Podstawy robotyki

Praca domowa nr 2

Maciej Dominiak

TEMAT: Programowanie trajektorii manipulatora

1. Planowanie trajektorii point-to-point

Do rozwiązania zadania dostajemy następujące dane: położenie początkowe w kątach na poszczególnych przegubach $(\theta_1, \theta_2, \theta_3)$, punkt docelowy we współrzędnych kartezjańskich (p_x, p_y, p_z) oraz czas t_1, w którym ma być wykonany ruch.

Położenie końcowe przegubów możemy wyliczyć z podanego punktu docelowego, korzystając z równań kinematyki odwrotnej, z poprzedniej pracy domowej. Mając do dyspozycji kilka rozwiązań wybieramy te, które będzie najlepsze, czyli znajduje się najbliżej położenia początkowego ramienia.

Idąc dalej, ustalamy że $t_0 = 0$. Zbierając wszystkie dane możemy wyliczyć:

Prędkość maksymalną, osiąganą w połowie ruchu: $V = \frac{2(u_1 - u_0)}{t_1}$

Wymagane przyśpieszenie: $a = \frac{V}{\frac{1}{2}t_1}$

Przechodząc do modelu matematycznego trajektorii bang-bang kolejno wyznaczamy, segment początkowy, dla $t \in < t_0, \frac{1}{2}t_1 >$ (przyśpieszenie po czasie, prędkość po czasie oraz położenie po czasie):

$$a(t) = a \quad stałe$$

$$v(t) = at$$

$$u(t) = u_0 + \frac{at^2}{2}$$

oraz segment końcowy, dla $t \in < \frac{1}{2}t_1, t_1 >$:

$$a(t) = -a \quad stałe$$

$$v(t) = a(t_1 - t)$$

$$u(t) = u_1 - \frac{a}{2}t_1^2 + at_1 t - \frac{a}{2}t^2$$

Wybierając krok czasowy oraz przeprowadzając obliczenia dla każdego przegubu otrzymujemy zakres kątów, jakie będą miały przeguby podczas wykonywania ruchu. By zobrazować ruch ramienia na wykresie, za pomocą kinematyki prostej, zamienię kąty przegubów na punkty w przestrzeni.

Kilka przykładów:

$$\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\end{bmatrix} = \begin{bmatrix}0\\0\\0\end{bmatrix}, \begin{bmatrix}p_x\\ p_y\\ p_z\end{bmatrix} = \begin{bmatrix}0\\-456\\350\end{bmatrix}, t_1 = 2, krok = 0.02$$

$$\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\end{bmatrix} = \begin{bmatrix}0\\0\\0\end{bmatrix}, \begin{bmatrix}p_x\\ p_y\\ p_z\end{bmatrix} = \begin{bmatrix}150\\350\\150\end{bmatrix}, t_1 = 2, krok = 0.02$$

Do pracy dołączam aplikację, w której możemy samodzielnie podać parametry wejściowe oraz wygenerować wykres przebiegu obrotu ramienia. By uruchomić aplikację należy wejść do folderu „aplikacja" i uruchomić plik „index.html". Program powinien działać w każdej przeglądarce internetowej(osobiście używałem przeglądarki Google Chrome). By obliczyć konfigurację końcową przegubów, przyśpieszenie kątowe, oraz maksymalną prędkość, należy wprowadzić wszystkie dane wejściowe i nacisnąć przycisk oblicz, gdy jakieś pole będzie puste program niczego nie obliczy. Precyzję wyniku podajemy w miejscach po przecinku. Gdy nie można zrealizować trajektorii, zamiast tabeli z wynikiem, wyświetli się komunikat „Brak rozwiązań". By pokazać listę kroków klikamy w przycisk „Pokaż listę kroków", dane po lewej stronie muszą być wprowadzone. Jeśli nie wpiszemy parametru kroku, domyślnie będzie to 0,02[s]. Klikając w przycisk „Pokaż tor ruchu", generujemy wykres mający obrazować położenie ramienia. Wykres można obracać przez przeciąganie kursorem.

2. Planowanie trajektorii liniowej

W tym zadaniu interesuje nas trajektoria jaką porusza się końcówka robota więc najpierw musimy wyznaczyć zbiór punktów na trasie tej trajektorii, a dopiero później, korzystając z kinematyki odwrotnej, znaleźć konfigurację przegubów.

Na początek dostajemy dane: położenie początkowe w kątach na poszczególnych przegubach $(\theta_1, \theta_2, \theta_3)$, punkt docelowy we współrzędnych kartezjańskich (p_x, p_y, p_z), czas t_1, w którym ma być wykonany ruch oraz V, prędkość liniowa ruchu w czasie centralnego segmentu trajektorii.

Liniowa trajektoria ma postać:

$$\begin{bmatrix} P_x(t) \\ P_y(t) \\ P_z(t) \end{bmatrix} = \begin{bmatrix} P_x(t_0) \\ P_y(t_0) \\ P_z(t_0) \end{bmatrix} + U(t)\vec{k}$$

gdzie,

$$\vec{k} = \begin{bmatrix} P_x(t_1) - P_x(t_0) \\ P_y(t_1) - P_y(t_0) \\ P_z(t_1) - P_z(t_0) \end{bmatrix} * \frac{1}{\sqrt{[P_x(t_1) - P_x(t_0)]^2 + \left[P_y(t_1) - P_y(t_0)\right]^2 + [P_z(t_1) - P_z(t_0)]^2}}$$

Pozostaje nam wyznaczyć jednowymiarową przestrzeń $U(t)$.

Zauważamy że:

$$U(t_0) = u_0 = 0$$

$$U(t_1) = u_1 = \sqrt{[P_x(t_1) - P_x(t_0)]^2 + \left[P_y(t_1) - P_y(t_0)\right]^2 + [P_z(t_1) - P_z(t_0)]^2}$$

Wprowadzając dane musimy pamiętać o następującym ograniczeniu:

$$\frac{u_1 - u_0}{V} < t_1 \leq \frac{2(u_1 - u_0)}{V}$$

Wyznaczamy czas przełączenia, czyli czas po jakim osiągniemy zadaną prędkość:

$$t_b = \frac{u_0 - u_1 + Vt_1}{V}$$

Mając t_b wyznaczamy przyśpieszenie:

$$a = \frac{V}{t_b}$$

Przechodząc do modelu matematycznego trajektorii LSPB kolejno wyznaczamy, segment początkowy $t \in < t_0, t_b >$ (przyśpieszenie po czasie, prędkość po czasie oraz położenie po czasie):

$$a(t) = a$$

$$v(t) = at$$

$$u(t) = \frac{at^2}{2}$$

Segment centralny $t \in < t_b, t_1 - t_b >$:

$$a(t) = 0$$

$$v(t) = V$$

$$u(t) = Vt + \frac{u_1 - Vt_1}{2}$$

Segment końcowy $t \in < t_1 - t_b, t_1 >$:

$$a(t) = -a$$

$$v(t) = a(t_1 - t)$$

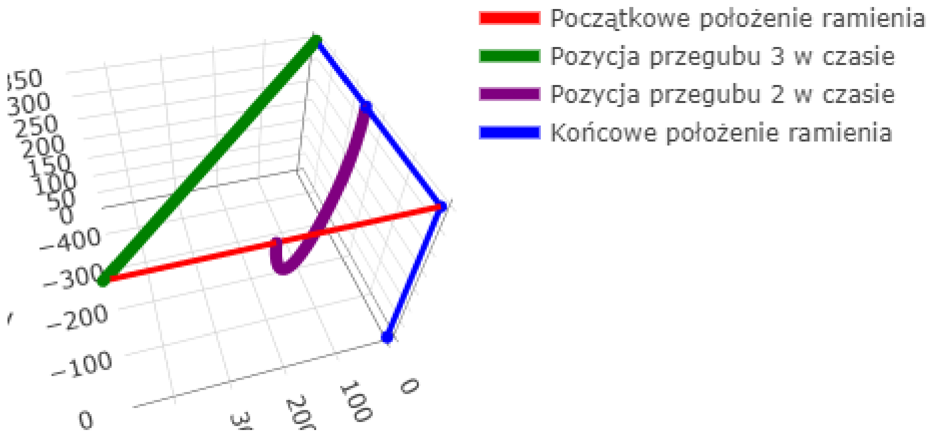
$$u(t) = u_1 - \frac{a}{2}t_1^2 + at_1 t - \frac{a}{2}t^2$$

Ustalając krok czasowy otrzymamy tablicę parametrów u, którą wykorzystujemy do obliczenia kolejnych punktów w przestrzeni, reprezentujących trajektorię liniową. Pozostaje nam zastosować kinematykę odwrotną by znaleźć listę konfiguracji przegubów podczas ruchu.

Równania mają postać: $y = ax^2 + bx + c$, więc tabela z ich współczynnikami będzie miała postać:

Równania\Współczynniki	a	b	c
Początkowe	$\frac{1}{2}a$	0	0
Centralne	0	V	$\frac{u_1 - Vt_1}{2}$
Końcowe	$-\frac{1}{2}a$	at_1	$u_1 - \frac{1}{2}at_1^2$

Kilka przykładów:

$\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\end{bmatrix} = \begin{bmatrix}0\\0\\0\end{bmatrix}, \begin{bmatrix}p_x\\p_y\\p_z\end{bmatrix} = \begin{bmatrix}0\\-456\\350\end{bmatrix}, V = 100, t_1 = 8, krok = 0.02$


$$\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\end{bmatrix}=\begin{bmatrix}0\\0\\0\end{bmatrix},\ \begin{bmatrix}p_x\\ p_y\\ p_z\end{bmatrix}=\begin{bmatrix}150\\350\\150\end{bmatrix}, V=100, t_1=8, krok=0.02$$

Ta część zadania również ma swoją realizację w aplikacji. Jest to ten sam program co wcześniej, z tym że pola do wprowadzania danych są niżej. Z racji ograniczenia na czas ruchu, po wpisaniu położenia początkowego, końcowego oraz prędkości, można kliknąć przycisk „oblicz”, a pod polem do wprowadzania czasu pojawi się zakres w jakim powinna być ta zmienna.

3. Planowanie trajektorii kołowych

Podobnie jak w zadaniu drugim, musimy najpierw wyznaczyć trajektorię w przestrzeni kartezjańskiej, a dopiero później przejść do konfiguracji przegubów.

Do trajektorii bang-bang potrzebne będą dane takie jak: czas początkowy t_0(domyślnie równy zero), czas ruchu t_1, trzy położenia końcówki ramienia w układzie współrzędnych, na początku, pomiędzy początkiem, a końcem oraz na końcu ruchu, przyjmiemy ich oznaczenia następująco *A*, *B*, *C*.

Mając trzy punkty, jesteśmy w stanie policzyć odległości pomiędzy nimi oraz stwierdzić czy można z nich stworzyć trójkąt. W przypadku gdy nie można stworzyć trójkąta, nie uda się również poprowadzić trajektorii kołowej. W drugim przypadku, wiemy, że na każdym trójkącie da się opisać okrąg, co więcej środek tego koła leży na przecięciu symetralnych boków trójkąta, które leżą na płaszczyźnie wyznaczanej przez dwa wektory np. AB i AC. Promień takiego okręgu wynosi:

$$r=\frac{abc}{4p}$$

gdzie, *a*, *b*, *c* to długości boków trójkąta, a *p* to pole trójkąta. Pole trójkąta policzymy ze wzoru Herona:

$$p=\sqrt{s(s-a)(s-b)(s-c)}$$

gdzie, *s* to połowa obwodu trójkąta.

Posiadając tą wiedzę jesteśmy w stanie sprowadzić problem do początku układu współrzędnych, dlatego w kolejnych wzorach będę pomijał dodawanie współrzędnych środka koła, czy sfery.

Rozpatrzmy najprostszy przypadek w przestrzeni dwuwymiarowej, gdzie skorzystalibyśmy z parametrycznego równania koła:

$$\begin{cases}x(u)=r*\cos(u)\\ y(u)=r*\sin(u)\end{cases}$$

Jednowymiarowa przestrzeń $U(t)$ przechodziłaby od kąta punktu A, do kąta punktu C, po drodze odwiedzając punkt B. Kąty tych punktów można łatwo pozyskać wykorzystując funkcję *atan2(y,x)*.

Przechodząc do przestrzeni trójwymiarowej, spróbujmy skorzystać z równania parametrycznego sfery:

$$\begin{cases} x = r * \sin(\theta) * \cos(\varphi) \\ y = r * \sin(\theta) * \sin(\varphi) \\ z = r * \cos(\theta) \end{cases}$$

Gdzie, kąt φ jest długością azymutalną, a kąt θ odległością zenitalną. Chcąc wyznaczyć trajektorię z takiej postaci musimy stworzyć dwie przestrzenie jednowymiarowe, jedną dla kąta φ, drugą dla kąta θ. Wiemy już że kąt φ, można wyznaczyć z funkcji *atan2(y,x)*, jeśli chodzi o kąt θ, to możemy zastosować funkcję *arcos(z/r)*. Jednak przy takim rozwiązaniu pojawiają się pewne problemy.

Po pierwsze, gdy koło leży w przestrzeni pionowej, na przykład x=0, y=0, czy x+y=0, jest potrzeba by kąt φ pozostał stały, a kąt θ zmieniał się o więcej niż 180 stopni. Można to obejść, wykrywając te przypadki i dostosowując kąty.

Drugim problemem jest jednowymiarowa przestrzeń dla kąta θ. Są przypadki gdy ten kąt najpierw wzrasta, a później opada(lub na odwrót). Ten problem trudniej jest rozwiązać, ponieważ trudno jest określić jak maksymalnie powinien wzrosnąć kąt oraz w jakim momencie będzie największy.

Możemy podejść do problemu inaczej, wyznaczając najpierw równanie płaszczyzny, na której znajdują się trzy dane punkty i zrzutujmy koło będące na tej płaszczyźnie na płaszczyznę OXY.

Tworzymy równanie płaszczyzny. Najpierw liczymy dwa wektory(nie zmieniam oznaczeń punktów ale są to punkty przesunięte do początku układu współrzędnych, środek koła, który wyznaczają jest w punkcie (0, 0, 0)):

$$\overrightarrow{AB} = B - A$$
$$\overrightarrow{AC} = C - A$$

Później obliczamy ich iloczyn wektorowy:

$$\overrightarrow{AB} \times \overrightarrow{AC} = [D, E, F]$$

Wypisujemy ogólną postać równania płaszczyzny i pamiętając, że płaszczyzna na pewno zawiera punkt (0, 0, 0) wstawiamy go do równania:

$$D(x - x_0) + E(y - y_0) + F(z - z_0) = 0$$

$$Dx + Ey + Fz = 0$$

Wyliczamy współrzędną *z*:

$$z = \frac{Dx + Ey}{-F}$$

W tym miejscu można zauważyć, że jeśli F=0 to mamy dzielenie przez zero. Jest to podobny problem co poprzednio, z równaniem parametrycznym sfery. Jeśli płaszczyzna jest pionowa(np. x=0, y=0, x+y=0) to obraz koła na płaszczyźnie OXY będzie prostą.

Idąc dalej, przypominamy sobie równanie sfery, gdy środek sfery jest w początku układu współrzędnych to równanie wygląda następująco:

$$x^2 + y^2 + z^2 = r^2$$

Jeśli podstawimy wyliczoną współrzędną *z* do równania sfery, to otrzymamy równanie elipsy. Będzie to odbicie koła z przestrzeni na płaszczyznę OXY.

$$\frac{F^2+D^2}{r^2F^2}x^2+2\frac{DE}{r^2F^2}xy+\frac{F^2+E^2}{r^2F^2}y^2=1$$

Przyjmijmy:

$$J=\frac{F^2+D^2}{r^2F^2}, K=\frac{DE}{r^2F^2}, L=\frac{F^2+E^2}{r^2F^2}$$

Co można zapisać w macierzy:

$$\begin{bmatrix} J & K \\ K & L \end{bmatrix}$$

Gdy wyznacznik macierzy oraz jej ślad są dodatnie to istnieje elipsa.

Inna postać równania elipsy:

$$\frac{(x*\cos(\beta)+y*\sin(\beta))^2}{a^2}+\frac{(y*\cos(\beta)-x*\sin(\beta))^2}{b^2}=1$$

Rozpisując dostajemy:

$$x^2\frac{(b^2\cos^2(\beta)+a^2\sin^2(\beta))}{a^2b^2}+y^2\frac{(b^2\sin^2(\beta)+a^2\cos^2(\beta))}{a^2b^2}+2xy\frac{(b^2\cos(\beta)\sin(\beta)-a^2\cos(\beta)\sin(2\beta))}{a^2b^2}=1$$

Przyrównując kolejne współczynniki rozpisanego równania do *J, L, K* dostajemy trzy równania z trzema niewiadomymi. Po ich rozwiązaniu znamy wartości *a, b* oraz β.

Dalej możemy zapisać równanie parametryczne elipsy:

$$\begin{cases} x=a*\cos(\alpha)*\cos(\beta)-b*\sin(\alpha)\sin(\beta) \\ y=a*\cos(\alpha)*\sin(\beta)+b*\sin(\alpha)\cos(\beta) \end{cases}$$

Wystarczy stworzyć jednowymiarową przestrzeń dla kąta α, od u_0 do u_1, wykorzystując funkcję *atan2(y,x)*.

Uwzględniając położenie początkowe, trajektoria ma postać:

$$\begin{bmatrix} P_x(t) \\ P_y(t) \\ P_z(t) \end{bmatrix}=\begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix}+\begin{bmatrix} a*\cos(\alpha(u))*\cos(\beta)-b*\sin(\alpha(u))\sin(\beta) \\ a*\cos(\alpha(u))*\sin(\beta)+b*\sin(\alpha(u))\cos(\beta) \\ z_p \end{bmatrix}$$

gdzie, x_k, y_k, z_k są współrzędnymi środka koła, a z_p obliczane jest z równania płaszczyzny.

Przechodząc do jednowymiarowej płaszczyzny, w trajektorii bang-bang, możemy zapisać wzory:

$$u_0=atan2(A_y,A_x)$$

$$u_1=atan2(C_y,C_x)$$

Jeśli $atan2(B_y,B_x)$, nie znajduje się pomiędzy u_0, a u_1, należy zmodyfikować u_0 oraz u_1 poprzez dodawanie 360 stopni do u_0 lub u_1.

Dalsze wzory tak jak w zadaniu pierwszym:

Prędkość maksymalna, osiągana w połowie ruchu: $V = \dfrac{2(u_1 - u_0)}{t_1}$

Wymagane przyśpieszenie: $a = \dfrac{V}{\frac{1}{2}t_1}$

Segment początkowy $t \in < t_0, \frac{1}{2}t_1 >$ (przyśpieszenie po czasie, prędkość po czasie oraz położenie po czasie):

$$a(t) = a \quad stałe$$

$$v(t) = at$$

$$u(t) = u_0 + \frac{at^2}{2}$$

oraz segment końcowy $t \in < \frac{1}{2}t_1, t_1 >$:

$$a(t) = -a \quad stałe$$

$$v(t) = a(t_1 - t)$$

$$u(t) = u_1 - \frac{a}{2}t_1^2 + at_1t - \frac{a}{2}t^2$$

Uwagi dotyczące realizacji zadania.

O ile jesteśmy w stanie analitycznie rozwiązać takie zadanie, tak chcąc je zaprogramować napotkamy pewne trudności.

Po pierwsze obliczenie środka koła może nie być łatwe. Jednak można znaleźć w Internecie informację jak to zrobić wykorzystując współrzędne barycentryczne. Nie zgłębiałem tematu ale programy, które znalazłem są stosunkowo proste do napisania.

Pominąłem również obliczanie współczynników elipsy *a, b* oraz β. Są to równania trygonometryczne, które nie zawsze udaje się rozwiązać analitycznie, czasami trzeba się wspomóc metodami numerycznymi.

Jest jeszcze jeden pomysł, który zauważyłem po przejrzeniu mojej pracy. Mianowicie, gdy podstawimy współrzędną *z* z równania płaszczyzny, do parametrycznego równania sfery, otrzymamy wówczas

$$\begin{cases} x = r * sin(\theta) * cos(\varphi) \\ y = r * sin(\theta) * sin(\varphi) \\ \dfrac{Dx + Ey}{-F} = r * cos(\theta) \end{cases}$$

trzy równania z trzema niewiadomymi x, y oraz θ, jednak znów są to równania trygonometryczne.