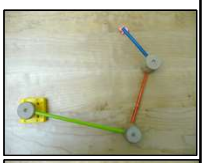
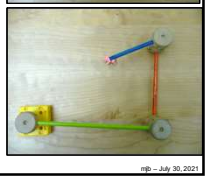
1

# Inverse Kinematics





Mike Bailey
mjb@cs.oregonstate.edu



Oregon State University
Computer Graphics

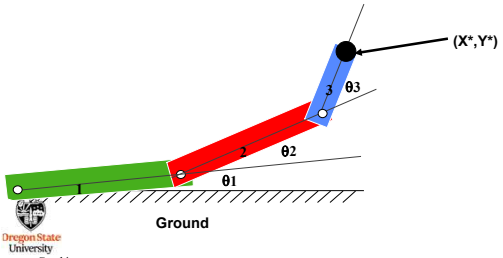


1

## 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?"

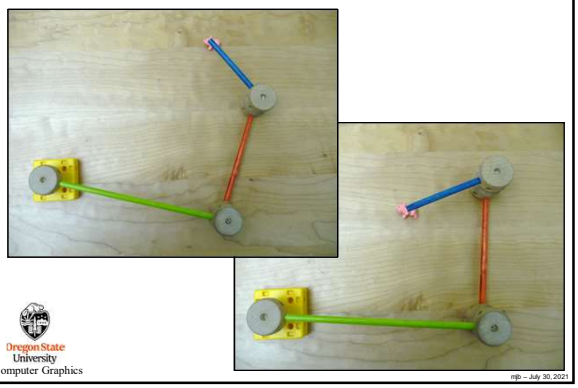



Oregon State University
Computer Graphics



2

## Inverse Kinematics (IK): Things Need to Move – What Parameters Will Make Them Do That?



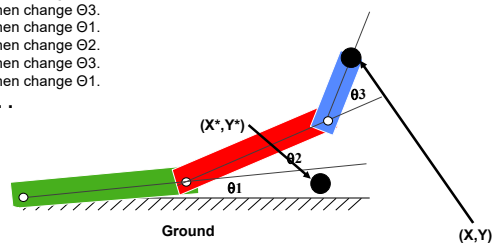
Oregon State University
Computer Graphics



3

## Cyclic Coordinate Descent (CCD) Method

The idea is to change Θ1 so that (X,Y) are as close to (X*,Y*) as possible.
Then change Θ2.
Then change Θ3.
Then change Θ1.
Then change Θ2.
Then change Θ3.
Then change Θ1.
. . .



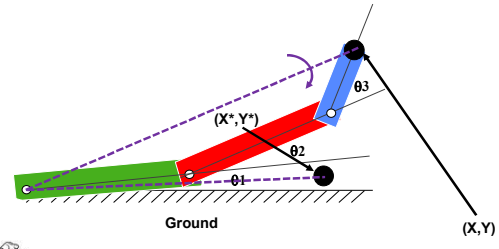

Oregon State University
Computer Graphics



4

## Changing Θ1

Holding Θ2 and Θ3 constant, rotate Θ1 towards (X*,Y*) so that the dashed purple lines line up.

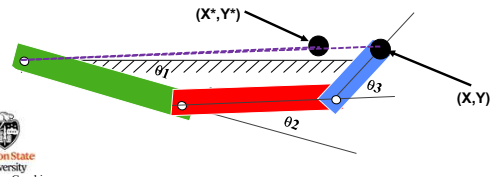



Oregon State University
Computer Graphics



5

## Changing Θ1




Oregon State University
Computer Graphics

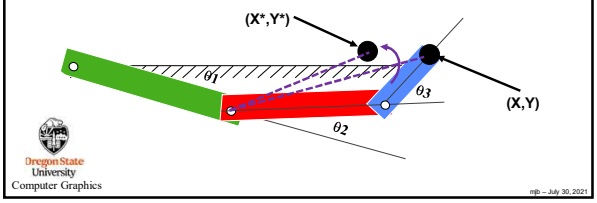


6

7

## Changing Θ2

Holding Θ1 and Θ3 constant, rotate Θ2 towards (X*,Y*) so that the dashed purple lines line up.



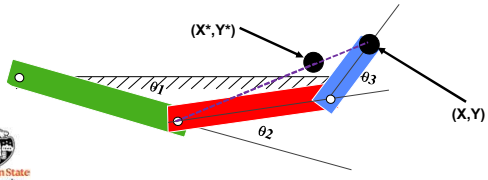

Oregon State University Computer Graphics

mjb – July 30, 2021

7

8

## Changing Θ2




Oregon State University Computer Graphics

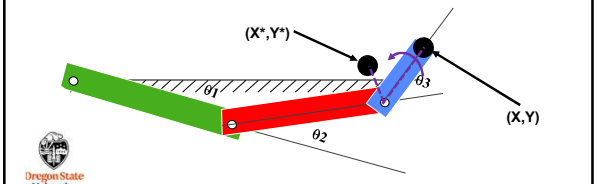
mjb – July 30, 2021

8

9

## Changing Θ3

Holding Θ1 and Θ2 constant, rotate Θ3 towards (X*,Y*) so that the dashed purple lines line up.



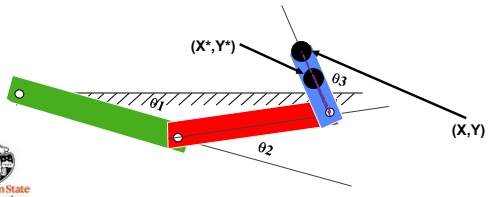

Oregon State University Computer Graphics

mjb – July 30, 2021

9

10

## Changing Θ3



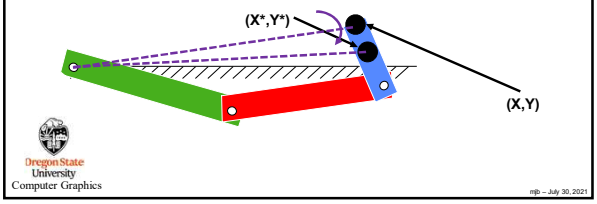

Oregon State University Computer Graphics

mjb – July 30, 2021

10

11

## Now, do it again -- Changing Θ1



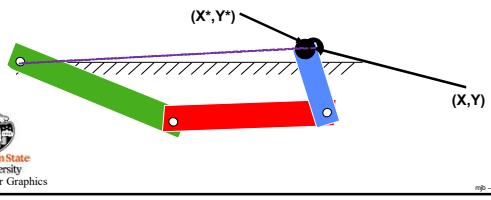

Oregon State University Computer Graphics

mjb – July 30, 2021

11

12

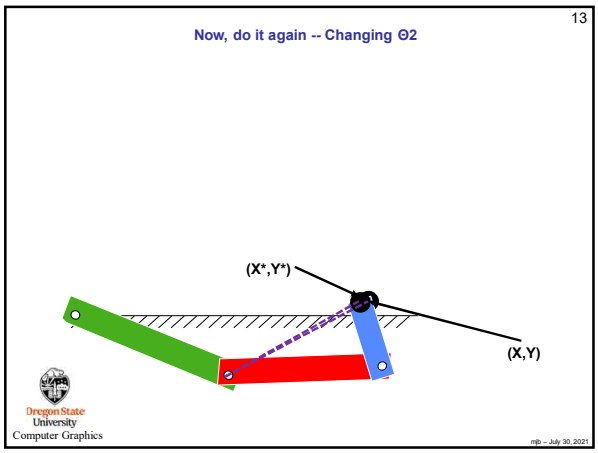
## Now, do it again -- Changing Θ1



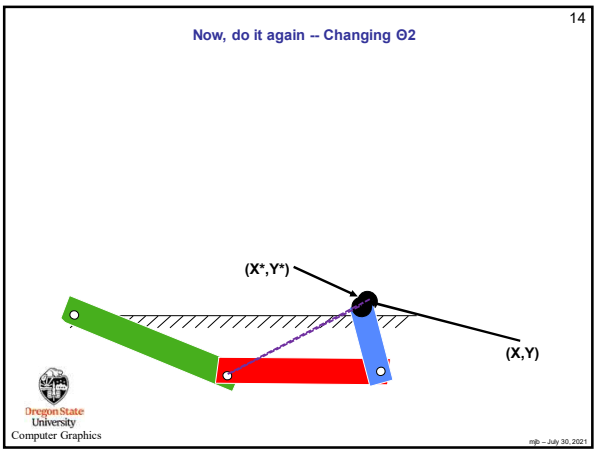

Oregon State University Computer Graphics

mjb – July 30, 2021

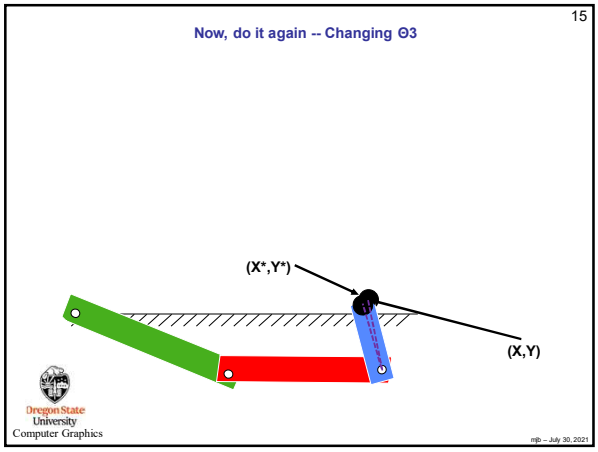
12


13
Now, do it again -- Changing Θ2
(X*,Y*)
(X,Y)
Oregon State
University
Computer Graphics
mjb – July 30, 2021

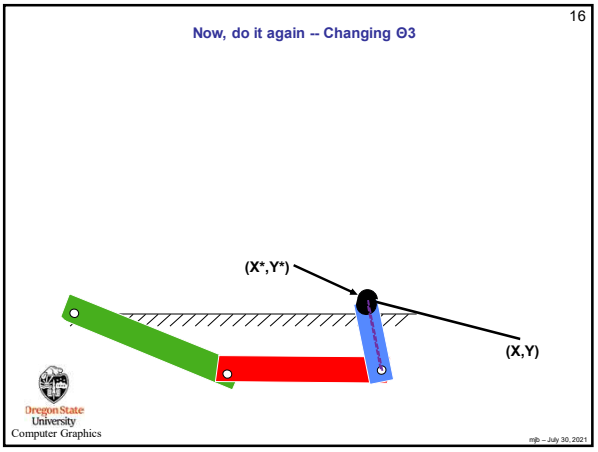
13


14
Now, do it again -- Changing Θ2
(X*,Y*)
(X,Y)
Oregon State
University
Computer Graphics
mjb – July 30, 2021

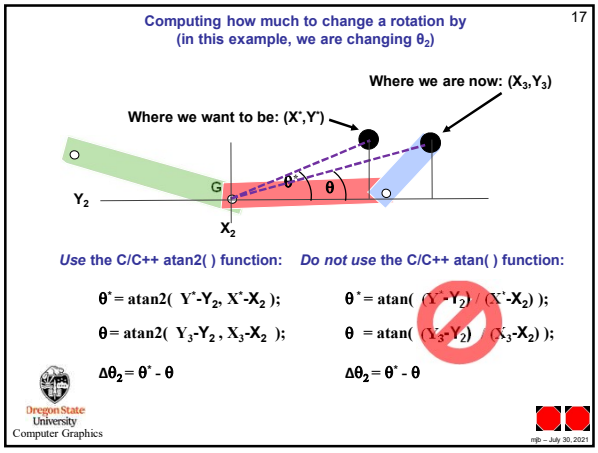
14


15
Now, do it again -- Changing Θ3
(X*,Y*)
(X,Y)
Oregon State
University
Computer Graphics
mjb – July 30, 2021

15


16
Now, do it again -- Changing Θ3
(X*,Y*)
(X,Y)
Oregon State
University
Computer Graphics
mjb – July 30, 2021

16


17
Computing how much to change a rotation by
(in this example, we are changing θ2)
Where we are now: (X3,Y3)
Where we want to be: (X*,Y*)
G
θ*
θ
Y2
X2
Use the C/C++ atan2( ) function:
Do not use the C/C++ atan( ) function:
θ* = atan2( Y*-Y2, X*-X2 );
θ = atan2( Y3-Y2 , X3-X2 );
Δθ2 = θ* - θ
θ* = atan( (Y*-Y2) / (X*-X2) );
θ = atan( (Y3-Y2) / (X3-X2) );
Δθ2 = θ* - θ
Oregon State
University
Computer Graphics
mjb – July 30, 2021

17