Inverse Kinematics

Forward Kinematics solves the problem "if I know the link transformation parameters, where are the links?".

Inverse Kinematics (IK) solves the problem "If I know where I want the links to be \((X^*, Y^*)\), what link transformation parameters will put them there?"
Inverse Kinematics (IK):
Things Need to Move – What Parameters Will Make Them Do That?

Cyclic Coordinate Descent (CCD) Method

The idea is to change $\Theta_1$ so that $(X,Y)$ are as close to $(X^*,Y^*)$ as possible.
Then change $\Theta_2$.
Then change $\Theta_3$.
Then change $\Theta_1$.
Then change $\Theta_2$.
Then change $\Theta_3$.
Then change $\Theta_1$.

\[ (X^*,Y^*) \]

\[ (X,Y) \]

Ground
Changing $\Theta_1$

Holding $\Theta_2$ and $\Theta_3$ constant, rotate $\Theta_1$ towards $(X^*, Y^*)$ so that the dashed purple lines line up.
Holding $\theta_1$ and $\theta_3$ constant, rotate $\theta_2$ towards $(X^*, Y^*)$ so that the dashed purple lines line up.
Holding $\theta_1$ and $\theta_2$ constant, rotate $\theta_3$ towards $(X^*, Y^*)$ so that the dashed purple lines line up.
Now, do it again -- Changing $\theta_1$
Now, do it again -- Changing $\Theta_2$
Now, do it again -- Changing Θ3

(X*,Y*)

(X,Y)

Now, do it again -- Changing Θ3

(X*,Y*)

(X,Y)
Computing how much to change a rotation by (in this example, we are changing $\theta_2$)

Where we are now: $(X_2, Y_2)$

Where we want to be: $(X_3, Y_3)$

Use the C/C++ $\text{atan2}()$ function:  

\[
\theta' = \text{atan2}( Y' - Y_2, X' - X_2 ); \\
\theta = \text{atan2}( Y_3 - Y_2, X_3 - X_2 ); \\
\Delta \theta_2 = \theta' - \theta
\]

Do not use the C/C++ $\text{atan}()$ function:

\[
\theta' = \text{atan}( (Y' - Y_2) / (X' - X_2) ); \\
\theta = \text{atan}( (Y_3 - Y_2) / (X_3 - X_2) ); \\
\Delta \theta_2 = \theta' - \theta
\]