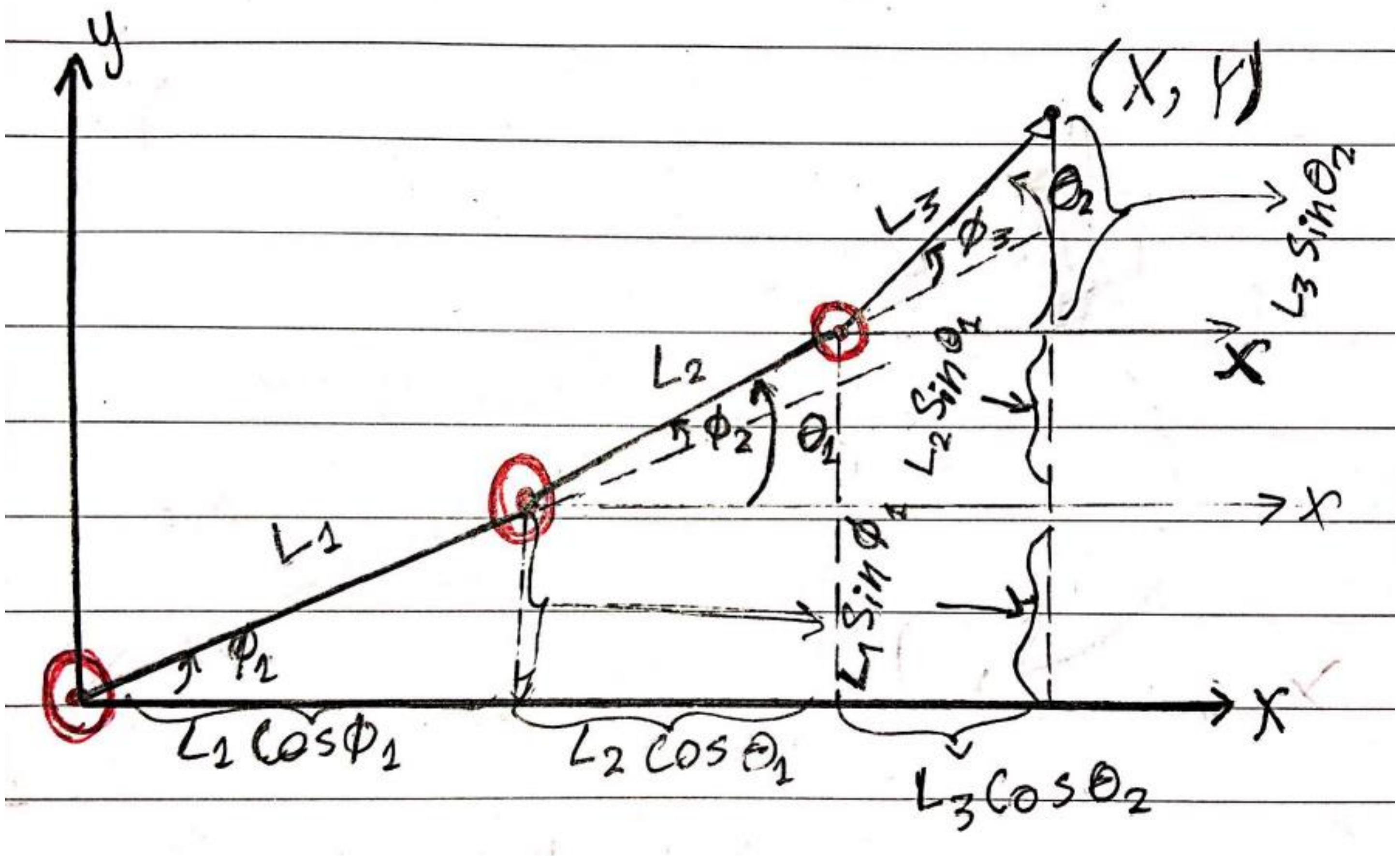
Forward Kinematics for 6 DoF Robotic Arm

By Mohammed Alsaggaf

Presentation Flow

- Why not using the old method?
- Denavit–Hartenberg Parameters & Rules
- Multiply Matrices in MATLAB
- Arduino Code
- Check Solution
- Double Check Solution

Forward Kinematics for 3 DoF Robotic Arm



Denavit–Hartenberg Parameters

a_i : The length distance from z_i to z_{i+1} measured along z_i

α_i: The twist angle between z_i and z_{i+1} measured about x_i

d_i: The offset distance from x_i to x_{i+1} measured along z_i

θ_i: The angle between x_i and x_{i+1} measured about z_i

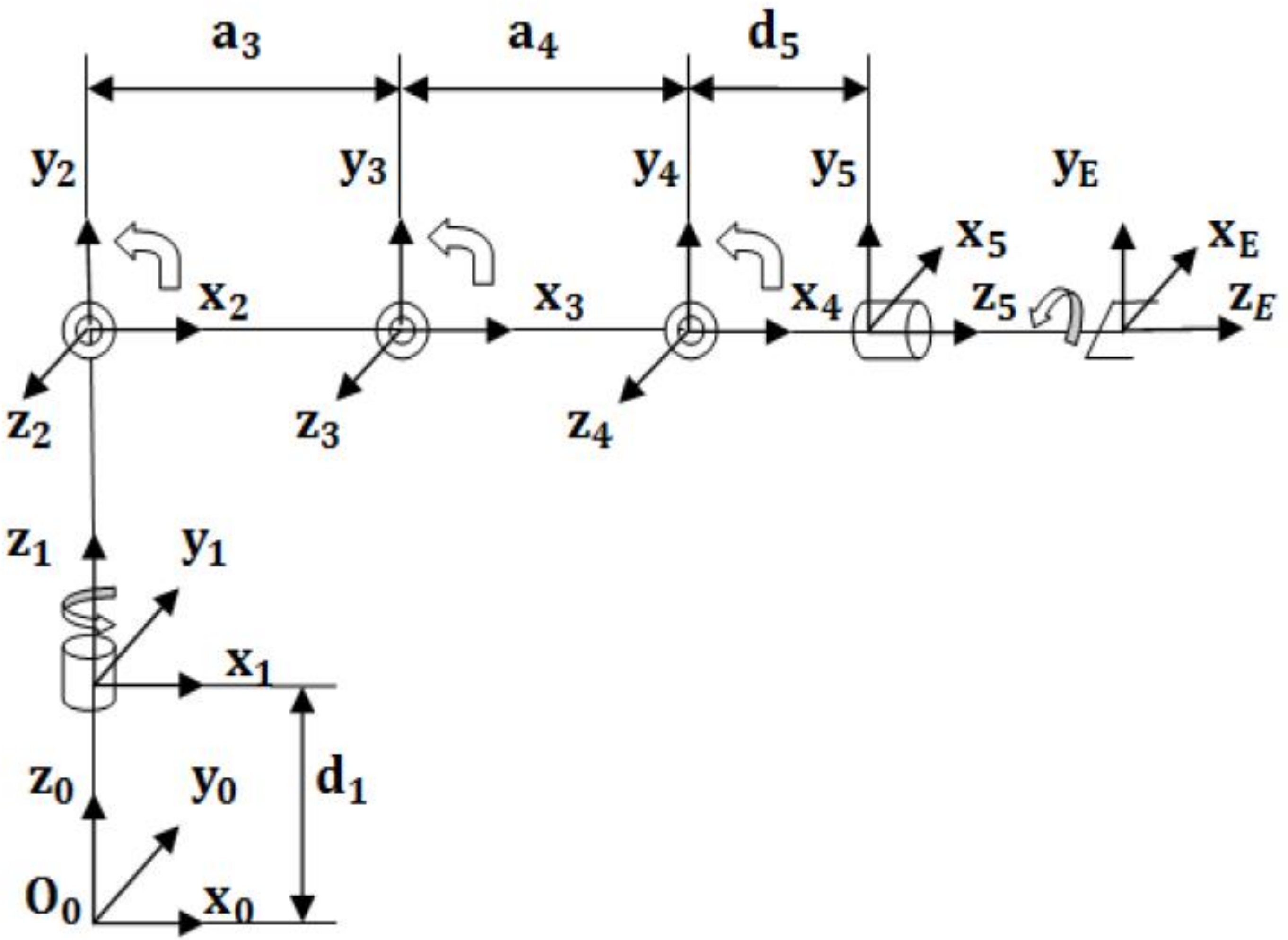
$$Ai = \begin{bmatrix} C\theta i & -S\theta iC\alpha i & S\theta iS\alpha i & aiC\theta i \\ S\theta i & C\theta iC\alpha i & -C\theta iS\alpha i & aiS\theta i \\ 0 & S\alpha i & C\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

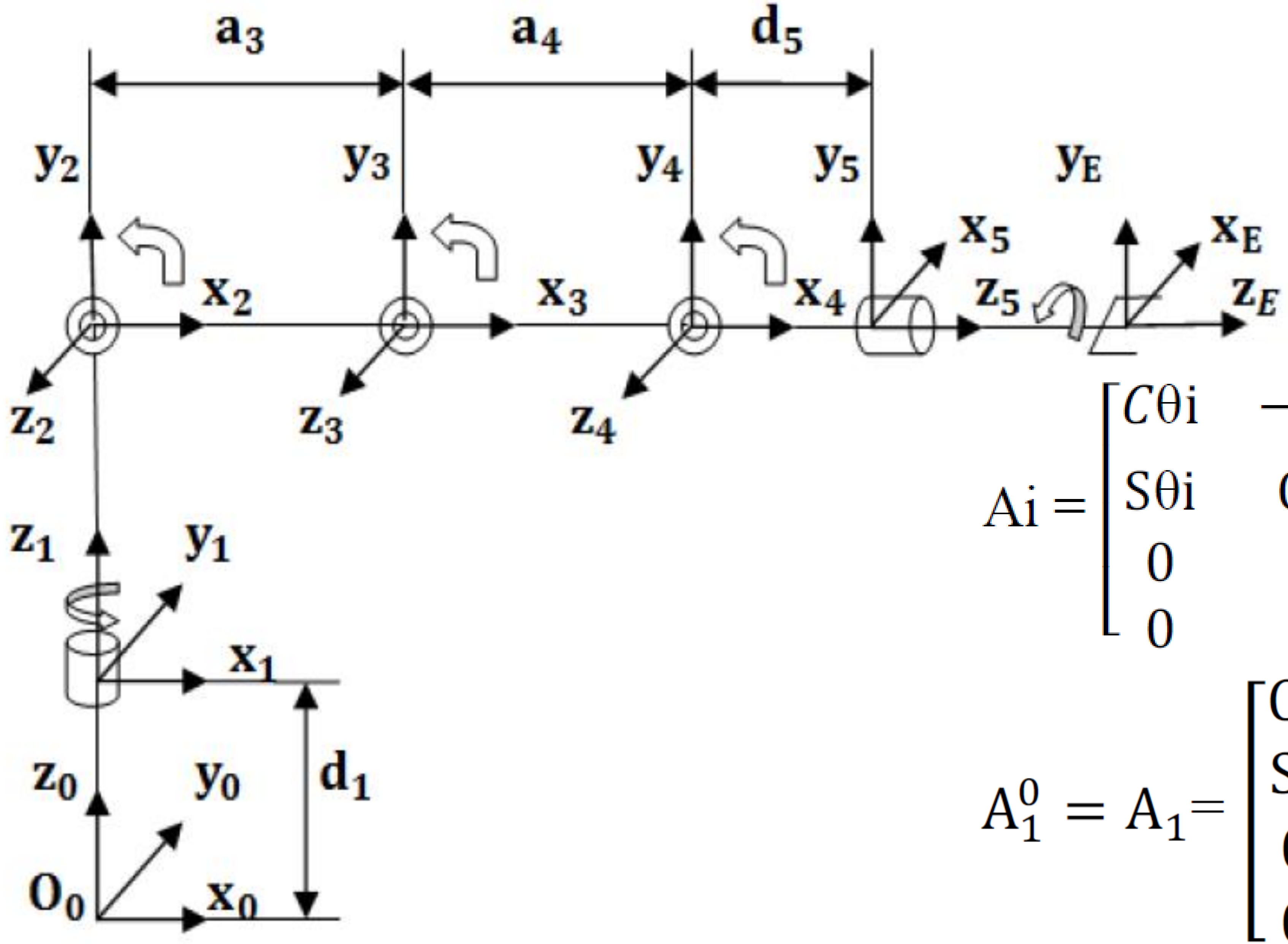
Denavit–Hartenberg Rules

I. Z in the axis of rotation

II. X_i axis is perpendicular to Z_i and Z_{i-1} and intersects Z_{i-1}

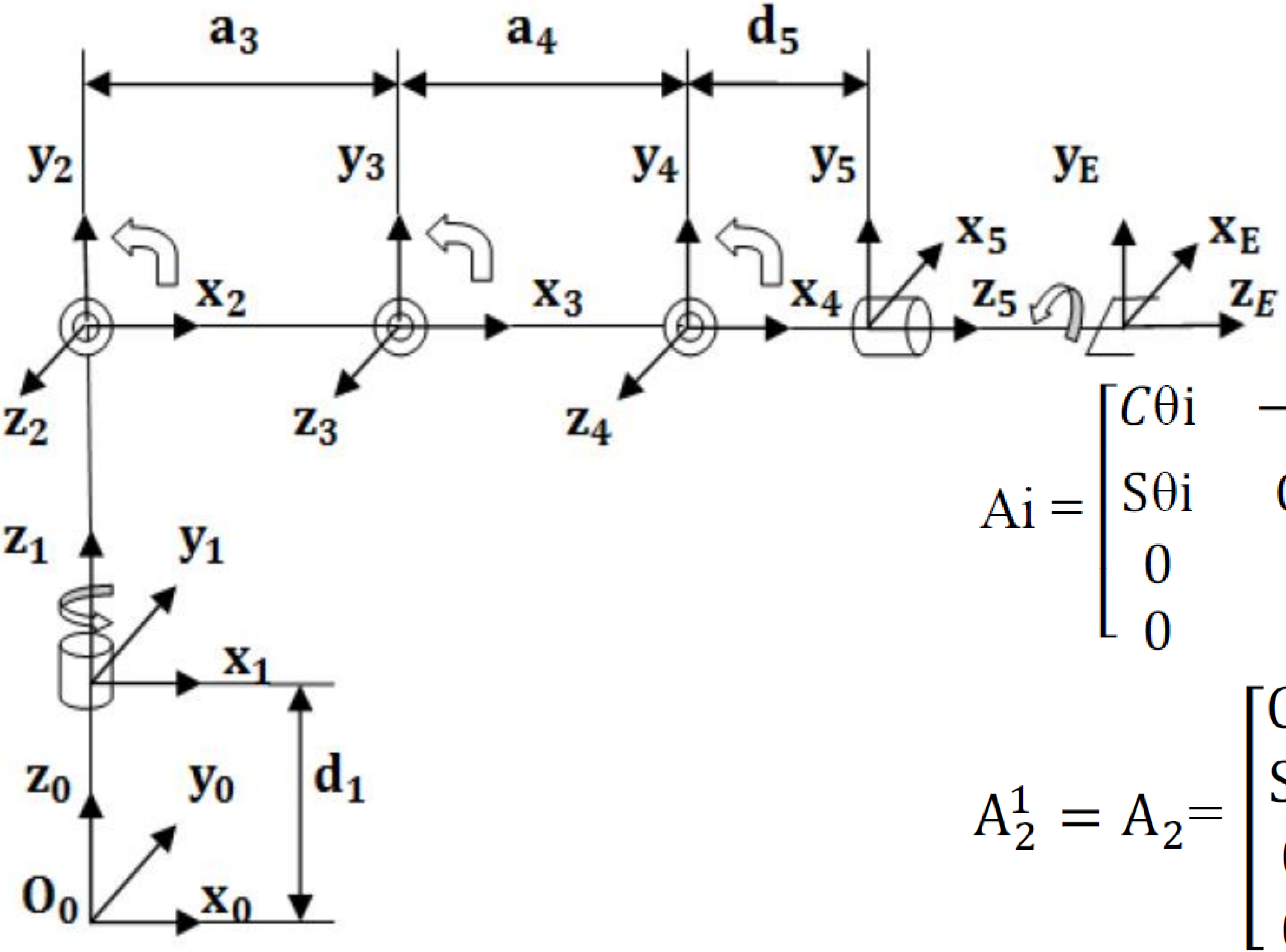
III. Y direction is found by the right hand rule





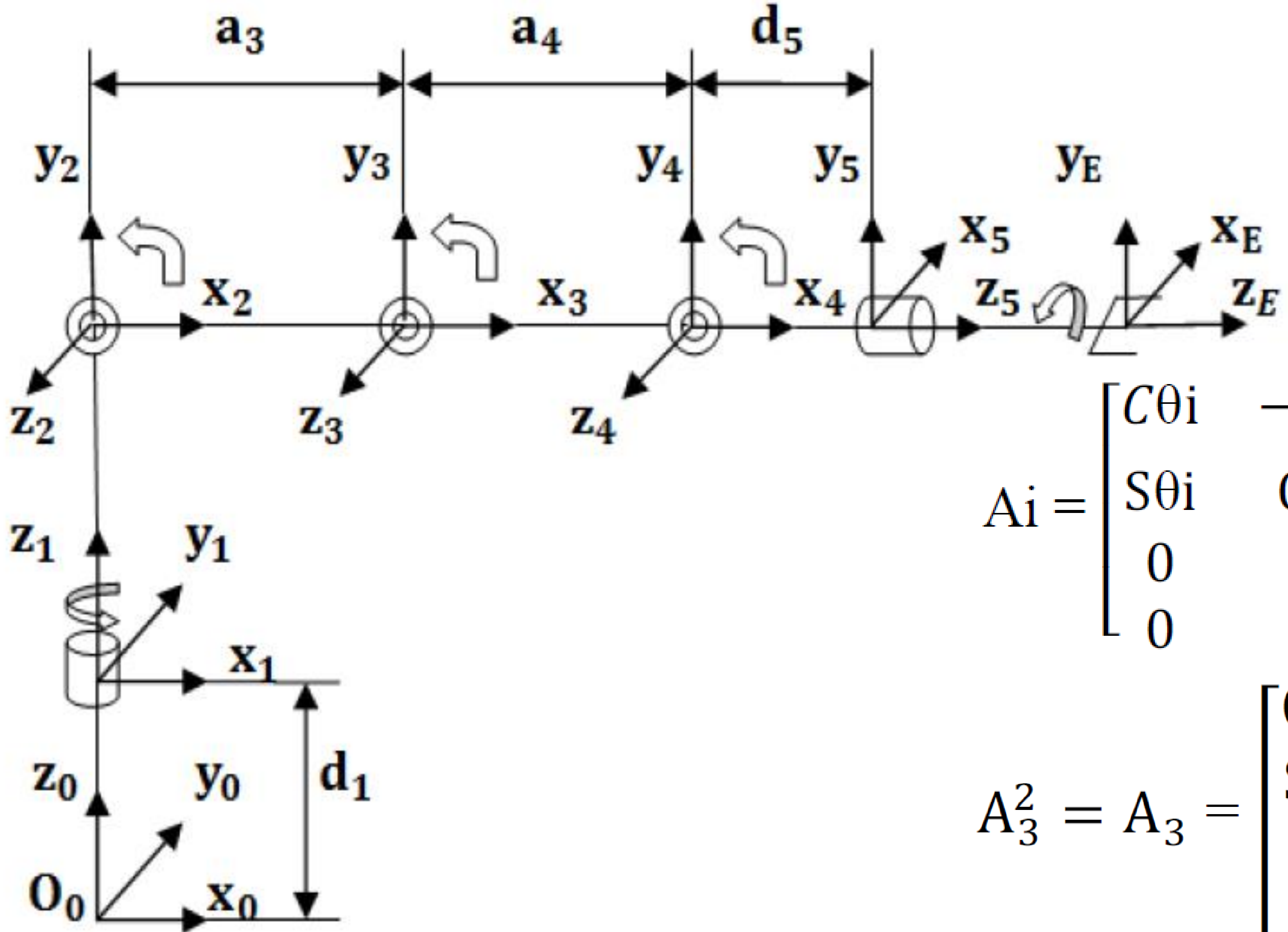
$$Ai = \begin{bmatrix} C\theta i & -S\theta iC\alpha i & S\theta iS\alpha i & aiC\theta i \\ S\theta i & C\theta iC\alpha i & -C\theta iS\alpha i & aiS\theta i \\ 0 & S\alpha i & C\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_1^0 = A_1 = \begin{bmatrix} C_1 & -S_1 & 0 & 0 \\ S_1 & C_1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



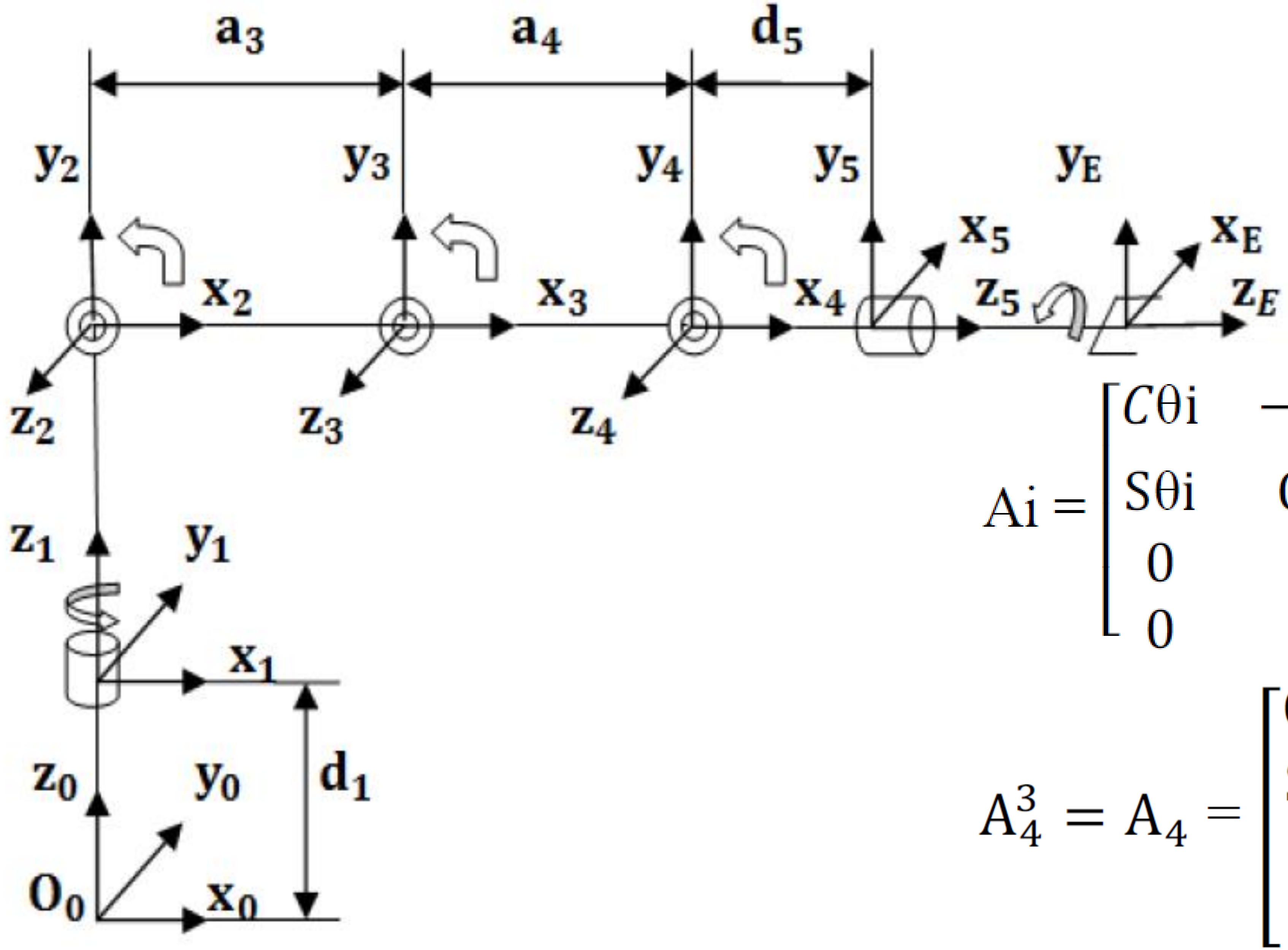
$$
Ai = \begin{bmatrix} C\theta i & -S\theta iC\alpha i & S\theta iS\alpha i & aiC\theta i \\ S\theta i & C\theta iC\alpha i & -C\theta iS\alpha i & aiS\theta i \\ 0 & S\alpha i & C\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
A_2^1 = A_2 = \begin{bmatrix} C_2 & 0 & S_2 & 0 \\ S_2 & 0 & -C_2 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$



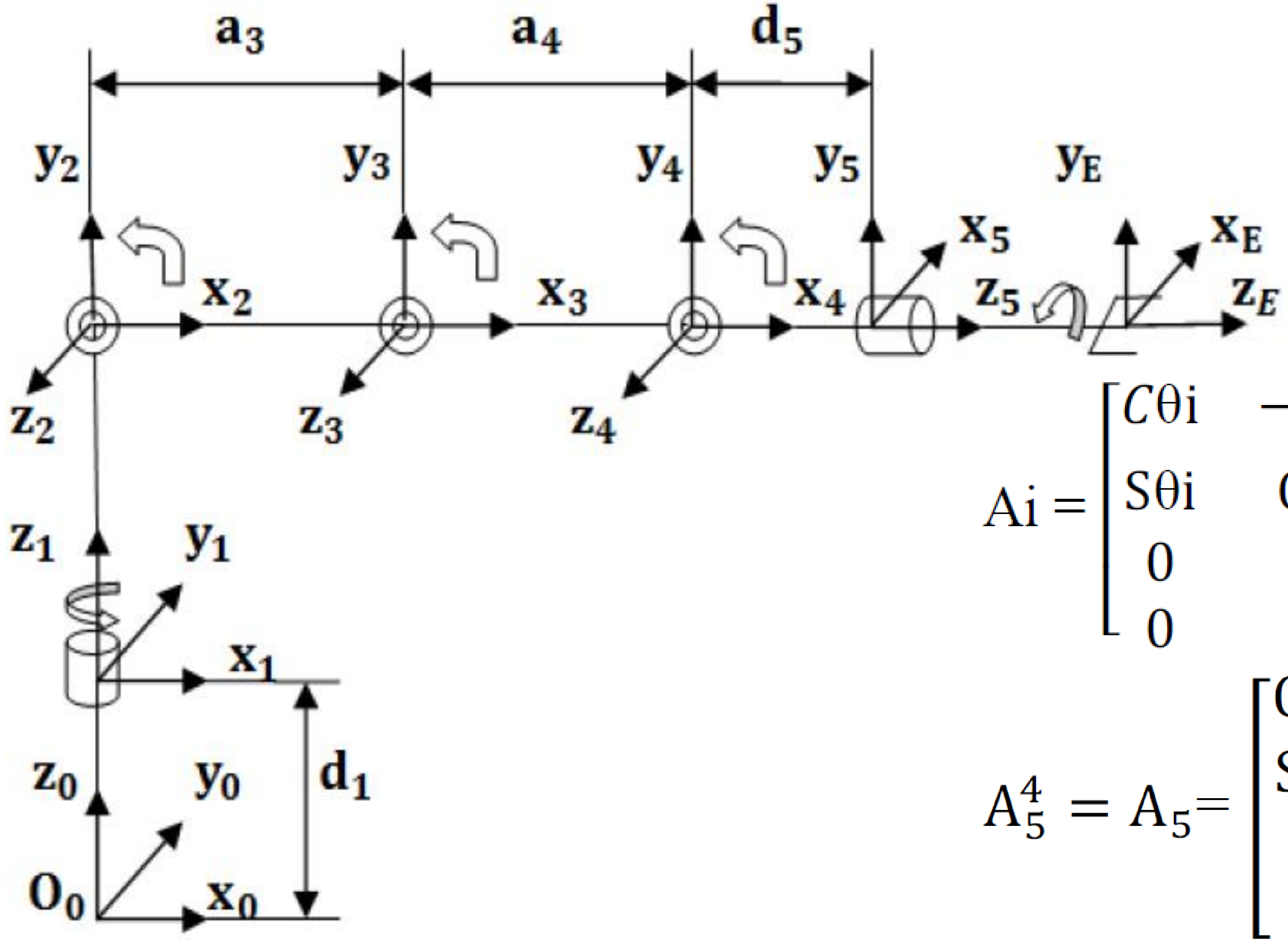
$$Ai = \begin{bmatrix} C\theta i & -S\theta iC\alpha i & S\theta iS\alpha i & aiC\theta i \\ S\theta i & C\theta iC\alpha i & -C\theta iS\alpha i & aiS\theta i \\ 0 & S\alpha i & C\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3^2 = A_3 = \begin{bmatrix} C_3 & -S_3 & 0 & a_3C_3 \\ S_3 & C_3 & 0 & a_3S_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



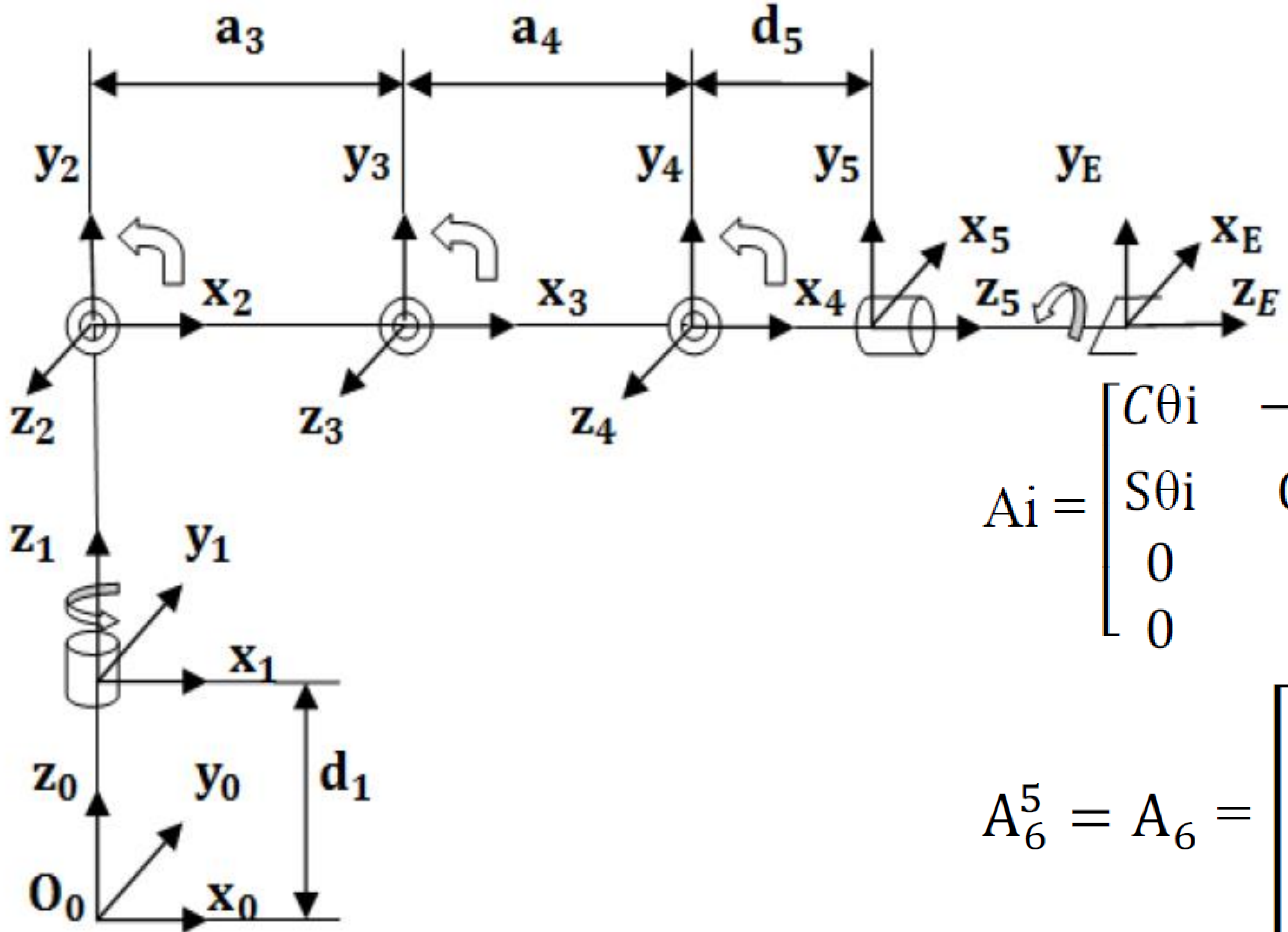
$$Ai = \begin{bmatrix} C\theta i & -S\theta iC\alpha i & S\theta iS\alpha i & aiC\theta i \\ S\theta i & C\theta iC\alpha i & -C\theta iS\alpha i & aiS\theta i \\ 0 & S\alpha i & C\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_4^3 = A_4 = \begin{bmatrix} C_4 & -S_4 & 0 & a_4C_4 \\ S_4 & C_4 & 0 & a_4S_4 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$$
Ai = \begin{bmatrix} C\theta i & -S\theta iC\alpha i & S\theta iS\alpha i & aiC\theta i \\ S\theta i & C\theta iC\alpha i & -C\theta iS\alpha i & aiS\theta i \\ 0 & S\alpha i & C\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

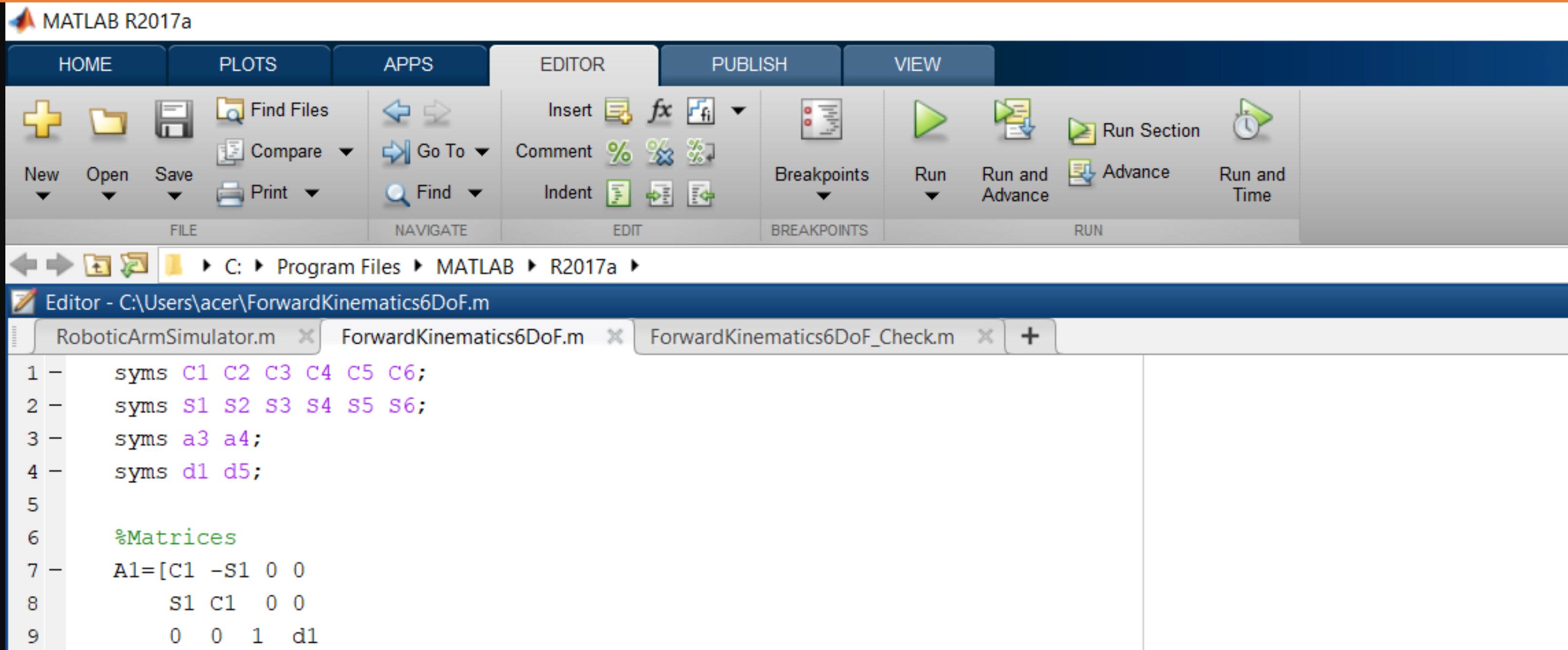
$$
A_5^4 = A_5 = \begin{bmatrix} C_5 & 0 & -S_5 & 0 \\ S_5 & 0 & C_5 & 0 \\ 0 & -1 & 0 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$



$$Ai=\begin{bmatrix} C\theta i & -S\theta iC\alpha i & S\theta iS\alpha i & aiC\theta i \\ S\theta i & C\theta iC\alpha i & -C\theta iS\alpha i & aiS\theta i \\ 0 & S\alpha i & C\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

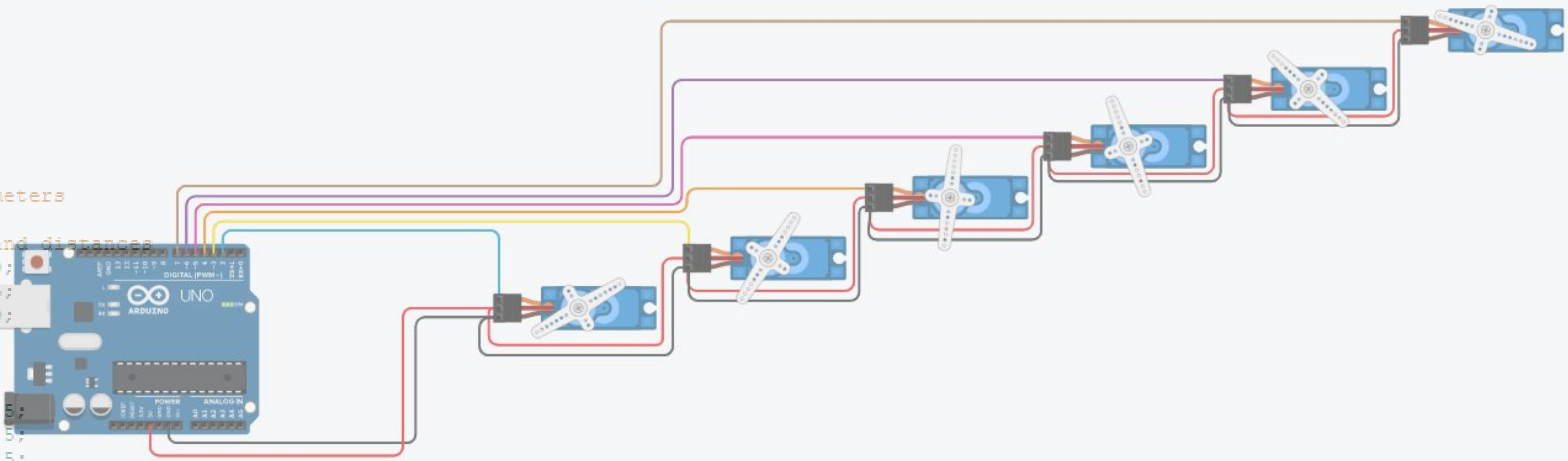
$$A_6^5=A_6=\begin{bmatrix} C_6 & -S_6 & 0 & 0 \\ S_6 & C_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Multiply Matrices



Arduino Code

```
#include <Servo.h>
Servo S1;
Servo S2;
Servo S3;
Servo S4;
Servo S5;
Servo S6;


//Given Parameters

  //Lenghts and distances
  float a3=20;
  float a4=15;
  float d1=10;
  float d5=5;

  //Angles
  float t1=0.5;
  float t2=0.5;
  float t3=0.5;
  float t4=0.5;
  float t5=0.5;
  float t6=0.5;
  float pi=3.141592654; //Pi, the mathmatical constant
//Known Orientation Parameters
float nx; float ny; float nz;
float ox; float oy; float oz;
float ax; float ay; float az;


//Unknown Position Parameters
float px; //End effector's X coordinate
float py; //End effector's Y coordinate
float pz; //End effector's Z coordinate


void setup()
{
  //Attaching Servo Motors to the digital pins
    S1.attach(2);
    S2.attach(3);
```



Thanks for listening

The end of the presentation

$$T_{13} = T_{12} * A_3 = \begin{bmatrix} C_{123} - S_{12}C_3 \\ -S_{13}C_2 - C_1S_{23} \\ 0 \\ 0 \end{bmatrix}$$

Forward kinematics for 6 DoF

* Draw the modeling block diagram with considering the Denavit-Hartenberg convention rules
    * D-H convention rules:
        1- Z_{i-1} is the axis of ~~revolution~~ rotation. of
        2- set X_i axis perpendicular to Z_i and Z_{i-1} and intercects Z_{i-1}.
        3- Find Y axis direction by the Right-Hand rule.
* Find the D-H parameters and substitute them in the transformation matrix.
* Find the rotation matrix R^0_n by multiplying each transformation matrix, e.g. $R^0_n = R^0_1 * R^0_2 \dots R^{n-1}_n$.
* Get the end-effectors position and rotation angle from the total transformation matrix.

References

[1] M. Alsaggaf, "6 DoF Robotic Arm Forward Kinematics," TinkerCAD, 28 07 2020. [Online]. Available: https://www.tinkercad.com/things/56EDMEckwu9 .

[2] A. Sodemann, "Robotics 1 U1 (Kinematics) S3 (Rotation Matrices) P4 (6-DoF Example and Error Checking)," YouTube, 27 08 2017. [Online]. Available: https://youtu.be/KslFPohHkxA . [Accessed 24 07 2020].

[3] milfordrobotics, "Forward and Inverse Kinematics Part 1," YouTube, 03 08 2011. [Online]. Available: https://youtu.be/VjsuBT4Npvk . [Accessed 24 07 2020].

[4] milfordrobotics, "Forward and Inverse Kinematics Part 2," YouTube, 04 08 2011. [Online]. Available: https://youtu.be/3ZcYSKVDlOc . [Accessed 24 07 2020].

[5] "Create Symbolic Numbers, Variables, and Expressions," MathWorks, [Online]. Available: https://www.mathworks.com/help/symbolic/create-symbolic-numbers-variables-and-expressions.html . [Accessed 27 07 2020].