# 3 EKSENLİ ROBOT KOLDA KİNEMATİK HESAPLAMALAR

**Mehmet ŞEN 141222117 ve Mustafa İlker EKMEN 141222115**

Elektrik ve Elektronik Mühendisliği Bölümü, Mühendislik Fakültesi, Selçuk Üniversitesi

**ÖZET**

Bu çalışmada, üç eksenli bir robotta, kinematik hesaplamalar yapılmıştır. Kinematik probleminde, robotun eklem açılarının alabilecegi değerler verilmiş ( θ1 , θ 2 , θ3 , θ 4 ), robotun uç noktasının gideceği yerin koordinatları (x, y, z) bulunmuştur. Yörünge planlaması yapılırken; pozisyonda, hızda ve ivmede süreklilik sağlamak için, matematiksel denklemler kullanılmıştır. Robotun kinematik hesaplamaları ve yörünge planlaması, Matlab kullanılarak gerçekleştirilmiştir. Robotun eklem açılarının, açısal hızlarının ve açısal ivmelerinin zamana göre değişimleri elde edilmiştir. Ayrıca, başlangıç ve hedef noktaları arasında, robotun geçmesi istenen noktalar dikkate alınarak,kinematik hesapları gerçekleştirilmiştir.

## 1.GİRİŞ

Bir robot koluna ilişkin iş planlaması, yörünge planlaması, dinamik ve kontrol problemleri ele alındığı zaman ilk gereksinime duyulan hususlardan biri, bu robot kolun kinematik modelinin oluşturulması ve buna dayanarak gerekli kinematik ilişkilerin elde edilmesidir (Yuan ve Feng, 2014). Robotik alanda kinematik, ileri ve ters olmak üzere ikiye ayrılır. Daha basit, karmaşık olmayan ve kapalı form yöntemleriyle kolaylıkla hesaplanan (Huang ve ark., 2014) ileri kinematik, eklem açıları kullanılarak son elemanın çalışma uzayındaki konumunun ve oryantasyonunun belirlenmesi işlemidir. Çok daha karmaşık, geometrik, iteratif ve cebirsel gibi klasik yöntemlerle çözümün yetersiz kaldığı ters kinematik ise uç elemanın konum bilgilerinden eklem açılarının hesaplanmasıdır (Qiao ve ark., 2010).

## 2.ROBOT KİNEMATİĞİ

Robot uzuvlarının eklemleri ve hareketleri arasındaki ilişki, robot kinematiği ile ifade edilmektedir. Robotik sistemlerde kinematik ifadeler büyük önem arz etmektedir. Robot kinematiği ile robotun hız, kuvvet, tork ve ivme analizleri yapılmaktadır. Bilindiği üzere bir robot, öteleme ve dönme hareketi yapan eklemlerden ve bu eklemleri birbirine bağlayan bağlardan (mafsal) oluşmaktadır. Robot eklemlerinin her birinin konumu, bir önceki ekleme veya bir sonraki ekleme göre ifade edilmektedir. Art arda oluşturulan bu ilişkiye açık kinematik zincir adı verilir. Denavit-Hartenberg [1] ise, bu dönüşüm matrislerini kullanarak herhangi bir koordinat sisteminin oryantasyon ve konum bilgilerini başka bir koordinat sisteminin durumuna göre belirtmiştir.

### 2.1 İleri Yön (Düz) Kinamatiği

Düz kinematik analiz yapılırken, robotun, pozisyon ve oryantasyonunu nasıl belirlediği ve bu belirlemenin eklem (joint) koordinatları cinsinden nasıl yapılacağı gösterilir. Bilindiği gibi herhangi bir manipülatör, birbirine eklemler ile bağlanmış bir seri uzuv (eleman) oluşmuş olarak düşünülebilir. Manipülatörün her bir elemanına bir koordinat sistemi bağlanır ve bu koordinat sistemleri arasındaki bağıl pozisyon ve oryantasyonu homojen transformasyonları kullanarak belirlenir. Bir link (koordinat düzlemi) ile bir sonraki link arasındaki ilişkiyi gösteren homojen transformasyon matrisi, “A” ile gösterilirken, temel dönüşüm matrisi ise “T” ile gösterilir. Ele alınan robotun eksen sayısı kadar homojen dönüşüm matrisi bulunur.

Robotlara düz kinematik analizi yapılmasının temel nedeni, robotun her bir ekseninin belirlenen açılarda hareketi ile robotun uç noktasının ulaştığı konumun koordinatlarını elde etmektir. Bu konum koordinatları elde edilirken Denavit-Hartenberg yöntemi kullanılır. Robotun uç noktasının pozisyonunu belirleyen ana homojen dönüşüm matrisi (2.1)'deki gibi elde edilir.

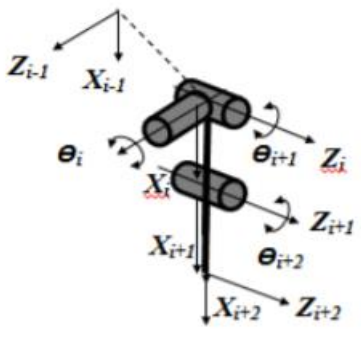
$$ {}_{0}^{N}T = A_1 A_2 \dots A_N \qquad \text{N: Eklem sayısı} \tag{2.1} $$

### 2.1.1. Denavit-Hartenberg Yöntemi

Robotikte kullanılan Denavit-Hartenberg yöntemi ile amaçlanan, robotun mafsallarına yerleştirilen sıralı eksenlerin birbirleri ile olan transformasyonu ile homojen dönüşüm matrisini elde etmek ve her eksenin sahip olduğu bu matrisler ile kinematik analizi yapmaktır. Bu yöntemin uygulamasındaki aşamaları sırasıyla aşağıda ifade edilmiştir.

- Öncelikle eklem eksenlerinin dönme ve kayma yönleri belirlenir. Bu işlem gerçekleştirilirken eklem eksenleri, döner mafsallar için dönme yönü z ekseni, prizmatik mafsallar için kayma yönü z ekseni olarak kabul edilir. Robotun uç noktasına da son eksene paralel bir eksen takımı yerleştirilir. Bunun amacı da, son eksenin transformasyon matrisini de diğer eksenler gibi doğru bir şekilde belirlemektir.
- Merkez koordinat düzleminde eklemin hareket yönüne dik olacak şekilde z ekseni yerleştirilir. Bu z eksenine, 90° açıyla x ekseni yerleştirilir
- z ve x eksenine göre sağ el kuralına göre y ekseni belirlenir. Ancak, ana dönüşüm matrisini oluştururken y ekseninin bir etkisi olmadığı için belirlenmesine gerek yoktur.
- x ekseni, z ekseninden z-1 eksenine geçerken hareketin aynı x ekseni üzerinde olması gerektiği gibi yerleştirilir. Kısaca şöyle ifade edilebilir; eğer, bir önceki eksen takımından bir sonraki eksen takımına geçerken, eksen takımlarının konumları farklı koordinatlarda bulunuyor ise, eksenler arasındaki öteleme bir önceki eksenin x ya da z ekseni doğrultusunda olmalıdır ve bir sonraki eksen takımındaki x ekseni bu kurala göre yerleştirilir.
- Bir seri robotun eklemine koordinat sistemleri yerleştirilirken 1. eksenin dönme yönü z ekseni olarak belirlendikten sonra, genellikle bu eksene x eksenince döndürüldüğünde komşu iki z ekseni çakışacak şekilde bir x ekseni yerleştirilir.

### 2.1.2. Eksen Yerleştirme



Şekil 1.1. Denavit-Hartenberg metodu ile eksen yerleştirme

Eksen takımı yerleştirildikten sonra, robotun uç noktasının koordinatlarını belirleyen dönüşüm matrislerini bulmaya yarayan D-H tablosu oluşturulur. D-H tablosu 4 geometrik büyüklüğe bağlıdır. Bu parametreler, herhangi bir döner veya prizmatik eklemi tam olarak belirler.

### 2.1.3. Homojen Dönüşüm Matrislerinin Bulunması

Robotun her bir ekleminin sahip olduğu parametreler doğrultusunda oluşturacağı dönüşüm matrisi, (2.2)'de verilmiştir.

$$A_i = Rot\ z,\theta_i\ * Trans\ 0,0,d_i\ * Trans\ a,0,0\ * Rot(x,\alpha_i)$$

$$A_i = \begin{bmatrix} Cos\theta_i & -Sin\theta_i & 0 & 0 \\ Sin\theta_i & Cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & Cos\alpha_i & -Sin\alpha_i & 0 \\ 0 & Sin\alpha_i & Cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.2)$$

Matris çarpımı sondan yapılır. Sondaki matrisle bir önceki matris çarpılarak sonuç matris elde edilir.

$$A_i = \begin{bmatrix} Cos\theta_i & -Sin\theta_i & 0 & 0 \\ Sin\theta_i & Cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & Cos\alpha_i & -Sin\alpha_i & 0 \\ 0 & Sin\alpha_i & Cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_i = \begin{bmatrix} Cos\theta_i & -Sin\theta_i & 0 & 0 \\ Sin\theta_i & Cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & Cos\alpha_i & -Sin\alpha_i & 0 \\ 0 & Sin\alpha_i & Cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_i = \begin{bmatrix} Cos\theta_i & -Sin\theta_i Cos\alpha_i & Sin\theta_i.Sin\alpha_i & a_i.Cos\theta_i \\ Sin\theta_i & Cos\theta_i.Cos\alpha_i & -Cos\theta_i.Sin\alpha_i & a_i.Sin\theta_i \\ 0 & Sin\alpha_i & Cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.3)$$

Görüldüğü üzere, bu matrisler çarpılarak, bir eklemin homojen dönüşüm matrisi elde edilir.

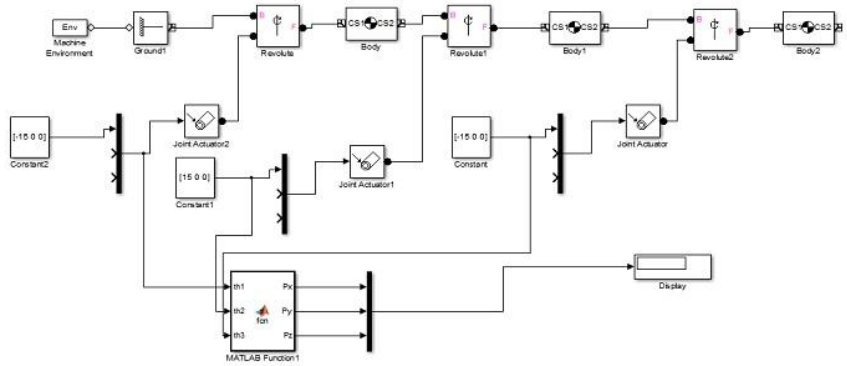
$$A_1 = \begin{bmatrix} Cos\theta_1 & 0 & Sin\theta_1 & 0 \\ Sin\theta_1 & 0 & -Cos\theta_1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.4)$$

$$A_3 = \begin{bmatrix} Cos\theta_3 & -Sin\theta_3 & 0 & a_2 * Cos\theta_3 \\ Sin\theta_3 & Cos\theta_3 & 0 & a_2 * Sin\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

$$A_2 = \begin{bmatrix} Cos\theta_2 & -Sin\theta_2 & 0 & a_1 * Cos\theta_2 \\ Sin\theta_2 & Cos\theta_2 & 0 & a_1 * Sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

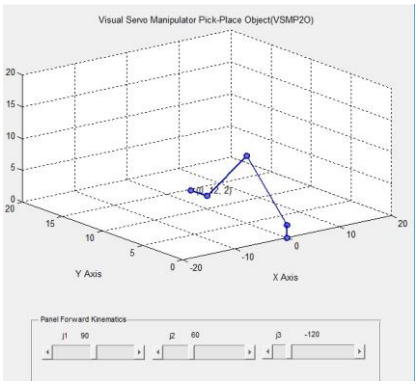
## 3.SONUÇ

Projede yazılımı yapılan Matlab&Simulink yazılımı aşağıda gösterilmektedir.




Şekil 3.1; Simulink Tasarımı

Simulink ile tasarımı yapılan projenin Matlab GUI ile oluşturulan arayüzü aşağıda gösterilmektedir.




Şekil 3.2; Matlab GUI Arayüüzü

**4.EK**

Matlab&Simulink ile gerçekleştirdiğimiz yazılım ortamında Matlab Function bloğunun içerisine gömdüğümüz yazılım;

```
function [Px,Py,Pz]= fcn(th1,th2,th3)

%uzunluklar
l1=50;
l2=50;
l3=70;
%randyan donusumu
rth1=th1*pi/180;
rth2=th2*pi/180;
rth3=th3*pi/180;

T_0_1=[cos(rth1) -sin(rth1) 0 0;sin(rth1) cos(rth1) 0 0;0 0 1
0 ;0 0 0 1];

T_1_2=[cos(rth2) -sin(rth2) 0 l1;sin(rth2) cos(rth2) 0 0;0 0 1
0 ;0 0 0 1];

T_2_3=[cos(rth3) -sin(rth3) 0 l2;sin(rth3) cos(rth3) 0 0;0 0 1
0 ;0 0 0 1];

T_3_4=[1 0 0 l3;0 1 0 0;0 0 1 0 ;0 0 0 1];

T_0_4=T_0_1*T_1_2*T_2_3*T_3_4;
Px=T_0_4(1,4);
Py=T_0_4(2,4);
Pz=T_0_4(3,4);
```

Matlab GUI ortamında oluşturduğumuz arayüzün kodları aşağıdadır.

```
function varargout = GUI_VSMP2O(varargin)// Fonksiyon yazıldı


gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @GUI_VSMP2O_OpeningFcn,
...
                   'gui_OutputFcn',  @GUI_VSMP2O_OutputFcn,
...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);//Giriş-Çıkış
Dosyaları yazıldı.
if nargin && ischar(varargin{1})
```

```
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State,
varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function GUI_VSMP2O_OpeningFcn(hObject, eventdata, handles,
varargin)

handles.output = hObject;


guidata(hObject, handles);


jinit=[90;60;-120];
assignin('base','jinit',jinit);

axes(handles.axes1);
FKdraw(jinit(1,1),jinit(2,1),jinit(3,1));
ax_properties = get(gca);
assignin('base','pov',ax_properties.View);


function varargout = GUI_VSMP2O_OutputFcn(hObject, eventdata,
handles)



varargout{1} = handles.output;
%% FK
function pushbutton1_Callback(hObject, eventdata, handles)
axes(handles.axes1);
jinit=evalin('base','jinit');
FKdraw(jinit(1,1),jinit(2,1),jinit(3,1));
set( handles.text1,'String', num2str(90) );
set( handles.text2,'String', num2str(60) );
set( handles.text3,'String', num2str(-120) );

function slider1_Callback(hObject, eventdata, handles)
val2=str2num(get(handles.text2,'String'));
val3=str2num(get(handles.text3,'String'));
val1 = get(hObject,'Value') ;
val=[val1;val2;val3];
assignin('base','val',val);
set( handles.text1,'String', num2str(val1,3) );
axes(handles.axes1);
```

```
FKdraw(val1,val2,val3);
view(evalin('base','pov'));


function slider1_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor',[.9 .9 .9]);
end

function slider2_Callback(hObject, eventdata, handles)
val1=str2num(get(handles.text1,'String'));
val3=str2num(get(handles.text3,'String'));
val2 = get(hObject,'Value') ;
val=[val1;val2;val3];
assignin('base','val',val);
set(handles.text2,'String', num2str(val2,3));
axes(handles.axes1);
FKdraw(val1,val2,val3);
view(evalin('base','pov'));

function slider2_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

function slider3_Callback(hObject, eventdata, handles)
val1=str2num(get(handles.text1,'String'));
val2=str2num(get(handles.text2,'String'));
val3 = get(hObject,'Value') ;
val=[val1;val2;val3];
assignin('base','val',val);
set( handles.text3,'String', num2str(val3,3) );
axes(handles.axes1);
FKdraw(val1,val2,val3);
view(evalin('base','pov'));


function slider3_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

**5.KAYNAKLAR**

[1] Denavit, J., Hartenberg R. S., June 1955. A kinematic notation for Lower-pair mechanisms based on matrices. ASME Jappl. Mechan. pp. 215-221.

Altan, A.,RPR (Dönel Prizmatik Dönel) Eklem Yapısına Sahip Bir Robotun Dinamik Denklemlerinin Vektör-Matris Formda Türetilmesi, Hitit Üniversitesi.

Bingül, Z., Küçük, S., 2005. Robot Tekniği I, Birsen Yayınevi, pp.104-200, pp.13-15.