confidential

Stewart Platform을 이용한 Dispenser Head 자세 제어 계산 알고리듬

2020.04.03

서울과기대 김정한

1. 개요

본 문서는 Stewart Platform을 이용한 디스펜서 헤드의 위치 결정 알고리듬에 대한 내용을 담고 있다. Stewart Platform 방식의 스테이지 위에 디스펜서 노즐이 장착되면 상당히 자연스러운 디스펜싱 작업이 가능하게 되며, 겐트리 타입의 스테이지에 단순히 축 수를 늘린 구조에서 수행할 수 없는 연속적인 작업이 가능해져서, 불필요한 움직임을 최소화 할 수 있어서 성능과 속도 두 가지 지표에서 모두 혁신적인 개선이 가능하다.

Fig.1 Stewart Platform

Stewart Platform은 병렬식 지지 구조로 인하여 기존의 시리얼 방식에 푸마타입 로봇에서

볼 수 없는 큰 강성을 가지며, 작업 영역work space은 좁지만, 스테이지 위의 공간을 최대한으로 활용하는 것이 가능해서 이를 디스펜서 헤드에 적용한다면, 타사의 장비와 비교하여 상당히 유리해진다.

이러한 아이디어를 상용화 하려면 몇 가지 기술적인 어려움이 있는데, 본 문서에서는 이러한 아이디어를 장비에 적용할 수 있도록 최대한 쉽고 단순하게 알고리듬을 개발하려고 한다.

2. 실행전략

본 문서는 학문적인 용도가 아닌, 혁신적인 장비 개발에 기술을 Porting 하는 것이 목표이므로, 최대한 쉽고 간결하게 장비 개발자들에게 단순한 방법으로 상기와 같은 헤드부를 설계/제어할 수 있는 문서를 제공함에 있다. 따라서 복잡한 수식이 뒤따르는 중간과정을 최대한 생략하고 결과위주로 정리하려고 하며, 소프트웨어로 구현이 가능하도록 알고리듬을 개조할 것이다.

실제 장비는 조립공차가 반드시 발생하므로 이를 보정하기 위한 알고리듬이 사전에 고려되어야 하며, 단계별로 과정을 정리한다. 만약 실제 장비의 설계시 기타의 사정으로 구조가 변경되어야 한다면, 따로 협의한다.

3. 좌표설정 및 형상정의

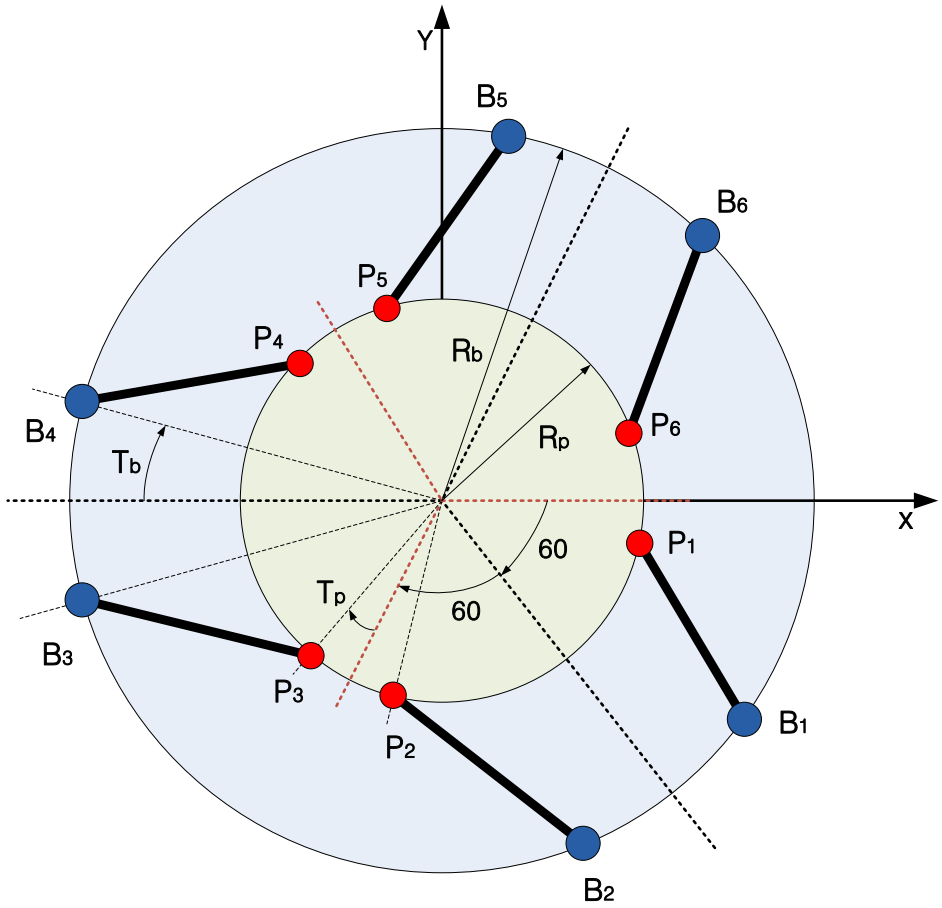

Fig.2 Coordinate definition

⊙ Nomenclature:

B_1 ~ B_6 : 하판(Base Plate)의 힌지 좌표

P_1 ~ P_6 : 상판(Upper Plate)의 힌지 좌표

R_P : 상판(Upper Plate)위의 힌지까지의 반지름

R_B : 하판(Base Plate)위의 힌지까지의 반지름

T_B : 하판에서의 힌지간 오프셋 각도

T_P : 상판에서의 힌지간 오프셋 각도

h : 상판과 하판 사이의 거리

T_o : Tool Offset 좌표(상판좌표계)

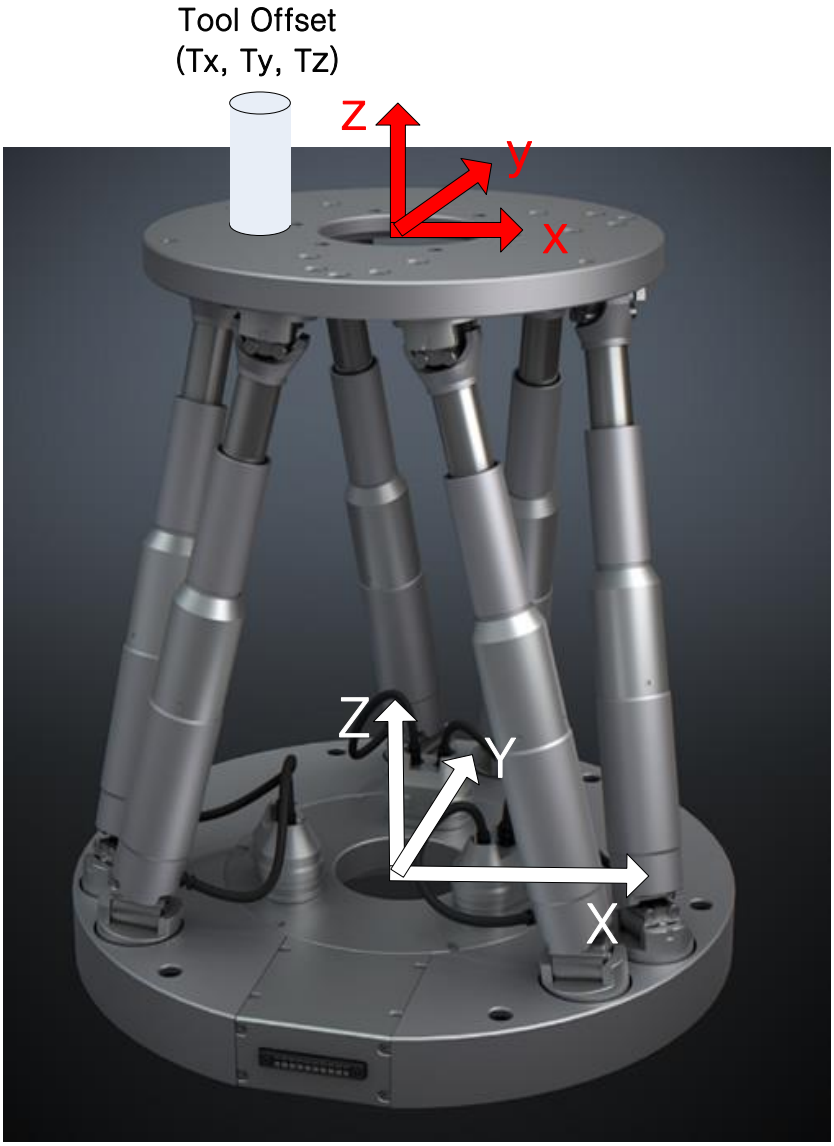

Fig.3 Coordinates with tool offset

본 연구 보고서의 대상은 상기와 같이 Stewart Platform을 사용한 Tool Head 이다. 상판에 기구적인 오프셋을 가진 툴이 장착되어 있고, **이 툴의 끝부분의 위치와 오리엔테이션의 6축제어**를 목표로 한다.

실제 장비에서는 툴 부분의 끝이 디스펜서 노즐 부분으로 설계될 수 있으며, 이러한 경우, 디스펜서 노즐의 액체 각도 분사를 임의의 방향으로 분사할 수 있어서 디스펜서 장비의 성능과 속도를 크게 향상시킬 수 있다.

4. 역기구학의 구현 순서

본 문서는 연구 논문이 아니고, 상기 구조의 디스펜서 용 6축 분사 헤드를 구현하는 것이 목표이므로 중간 유도 과정은 생략하고 결과식을 중심으로 소프트웨어 구현의 관점에서 순차적으로 서술하도록 한다.

1. 좌표계는 기본적으로 Base Plate의 원점을 중심으로 한 **기준좌표계 (X-Y-Z)**와, 상판의 중심점을 기준으로 하는 **상판좌표계 (x-y-z)**를 사용한다.

2. 기준좌표계**(X-Y-Z)**상에서 설계도면에 의하여 초기상태의 상판 조인트 위치벡터를 구한다.

$$p_i = R_p(cos\lambda_i \;\; sin\lambda_i \;\; \frac{h}{R_p})' \tag{1}$$

$$\lambda_i = -\frac{\pi}{3}(\mathrm{i}-1) - T_P \qquad \text{for i=1, 3, 5} \tag{2}$$

$$\lambda_i = -\frac{\pi}{3}i + T_P \qquad \text{for i=2, 4, 6} \tag{3}$$

3. 기준좌표계**(X-Y-Z)**상에서 설계도면에 의하여 초기상태의 하판 조인트 위치벡터를 구한다.

$$b_i = R_b(cos\Lambda_i \;\; sin\Lambda_i \;\; 0)' \tag{4}$$

$$\Lambda_i = -\frac{\pi}{3}i + T_B \qquad \text{for i=1, 3, 5} \tag{5}$$

$$\Lambda_i = -\frac{\pi}{3}(\mathrm{i}-1) - T_B \qquad \text{for i=2, 4, 6} \tag{6}$$

4. 기준좌표계**(X-Y-Z)**상에서 X-Y-Z 순으로 회전하는 오일러 앵글 회전변환 매트릭스를 정의한다 (결과식은 (9)).

$$\boldsymbol{R}_M = R_Z(\gamma)R_Y(\beta)R_X(\alpha) \tag{7}$$

$$= \begin{bmatrix} C_\gamma & -S_\gamma & 0 \\ S_\gamma & C_\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C_\beta & 0 & S_\beta \\ 0 & 1 & 0 \\ -S_\beta & 0 & C_\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & C_\alpha & -S_\alpha \\ 0 & S_\alpha & C_\alpha \end{bmatrix} \tag{8}$$

$$= \begin{bmatrix} C_\beta C_\gamma & S_\alpha S_\beta C_\gamma - C_\alpha S_\gamma & C_\alpha S_\beta C_\gamma + S_\alpha S_\gamma \\ C_\beta S_\gamma & S_\alpha S_\beta S_\gamma + C_\alpha C_\gamma & C_\alpha S_\beta S_\gamma - S_\alpha C_\gamma \\ -S_\beta & S_\alpha C_\beta & C_\alpha C_\beta \end{bmatrix} \tag{9}$$

여기서 $S_\alpha = sin\alpha \;\; C_\alpha = cos\alpha,\; S_\beta = sin\beta \;\; C_\beta = cos\beta,\; S_\gamma = sin\gamma \;\; C_\gamma = cos\gamma.$

5. 상판 조인트 위치벡터 p_i의 오일러 앵글 회전변환 매트릭스를 구한다(i=1, ... ,6).

$$P_i = \boldsymbol{R}_M p_i \tag{10}$$

6. 오일러 앵글 α, β, γ의 회전변환에 따른 상판의 원점의 이동을 보정한다.

우선 기준 좌표계에 대한 상판의 원점의 초기값 [0 0 h]'의 회전변환 값을 구한 후, 그 벡터 차이를 구하여 보정한다.

회전변환에 의해 이동한 벡터차이 $\mathrm{O_{comp}} = \boldsymbol{R}_M[0\ 0\ \mathrm{h}]' - [0\ 0\ \mathrm{h}]'$ (11)

$$P_i = \boldsymbol{R}_M p_i - O_{comp} \tag{12}$$

7. 오일러 앵글 α, β, γ의 회전변환에 따른 상판의 Tool Offset의 이동을 보정한다.

회전변환에 의해 이동한 툴오프셋량 $\mathrm{T_{comp}} = \boldsymbol{R}_M T_o$ (13)

$$P_i = \boldsymbol{R}_M p_i - O_{comp} - T_{comp} \tag{14}$$

여기서 $\mathrm{T_O}$ 는 상판 좌표계(**x-y-z**)에 대한 Tool Offset 량

8. 최종적으로 구한 식(14)에서 이동하고 싶은 위치를 더해주어 최종 목표 벡터 $\boldsymbol{P}_i$ 를 구한다.

$$\boldsymbol{P}_i = \boldsymbol{R}_M p_i - O_{comp} - T_{comp} + T_P \tag{15}$$

여기서 $\mathrm{T_P}$ 는 상판 좌표계(**x-y-z**)에 대한, 이동하고 싶은 위치 좌표

9. 최종 목표 벡터 $\boldsymbol{P}_i$에서(15) 베이스 벡터 b_i 를 빼서 Stewart Platform의 6개의 다리의 길이 벡터 S_i를 구한다.

$$S_i = \boldsymbol{P}_i - \boldsymbol{b}_i \tag{16}$$

10. 6개의 다리의 길이 벡터 S_i의 Norm을 구하여 해당 길이로 제어한다. 식(17)에서 구해지는 L 값이 해당 리니어모터(혹은 실린더)의 제어 대상값인 Target Length가 된다.

$$L_i = |\boldsymbol{P}_i - \boldsymbol{b}_i| \tag{17}$$

■

cf.) 첨부 matlab 파일 참고.

5. Matlab simulation file

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Dispenser head using Stewart platform
%   Han Kim
%   2020.03
%   Seoul National univ. of Sci. and Tech.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear;
%%%%%%%%%%%%%%%%%%%%%%%%% HARDWARE SPECIFICATION
Ph_b=15*pi/180; % Offset angle of the hinge in the base
Ph_p=10*pi/180; % Offset angle of the hinge in the upper plate
Rb=0.16;             % radious of the base
Rp=0.08;             % radious of the upper plate
h=0.16;              % height between two plates
T_Offset = [0 0.01 0.01]';     % Tool offset on the upper plate


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%% COMMANDS with compensation
al=(10*pi)/180;     % Target Roll
be=(0*pi)/180;     % Target Pitch
ga=(0*pi)/180;     % Target Yaw

RM=[cos(be)*cos(ga), sin(al)*sin(be)*cos(ga)-cos(al)*sin(ga), cos(al)*sin(be)*cos(ga)+ sin(al)*sin(ga);
cos(be)*sin(ga), sin(al)*sin(be)*sin(ga)+cos(al)*cos(ga), cos(al)*sin(be)*sin(ga)- sin(al)*cos(ga);
-sin(be), sin(al)*cos(be), cos(al)*cos(be)];

O_mod = RM*[0 0 h]';
O_com = [O_mod(1)   O_mod(2) (O_mod(3)-h)]';

T_com = RM*T_Offset;

Xpe = [0 0 0]';
Xp = Xpe - O_com - T_com;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%% Angles of 12 hinges(base, upper)
Pb1= -pi/3 + Ph_b;
Pb3= -pi + Ph_b;
Pb5= -5*pi/3 + Ph_b;

Pb2= -pi/3 - Ph_b;
Pb4= -pi - Ph_b;
```

```
Pb6= -5*pi/3 - Ph_b;

Pp1=    - Ph_p;
Pp3= -2*pi/3 - Ph_p;
Pp5= -4*pi/3 - Ph_p;

Pp2= -2*pi/3 + Ph_p;
Pp4= -4*pi/3 + Ph_p;
Pp6= -2*pi + Ph_p;

%%%%%%%%%%%%%%%%% Coordinate of 12 hinges(base, upper)
B1x=Rb*cos(Pb1);
B1y=Rb*sin(Pb1);
B2x=Rb*cos(Pb2);
B2y=Rb*sin(Pb2);

B3x=Rb*cos(Pb3);
B3y=Rb*sin(Pb3);
B4x=Rb*cos(Pb4);
B4y=Rb*sin(Pb4);

B5x=Rb*cos(Pb5);
B5y=Rb*sin(Pb5);
B6x=Rb*cos(Pb6);
B6y=Rb*sin(Pb6);

P1x=Rp*cos(Pp1);
P1y=Rp*sin(Pp1);
P2x=Rp*cos(Pp2);
P2y=Rp*sin(Pp2);

P3x=Rp*cos(Pp3);
P3y=Rp*sin(Pp3);
P4x=Rp*cos(Pp4);
P4y=Rp*sin(Pp4);

P5x=Rp*cos(Pp5);
P5y=Rp*sin(Pp5);
P6x=Rp*cos(Pp6);
P6y=Rp*sin(Pp6);

%%%%%%%%%%%%%%%%% Figure drawing
figure(1);hold on;axis([ -0.3 0.3 -0.3 0.3]); grid on;
plot(B1x,B1y,'o-');plot(B2x,B2y,'o-');
plot(B3x,B3y,'o-');plot(B4x,B4y,'o-');
plot(B5x,B5y,'o-');plot(B6x,B6y,'o-');

plot(P1x,P1y,'+');plot(P2x,P2y,'+');
```

```
plot(P3x,P3y,'+');plot(P4x,P4y,'+');
plot(P5x,P5y,'+');plot(P6x,P6y,'+');

%%%%%%%%%%%%%%%%% Final coordinates of the hinges(base, upper)
B = [ B1x, B2x, B3x, B4x, B5x, B6x;
      B1y, B2y, B3y, B4y, B5y, B6y;
      0,0,0,0,0,0];

P = [ P1x, P2x, P3x, P4x, P5x, P6x;
      P1y, P2y, P3y, P4y, P5y, P6y;
      h,h,h,h,h,h];

%%%%%%%%%%%%%%%%% target postion of the P vector
PP1 = RM*P(:,1) + Xp;
PP2 = RM*P(:,2) + Xp;
PP3 = RM*P(:,3) + Xp;
PP4 = RM*P(:,4) + Xp;
PP5 = RM*P(:,5) + Xp;
PP6 = RM*P(:,6) + Xp;

%%%%%%%%%%%%%%%%% length vector
S1=PP1 - B(:,1);
S2=PP2 - B(:,2);
S3=PP3 - B(:,3);
S4=PP4 - B(:,4);
S5=PP5 - B(:,5);
S6=PP6 - B(:,6);

%%%%%%%%%%%%%%%%% target length of the each actuator
L1 = norm(S1);
L2 = norm(S2);
L3 = norm(S3);
L4 = norm(S4);
L5 = norm(S5);
L6 = norm(S6);
```