CHAPTER 8

ITERATIVE INVERSE KINEMATICS

8.1 STAGE V: GENERATION OF OBJECTIVE FUNCTION

8.1.1 Introduction

In order to obtain the inverse kinematics solution, an iterative technique is used. In this technique, an initial guess for the joint parameters is made. Forward analysis is performed to determine the position and orientation of the end effector for the selected joint parameters. The objective function is to minimize the difference between the position and orientation calculated from the forward analysis and goal position and orientation, which is represented as an error.

8.1.2 Objective Function

An objective function in terms of joint, D-H parameters $F(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5)$ is formulated, and search techniques are used to obtain the set of input parameters to minimize $F$. The objective function is the sum of the squares of the position error and orientation errors.

Thus the objective function could be written as

$$F(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5) = (n_{xp} - n_{xg})^2 + (n_{yp} - n_{yg})^2 + (n_{zp} - n_{zg})^2 +$$

$$+ (o_{xp} - o_{xg})^2 + (o_{yp} - o_{yg})^2 + (o_{zp} - o_{zg})^2$$
\[
(a_{xp} - a_{xg})^2 + (a_{yp} - a_{yg})^2 + (a_{zp} - a_{zg})^2 + \\
(p_{xp} - p_{xg})^2 + (p_{yp} - p_{yg})^2 + (p_{zp} - p_{zg})^2
\]

Where the subscript ‘p’ refers to a position and orientation calculated by performing a forward kinematics analysis using the current design parameters, and the subscript ‘g’ refers to the goal position value specified at the start of the inverse kinematics analysis problem. The iterative solution yields correct results when the specified end effector position and orientation is in the reachable workspace of the SCORBOT, because sets of joint angles will exist that will position and orient the end effector as desired. The objective function will thereby attain its optimal value of zero.

The optimal solution for this problem is to obtain a set of values for the design parameters that cause the objective function to equal zero.

8.1.3 Methodology for Complete Iterative Solution

The complete iterative solution for the inverse kinematics is obtained as follows: The goal position is assumed initially. The input variables for inverse kinematics problem are twelve goal position and orientation vectors.

The goal position and orientation are given as a matrix

\[
\begin{bmatrix}
n_{xg} & o_{xg} & a_{xg} & p_{xg} \\
n_{yg} & o_{yg} & a_{yg} & p_{yg} \\
n_{zg} & o_{zg} & a_{zg} & p_{zg} \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

In the first phase, \( \theta_1 \) starts from -155 and adding +1 in the next iteration keeping \( \theta_2 = -35, \theta_3 = -130, \theta_4 = -130, \) and \( \theta_5 = -570 \) constant. It goes up to 310 iterations (Range of \( \theta_1 \) Minimum -155 to Maximum +155). In phase
two, $311^{th}$ iteration starts with $\theta_1$ again from -155 and adding +1 in the next iteration keeping $\theta_2 = -34$, $\theta_3 = -130$, $\theta_4 = -130$, and $\theta_5 = -570$ constant. It goes up to 310 iterations and so on. If the objective function attains the zero value the iterative process will stop and solution for the problem will be displayed in the front panel.

### 8.1.4 Methodology for Partial Iterative Solution

The partial iterative solution for the inverse kinematics obtained is explained as follows: The goal position is assumed initially. The input variables for inverse kinematics problem are twelve goal position and orientation vectors.

The goal position and orientation are given as a matrix

$$
\begin{bmatrix}
  n_x & o_x & a_x & p_x \\
  n_y & o_y & a_y & p_y \\
  n_z & o_z & a_z & p_z \\
  0 & 0 & 0 & 1
\end{bmatrix}
$$

The following values are calculated using equations

$$
\begin{align*}
\theta_1 &= \text{atan2} \ (p_y, p_x) \\
\theta_5 &= \text{atan2} \ (-\sin(\theta_1) \ n_x + \cos(\theta_1) \ n_y, -\sin(\theta_1) \ o_x + \cos(\theta_1) \ o_y) \\
\theta_{234} &= \text{atan2} \ (-a_{zg}, \cos(\theta_1) \ a_{xg} + \sin(\theta_1) \ a_{yg})
\end{align*}
$$

If we assume $\theta_2$ and $\theta_3$ iteratively as explained below we can calculate $\theta_4$ using the following relation

$$
\theta_4 = \theta_{234} - \theta_2 - \theta_3
$$
In the first phase, \( \theta_2 \) starts from -35 and adding +1 in the next iteration keeping \( \theta_3 = -130 \) constant. \( \theta_1, \theta_4, \theta_5 \) are calculated from the equations. It goes up to 165 iterations (Range of \( \theta_2 \) Minimum -35 to Maximum +130). In phase two, 166\(^{th}\) iteration starts with \( \theta_2 \) again from -35 and adding +1 in the next iteration keeping \( \theta_3 = -129 \) constant. \( \theta_1, \theta_4, \theta_5 \) are calculated from the equations. It goes up to 165 iterations and so on. If the objective function attains the zero value the iterative process will stop and solution for the problem will be displayed in the front panel.

**8.1.5 Total Number of Iterations and Phases (CIKSM)**

The complete algorithm runs with many phases. The total number of phases expected maximum of 97812000. Total number of iterations per phase is 310. The number of generations required for solving the problem is a maximum of 3.0327\times10^{10}. But depending on the goal position the total number of iterations will vary.

**Phase 1:**

Iteration 1: \( \theta_1 = -155, \theta_2 = -35, \theta_3 = -130, \theta_4 = -130, \theta_5 = -570 \)

Iteration 2: \( \theta_1 = -154, \theta_2 = -35, \theta_3 = -130, \theta_4 = -130, \theta_5 = -570 \)

Iteration 3: \( \theta_1 = -153, \theta_2 = -35, \theta_3 = -130, \theta_4 = -130, \theta_5 = -570 \)

Iteration 4: \( \theta_1 = -152, \theta_2 = -35, \theta_3 = -130, \theta_4 = -130, \theta_5 = -570 \)

Iteration 5: \( \theta_1 = -151, \theta_2 = -35, \theta_3 = -130, \theta_4 = -130, \theta_5 = -570 \)

........and so on.

Iteration 310: \( \theta_1 = 155, \theta_2 = -35, \theta_3 = -130, \theta_4 = -130, \theta_5 = -570 \)
Phase 2:

Iteration 311: $\theta_1 = -155$, $\theta_2 = -34$, $\theta_3 = -130$, $\theta_4 = -130$, $\theta_5 = -570$

Iteration 312: $\theta_1 = -154$, $\theta_2 = -34$, $\theta_3 = -130$, $\theta_4 = -130$, $\theta_5 = -570$

Iteration 313: $\theta_1 = -153$, $\theta_2 = -34$, $\theta_3 = -130$, $\theta_4 = -130$, $\theta_5 = -570$

Iteration 314: $\theta_1 = -152$, $\theta_2 = -34$, $\theta_3 = -130$, $\theta_4 = -130$, $\theta_5 = -570$

Iteration 315: $\theta_1 = -151$, $\theta_2 = -34$, $\theta_3 = -130$, $\theta_4 = -130$, $\theta_5 = -570$

……..and so on.

Iteration 620: $\theta_1 = 155$, $\theta_2 = -34$, $\theta_3 = -130$, $\theta_4 = -130$, $\theta_5 = -570$

Phase 3:……..and so on.

.
.
.

Last Phase:

Last Iteration: $\theta_1 = 155$, $\theta_2 = 130$, $\theta_3 = 130$, $\theta_4 = 130$, $\theta_5 = 570$

8.1.6 Total Number of Iterations and Phases (PIIKM)

The complete algorithm runs with many phases. The total number of phases expected maximum of 260. Total number of iterations per phase is 165. The number of generations required for solving the problem is a maximum of 42900. But depends on the goal position the total number of iterations will vary.
Phase 1:

Iteration 1: $\theta_2=-35$, $\theta_3=-130$, $\theta_1=10$, $\theta_4=195$, $\theta_5=10$

Iteration 2: $\theta_2=-34$, $\theta_3=-130$, $\theta_1=10$, $\theta_4=194$, $\theta_5=10$

Iteration 3: $\theta_2=-33$, $\theta_3=-130$, $\theta_1=10$, $\theta_4=193$, $\theta_5=10$

Iteration 4: $\theta_2=-32$, $\theta_3=-130$, $\theta_1=10$, $\theta_4=192$, $\theta_5=10$

Iteration 5: $\theta_2=-31$, $\theta_3=-130$, $\theta_1=10$, $\theta_4=191$, $\theta_5=10$

……..and so on.

Iteration 165: $\theta_2=130$, $\theta_3=-130$, $\theta_1=10$, $\theta_4=30$, $\theta_5=10$

Phase 2:

Iteration 166: $\theta_2=-35$, $\theta_3=-129$, $\theta_1=10$, $\theta_4=194$, $\theta_5=10$

Iteration 167: $\theta_2=-34$, $\theta_3=-129$, $\theta_1=10$, $\theta_4=193$, $\theta_5=10$

Iteration 168: $\theta_2=-33$, $\theta_3=-129$, $\theta_1=10$, $\theta_4=192$, $\theta_5=10$

Iteration 169: $\theta_2=-32$, $\theta_3=-129$, $\theta_1=10$, $\theta_4=191$, $\theta_5=10$

Iteration 170: $\theta_2=-31$, $\theta_3=-129$, $\theta_1=10$, $\theta_4=190$, $\theta_5=10$

……..and so on.

Iteration 330: $\theta_2=130$, $\theta_3=-129$, $\theta_1=10$, $\theta_4=29$, $\theta_5=10$

Phase 3:……..and so on.
Phase 260:

Iteration 42900: $\theta_2=130, \theta_3=130, \theta_4=10, \theta_5=-230, \theta_6=10$

But for the assumed goal position (in Stage IV) the iterative process will stop at phase 140 and at 23145th iteration as the objective function FC attains zero.

8.1.7 Programming Algorithm for Complete Iteration

By this approach the algorithm is written and successfully executed in LabVIEW. The twelve input variables of the goal position ($n_{xg}, n_{yg}, n_{zg}, o_{xg}, o_{yg}, o_{zg}, a_{xg}, a_{yg}, a_{zg}, p_{xg}, p_{yg}$ and $p_{zg}$) are used in the first formula node. In the second formula node $\theta_1, \theta_2, \theta_3, \theta_4$ and $\theta_5$ are continuously varying and objective function is calculated dynamically for the respective assigned values of succeeded iterations. The twelve outputs of the second formula node (position and orientation vector components of the dynamic values of each iteration) are given to the first formula node as inputs. The objective function is checked for optimality. If the objective function is an optimized one, the iteration will stop and the required solution will be displayed.

8.1.8 Programming Algorithm for Partial Iteration

By this approach the algorithm is written and successfully executed in LabVIEW. The twelve input variables of the goal position ($n_{xg}, n_{yg}, n_{zg}, o_{xg}, o_{yg}, o_{zg}, a_{xg}, a_{yg}, a_{zg}, p_{xg}, p_{yg}$ and $p_{zg}$) are used to calculate $\theta_1$, $\theta_2$, $\theta_3$, $\theta_4$ in the first formula node. In the second formula node $\theta_2, \theta_3$ are continuously varying and $\theta_4$ is calculated dynamically for the respective assigned values of succeeded iterations ($\theta_4=\theta_{234}-\theta_2-\theta_3$). The 12 outputs of the second formula node (position and orientation vector components of the dynamic values of each iteration) are given to the first formula node as inputs. The objective function
is checked for optimizing. If the objective function is an optimized one, the iteration will stop and the required solution will be displayed.

8.2 ITERATIVE SOLUTION

8.2.1 Complete Iteration IK Method (CIIKM)

In this method, the goal position is set first before the programme is run as shown in Figure 8.1. The ranges of joints listed in Table 4.2 are as follows. T1: Base (±155), T2: Shoulder (-35 to +130), T3: Elbow (±130), T4: Wrist Pitch (±130), T5: Wrist Roll (±570).

```
Q1=q1*(227/180);
Q2=q2*(227/180);
Q3=q3*(227/180);
Q4=q4*(227/180);
Q5=q5*(227/180);

nxg=cos(Q1)*sin(Q2+Q3+Q4)*cos(Q5)-sin(Q1)*sin(Q5);
nyg=cos(Q1)*sin(Q2+Q3+Q4)*sin(Q5)+cos(Q1)*sin(Q5);
ngx=cos(Q2+Q3+Q4)*cos(Q5);
ngy=cos(Q2+Q3+Q4)*sin(Q5)-sin(Q1)*cos(Q5);
ngz=cos(Q2+Q3+Q4)*sin(Q5);
ngz=cos(Q1)*cos(Q2+Q3+Q4);
ngz=cos(Q1)*cos(Q2+Q3+Q4);
ngz=cos(Q1)*cos(Q2+Q3+Q4);
ngz=cos(Q1)*cos(Q2+Q3+Q4);
ngz=cos(Q1)*cos(Q2+Q3+Q4);
ngz=cos(Q1)*cos(Q2+Q3+Q4);
ngz=cos(Q1)*cos(Q2+Q3+Q4);
ngz=cos(Q1)*cos(Q2+Q3+Q4);
```

Figure 8.1 CIIKM Formula Node to get Vectors of Goal Position

The five joint parameters T1, T2, T3, T4, and T5 (θ1, θ2, θ3, θ4, and θ5 are indicated as T1, T2, T3, T4, and T5 respectively in the LabVIEW model) are started from minimum range up to maximum range set as iterative trial. To do this five loops are used in the model as shown in Figure 8.2.
From the five loops the iterative parameter values are given as input for the next formula node as shown in Figure 8.3. All the components are substituted in the objective function for obtaining inverse kinematics solution (Figure 8.4).

\[
F(t_1, t_2, t_3, t_4, t_5) = (n_{xp}-n_{xg})^2+(n_{yp}-n_{yg})^2+(n_{zp}-n_{zg})^2+(o_{xp}-o_{xg})^2+(o_{yp}-o_{yg})^2+(o_{zp}-o_{zg})^2+(a_{xp}-a_{xg})^2+(a_{yp}-a_{yg})^2+(a_{zp}-a_{zg})^2+(p_{xp}-p_{xg})^2+(p_{yp}-p_{yg})^2+(p_{zp}-p_{zg})^2
\]

A case structure is used to stop the programme, when the objective function attained minimum value as shown in Figure 8.5.
Figure 8.3 CIIKM Formula node for Iterative Joint Parameters

```matlab
Q1=q1*(22/7)/180;
Q2=q2*(22/7)/180;
Q3=q3*(22/7)/180;
Q4=q4*(22/7)/180;
Q5=q5*(22/7)/180;

nx=cos(Q1)*sin(Q2+Q3+Q4)*cos(Q5)-sin(Q1)*sin(Q5);
y=cos(Q1)*sin(Q2+Q3+Q4)*cos(Q5)+cos(Q1)*sin(Q5);
z=cos(Q2+Q3+Q4)*sin(Q5);
```

Figure 8.4 Case Structure to Display the Optimised Output

```matlab
px=cos(Q2)*cos(Q2+Q3+Q4)*cos(Q5)+A3*cos(Q2+Q3)+A2*cos(Q2);
py=cos(Q1)*cos(Q2+Q3+Q4)*cos(Q5)+A3*cos(Q2+Q3)+A2*cos(Q2);
```

```matlab
F=(nx-ny)*t+(ny-nx)*t+(nx-nz)*t;
```

```matlab
F0=(nx-ny)*t+(ny-nx)*t+(nx-nz)*t;
```

```matlab
FA=(nx-ny)*t+(ny-nx)*t+(nx-nz)*t;
```

```matlab
FP=(nx-ny)*t+(ny-nx)*t+(nx-nz)*t;
```
Figure 8.5 Front Panel of CIIKM (Before Execution)

All the components are substituted in the objective function. Now it will be ready for inverse kinematics solution (Refer Figure 8.5).

When the programme is executed, the combined objective function becomes zero at particular set of values, and the values will be indicated in the front panel as shown in the Figure 8.6.
8.6 CIIKM Front Panel (After Execution)

8.2.2 Partial Iteration IK Method (PIIKM)

The ranges of joints listed in Table 4.2 are as follows. T1: Base (±155), T2: Shoulder (-35 to +130), T3: Elbow (±130), T4: Wrist Pitch (±130), T5: Wrist Roll (±570). (θ₁, θ₂, θ₃, θ₄, and θ₅ are indicated as T1, T2, T3, T4, and T5 respectively in the LabVIEW model). In this method, the goal position is set first before the programme is run as shown in Figure 8.7.
The three formulae used to find the joint parameters are as follows: T1, T5, T234 (θ_1, θ_5 and θ_{234} are indicated as T1, T5 and T234 respectively in the LabVIEW model). It is shown in Figure 8.8.
The other two parameters \( T_3 (\theta_3) \) and \( T_4 (\theta_4) \) are started from minimum range -130 up to maximum range +130 as iterative trial. The values are given as input for the next formula node shown in Figure 8.9.

![Formula node for Iterative Joint Parameters](image)

**Figure 8.9 PIIKM Formula node for Iterative Joint Parameters**

The objective function is shown in the next node as shown in Figure 8.10. Totally four formula node are generated.

![Formula node for Objective Functions](image)

**Figure 8.10 Formula Node for Objective Functions**
In this method, the goal position is set first before the programme is run as shown in Figure 8.11. When the programme is executed, the combined objective function becomes zero at particular set of values, and the values will be indicated in the front panel as shown in the Figure 8.12.

Figure 8.11 Front Panel PIKKM (Before Execution)
Figure 8.12 Front Panel PIiKM (After Execution)

Both CIIKM and PIiKM iterations have been successfully developed using LabVIEW.