinematik_V1.Elbow +
      InverseKinematik_V1.Shoulder))+
      Coordinate.GripperUpDown ;
InverseKinematik_V1.Gripper :=
              Coordinate.Gripper;
InverseKinematik_V1.ElbowRotate:=
    (90-InverseKinematik_V1.Base)+
     Coordinate.ElbowRotate;
InverseKinematik_V1.GripperRotate:=
    (90-InverseKinematik_V1.Base)+
        Coordinate.GripperRotate;
```

ЗАКЛЮЧЕНИЕ

В статията е представено управлението на манипулатор с три степени на свобода от PLC изградено на базата на едноплатков компютър Raspberry Pi 3. Решени са правата и обратна кинематични задачи. Разработен е софтуер за управление на манипулатора в среда на CodeSys. В заключение софтуерната среда за програмиране CoDeSys успешно превръща вградената система Raspberry Pi в контролер, управляващ технологични процеси. Чрез нея могат да се автоматизират различни дейности в индустрията, като товароразтоварни, асемблиращи, палетизиращи комплекси и др. Въпреки малкия си размер, тя не отстъпва по възможности на специализираните технологични контролери.

Недостатък е ниското напрежение на входовете и изходите, които не могат да се използват директно в индустриалната автоматизация, но със съгласуващи схеми и този проблем може да бъде решен.

REFERENCE

[1] Kucuk S. , Z. Bingul, Robot Kinematics: Forward and Inverse Kinematics, INTECH Open Access Publisher, 2006, ISBN 3866112858, 9783866112858

[2] Kim, J. H. & Kumar, V. R. Kinematics of robot manipulators via line transformations, J. Robot. Syst.,1990, Vol. 7, No., 4, pp. 649–674

[3] Sam Cubero, Industrial Robotics.Theory, Modelling and Control, Advanced Robotic Systems International 2007, ISBN 3-86611-285

[4] Raspberry Pi — Teach, Learn, and Make with Raspberry Pi, https://www.raspberrypi.org, . 04.10.2018

[5] CODESYS Control for Raspberry Pi SL, https://store.codesys.com/codesys-control-for-raspberry-pi-sl.html. 04.10.2018

БЛАГОДАРНОСТ

1. Докладът се публикува във връзка с проект № 1710E/2017
2. Резултатите публикувани в доклада са свързани с НИР по проект № 1710E/2017 към ФЕЕ