Development of Complex Curricula for Molecular Bionics and Infobionics Programs within a consortial* framework**

Consortium leader

PETER PAZMANY CATHOLIC UNIVERSITY

Consortium members

SEMMELWEIS UNIVERSITY, DIALOG CAMPUS PUBLISHER

The Project has been realised with the support of the European Union and has been co-financed by the European Social Fund ***

**Molekuláris bionika és Infobionika Szakok tananyagának komplex fejlesztése konzorciumi keretben

***A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

Neuromorph Movement Control

(Neuromorf mozgás vezérlés)

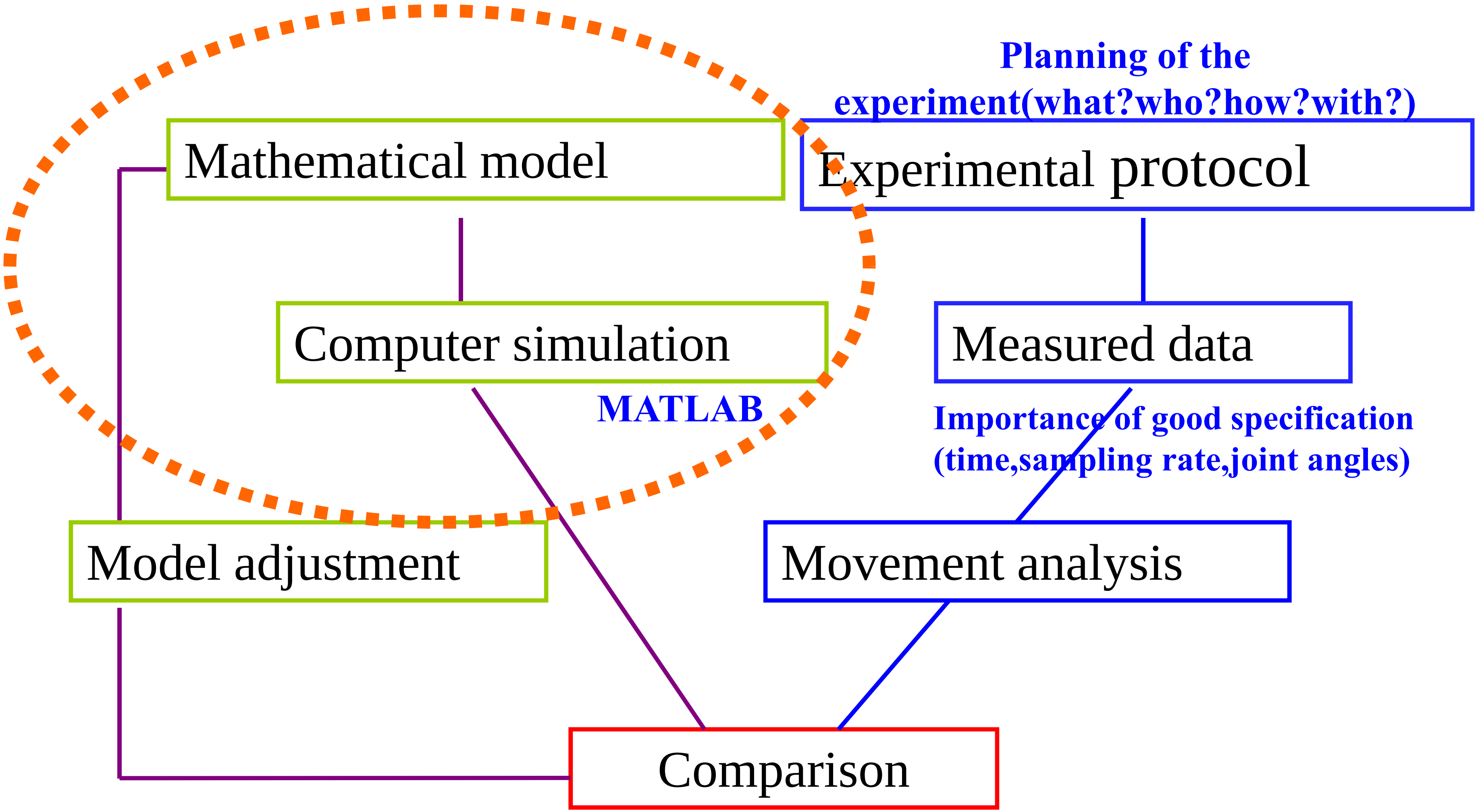
Solving the issue of Inverse Kinematics

(Inverz kinematikai probléma megoldása)

József LACZKÓ PhD; Róbert TIBOLD

Main points of the lecture

- The relation between mathematical modeling and experimental protocols
- Definition of Inverse kinemaic problem
    - The Jacobian matrix
    - An iterative solution of the invers kinematic problem
- Generating the inverse of the Jacobian (J) matrix
    - Is it possible to invert J?
    - Pseudo inverse methods(Moore-Penroose,Truncated pseudo inverse, Damped Least Square)
- Problems with the inverse kinematic problem


Planning of the experiment(what?who?how?with?)
Mathematical model
Experimental protocol
Computer simulation
Measured data
MATLAB
Importance of good specification (time,sampling rate,joint angles)
Model adjustment
Movement analysis
Comparison

2 main direction of implementation

Bottom-Up(BU) = Solving the Direct problem

Top-Down(TD) = Solving the Inverse problem

Given: Measured/computed angular changes

Parameters: inertial properties,gravitational torque

Biomechanical parameters: force-length,force-velocity,frequency-force relations

Compute: Activity of motoneurons

Relation between the complexity of the two problems: BU < TD

- Human limb movements are controlled by motor commands originated from the central nervous system (CNS). These commands descend to spinal motoneurons and stimulate muscles to invoke muscle forces and generate torques in the joints.
- Human limb movements are characterized by kinematic redundancy: the number of independent axes of joint rotations and the number of available muscles exceeds the number of parameters describing a given motor task.
- A general question is which parameters are controlled by the CNS for a successful execution of a given motor task.

Human motor redundancy and inverse problems

The term "synergy" is used in the motor control literature in relation to the problem of motor redundancy.

How does the central nervous system select particular solutions for motor tasks, which typically have an infinite number of solutions?

Which solutions might be chosen by human neural motor control?

Solutions of the inverse problems in healthy cases and in patients with neurologically based movement disorders might be significantly different

Inverse Kinematic problem

a)

- **Given:** position of the endpoint of a kinematic chain (limb) and the lengths of the chain's segments.
- **Question:** intersegmental angles in the joints

b)

- **Given:** position and the velocity of the endpoint of a kinematic chain and the lengths of the chain's segments.
- **Question:** the angular velocities in the joints.

In other words:

Given: a limb and a desired trajectory or target position of its endpoint.
Compute: the time course of joint angles that would result the given endpoint trajectory.

Problems with the IK solution

Redundancy problem

- At least 6 degrees of freedom (DOF) required a chain to reach a target in 3D space
- A chain with many links each having 6 DOF has more DOF needed to reach the goal
- Infinite number of solutions

Targets out of workspace (unreachable)

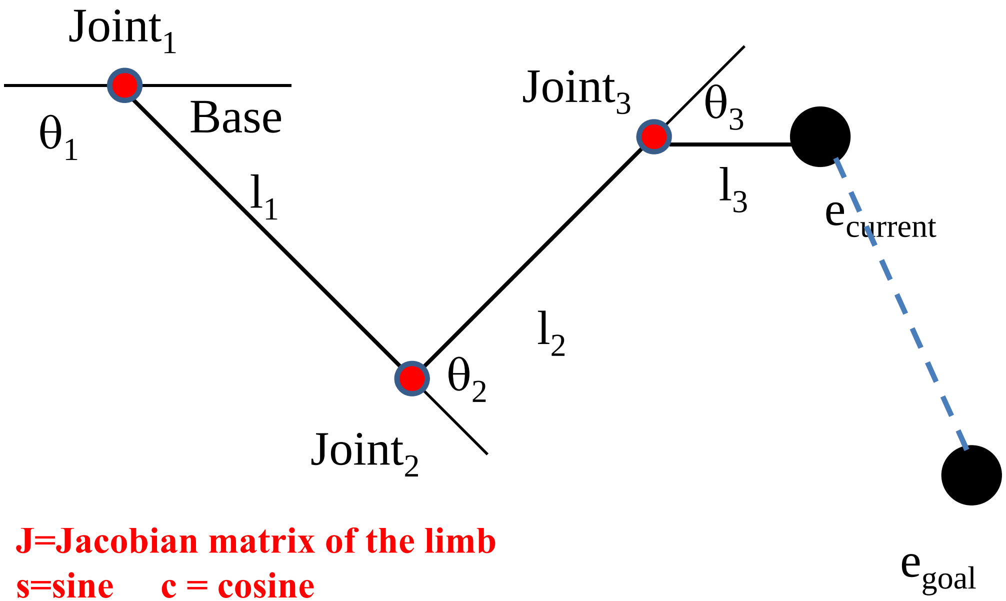
- Targets could be farther the chain reaches

Singularities

- Occurs when no angular changes can achieve a desired change in endposition

Limb: an object containing

- different **segments (links)** linked to each other via **joints**
- **Segments** are capable of **rotating** around **joints**



Θ_i = joint angles

$Link_i$ = segments

$Joint_i$ = joints

$e_{current}$ = current position of endpoint

e_{goal} = goal end point to be reached

- - - = trajectory to be followed

J=Jacobian matrix of the limb
s=sine c = cosine

$$J = \begin{bmatrix} -l_1 s(\Theta_1) - l_2 s(\Theta_1+\Theta_2) - l_3 s(\Theta_1+\Theta_2+\Theta_3) & -l_2 s(\Theta_1+\Theta_2) - l_3 s(\Theta_1+\Theta_2+\Theta_3) & -l_3 s(\Theta_1+\Theta_2+\Theta_3) \\ l_1 c(\Theta_1) + l_2 c(\Theta_1+\Theta_2) + l_3 c(\Theta_1+\Theta_2+\Theta_3) & l_2 c(\Theta_1+\Theta_2) + l_3 c(\Theta_1+\Theta_2+\Theta_3) & l_3 c(\Theta_1+\Theta_2+\Theta_3) \end{bmatrix}$$

$$\Theta = \begin{bmatrix} \Theta_1 \\ \Theta_1 \\ .... \\ \Theta_n \end{bmatrix}$$ Set of rotation angles of the entire chain

$$e = \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix}$$ The position of the endpoint of the chain

IK can be calculated iteratively

$\Theta = f^{-1}(e)$ The IK solution $\longrightarrow$ $\dot{\Theta} = J^{-1}\dot{e}$

Where J is the Jacobian matrix: generated by the partial derivatives of the endpoint of the kinematic chain.

$$J \equiv \frac{\partial e}{\partial \Theta} = \begin{bmatrix} \frac{\partial e_x}{\partial \Theta_1} & \frac{\partial e_x}{\partial \Theta_2} & \cdots & \frac{\partial e_x}{\partial \Theta_n} \\ \frac{\partial e_y}{\partial \Theta_1} & \frac{\partial e_y}{\partial \Theta_2} & \cdots & \frac{\partial e_y}{\partial \Theta_n} \\ \frac{\partial e_z}{\partial \Theta_1} & \frac{\partial e_z}{\partial \Theta_2} & \cdots & \frac{\partial e_z}{\partial \Theta_n} \end{bmatrix}$$

Number of rows = number of dimension
Number of columns=number of joints

An alternate way to determine:

$$J_i = z_i \times n_i$$

$$n_i = e - p_i$$

- J_i the *i*.th column of the Jacobian
- z_i unit vector parallel to the axis of rotation in joint *i*
- p_i the world position of joint *i*
- n_i the vector pointing from joint i to the endpoint of the chain

General algorithm to solve inverse kinematics (IK) problem

- 1. Compute J *(jacobian matrix)*
- 2. Compute $\Delta\Theta$ *(θ is a vector of joint angles)*
- 3. $\Theta_{t+1} = \Theta_t + \Delta\Theta$
- 4. Update the joint angles and endpositions using 3.
    - Solved by Direct Kinematics
- 5. Repeat until $\mathbf{e}_{\text{current}}$ is within tolerance of $\mathbf{e}_{\text{goal}}$ or iteration count exhausted.

An iterative approach to get the optimum solution of kinematic problem

- Δe is to be small enough to get $e_{current}$ as close to e_{goal} as it is possible

$$\Delta e = e_{goal} - e_{current}$$

$$\Delta\Theta = J^{-1}\Delta e$$

$$\Theta_{t+1} = \Theta_t + \Delta\Theta$$

- **If Δe is not small enough:**
  - Iteration steps of the IK solution algorithm are increasing exponentially as a function of Δe
  - e_{goal} cannot be reached within small tolerance of distance error
- **The most optimal solution would be an:**
  - Always invertible Jacobian matrix

Is it always possible to invert *J* ?

– Generally speaking: NO

– Redundancy in the system means:
- *J* is an *n x m* matrix. If **n≠m** than it can’t be **inverted**
- If **n < m** (if the dimension of the workspace is smaller than the number joint) there is an infinite soluiton
- If **n > m** (if the ….) there may not be exact solution
- Even if **n=m** det(J) may vanish (det(J)=0)

A simple example for J if n=m (dimensons=joints)

- **Consider:** twe movement of a 2-joint system in which the lengths of two adjacent segments are equal ($l_1=l_2$)

- In this case

$$J=\begin{bmatrix} -ls(\alpha_1)-ls(\alpha_1+\alpha_2) & -ls(\alpha_1+\alpha_2) \\ lc(\alpha_1)+lc(\alpha_1+\alpha_2) & lc(\alpha_1+\alpha_2) \end{bmatrix}$$

- **Compute the determinant of J:**

$$\det(J)=l((-\sin(\alpha_1)-\sin(\alpha_1+\alpha_2))(\cos(\alpha_1+\alpha_2))-(-\sin(\alpha_1+\alpha_2)(\cos(\alpha_1)+\cos(\alpha_1+\alpha_2)))=$$
$$=\sin(\alpha_2)$$

- Singular configuration of this kinematic chain:
  if **sin(α2)=0** than det=(0)

Practically, singular configuration of this two segment limb means that $\alpha 2 = \pi$, thus the limb is extended. In the case of the 2-joint system of the upper and lower arm singularity: the elbow is fully streched.

- **Because fo multi-joint systems, J in generally is not invertible**

another way must be found to get the quasi inverse of J

- **The question is: how to substitute the inverse of *J* ?**
- **Here are some approaches to do this:**
  - Moore-Penrose pseudo inverse **(J+)**
  - SVD (Singular Value Decomposition)
  - Truncated pseudo inverse
  - Damped Least Squares (DLS)

Moore-Penrose (MP) pseudo inverse

•Replace J^{-1} with J^{+} computed as follows: $J^{-1} \cong J^{+} = \left(J^{T} J\right)^{-1} J^{T}$

•To compute the changes in the set of joint angles $\Delta\Theta = J^{+}\Delta e$

•This method tends to converge on the solution and error can be reduced by having small steps per iteration.

•**Iteration method to compute Δθ by minimizing Δe:**

1. Compute J+
2. Compute $\Delta e = e_{goal} - e_{current}$
3. Compute $error = \left\| (I - J^{+} J)\Delta e \right\|$
4. If error>tolerance then Δe:=Δe/2 and repeat from 3.
5. $\Delta\Theta = J^{+}\Delta e$

Pseudo Inverse based on Singular Value Decomposition (SVD)

- Pseudo inverse of the Jacobian can be calculated using reduced SVD

$$J = USV^T$$

$$J^+ = VS^+U^T$$

- **The columns of *V*** form a set of orthonormal "input“ basis vector directions for *J*. (eigenvectors of J^TJ)
- **The columns of *U*** form a set of orthonormal "output" basis vector directions for *J*. (eigenvectors of JJ^T)
- **The diagonal values in matrix S** are the singular values,
  (**S^+ is pseudo inverse**)

SVD is regarded as scalar "gain controls" by which each corresponding input is multiplied to give the output: square roots of the eigenvalues of JJ^T and J^TJ corresponding with the same columns in U and V.
(if each of the joints has only one DOF than U and V has as many columns as many joints are in the chain))

Truncated pseudo inverse

- **Singularities** (See Problems with the IK solution) in the system are manifested as singular values **equal to or close to zero**.
  - Omit these components of the solution (They cause the excessive joint angle swings that inhibit convergence)

- **The components corresponding to the larger singular values**, contribute the most toward smooth predictable convergence.

Compute J+:

$$J^{+} = \sum_{i=1}^{k} \frac{1}{\sigma_i} v_i u_i^T$$

where **k ≤ r** where **r** is the rank of **J** and σ_i is the smallest singular value, determined by a threshold. (**$u_i v_i$ are column vectors from SVD**)

Multilink chains are redundat thus the rank of J is smaller than the number of joints

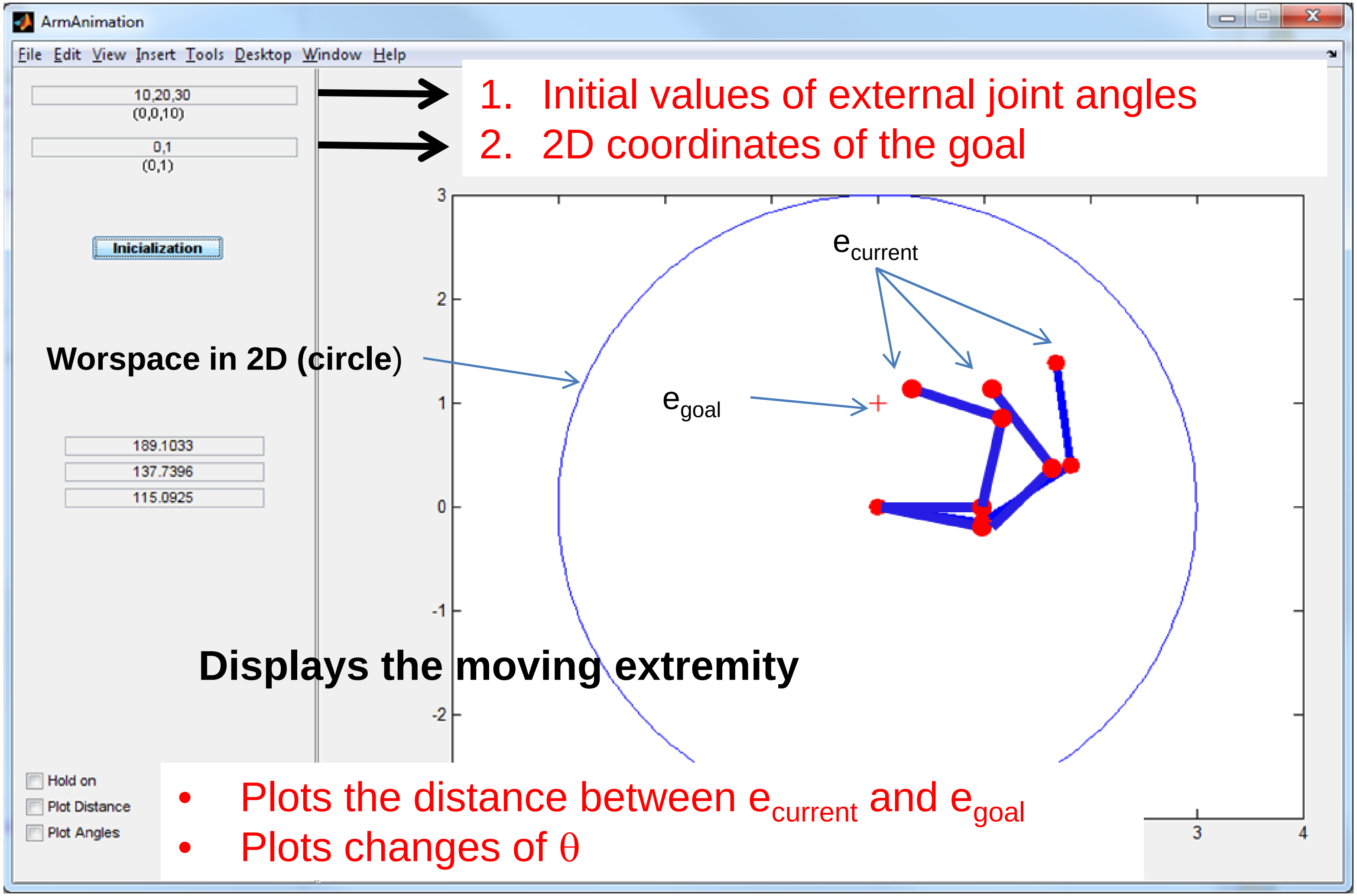
Damped least squares (DLS)

- **Pseudo inverse methods are vulnarable to singularities.**
  1. these tend to oscillate with high amplitude
  2. DLS solves this problem by introducing a damping constant (λ)

$$J^{+} = \sum_{i=1}^{r} \frac{\sigma_i}{\sigma_i + \lambda^2} v_i u_i^T$$

- Opposingly to the truncated pseudo inverse in this case DLS works fine because when the system is near to a singularity and σ_i tend to converge to zero λ^2 will „dominate" the sum preventing from growing excessively.
  - λ must be large enough to restrain angular velocity near singularities but small enough to allow rapid convergence.

- **A MATLAB based program was developed to**
  - Solve the IK
  - Visualise the solution (**moving limb, trajectory**)
    - Self-developed GUI(**graphical user interface**)
  - Present:
    1. Angular changes
    2. Distance between endpoint($\mathbf{e}_{\mathbf{current}}$) and the target($\mathbf{e}_{\mathbf{goal}}$)
- **In the next slides:**
  - Upper limb (reaching e_{goal} from arbitrary location within reachable distance)
  - In the lecture we present realtime animation

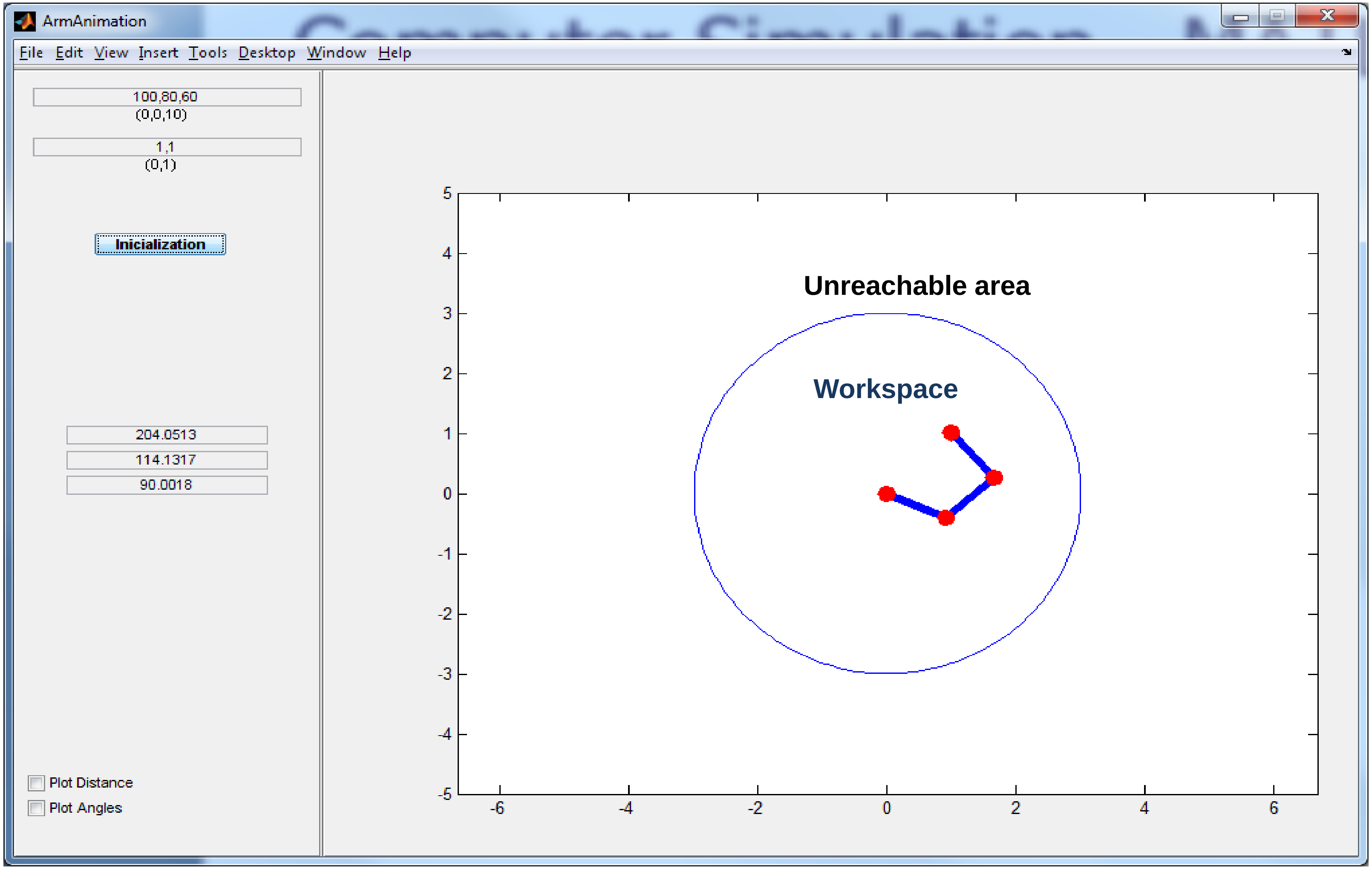


1. Initial values of external joint angles
2. 2D coordinates of the goal

Worspace in 2D (circle)

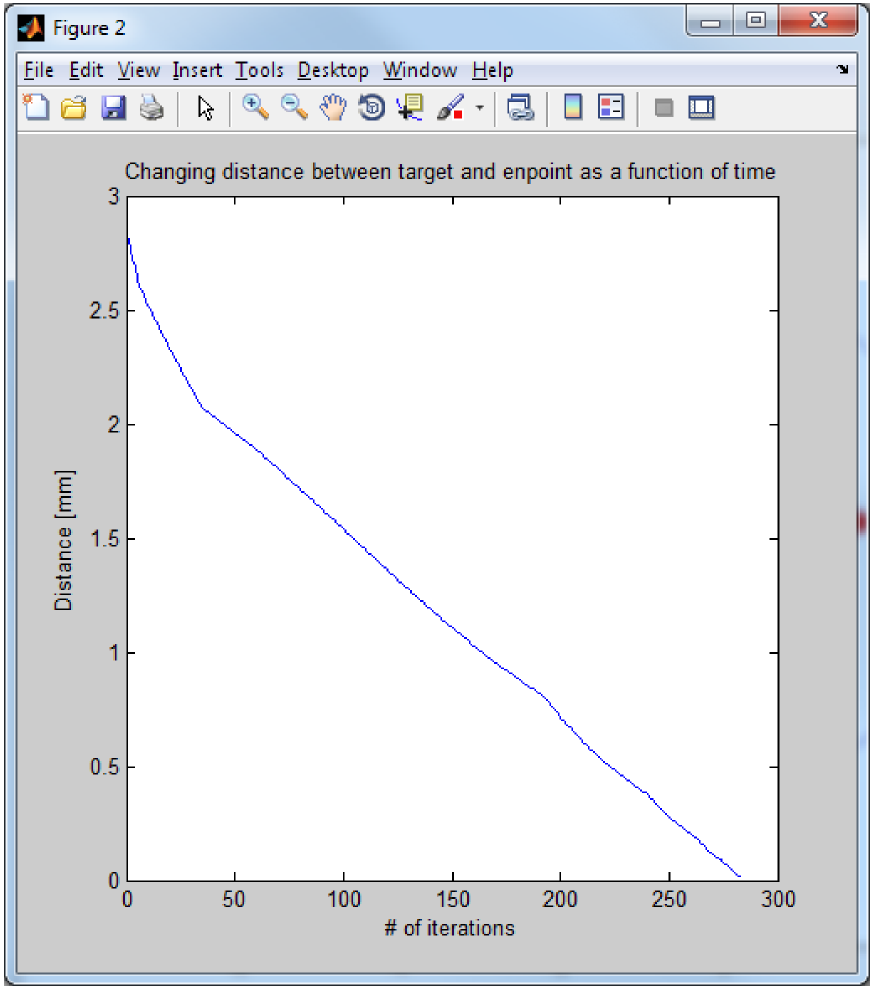
Displays the moving extremity

- Plots the distance between $e_{current}$ and e_{goal}
- Plots changes of θ

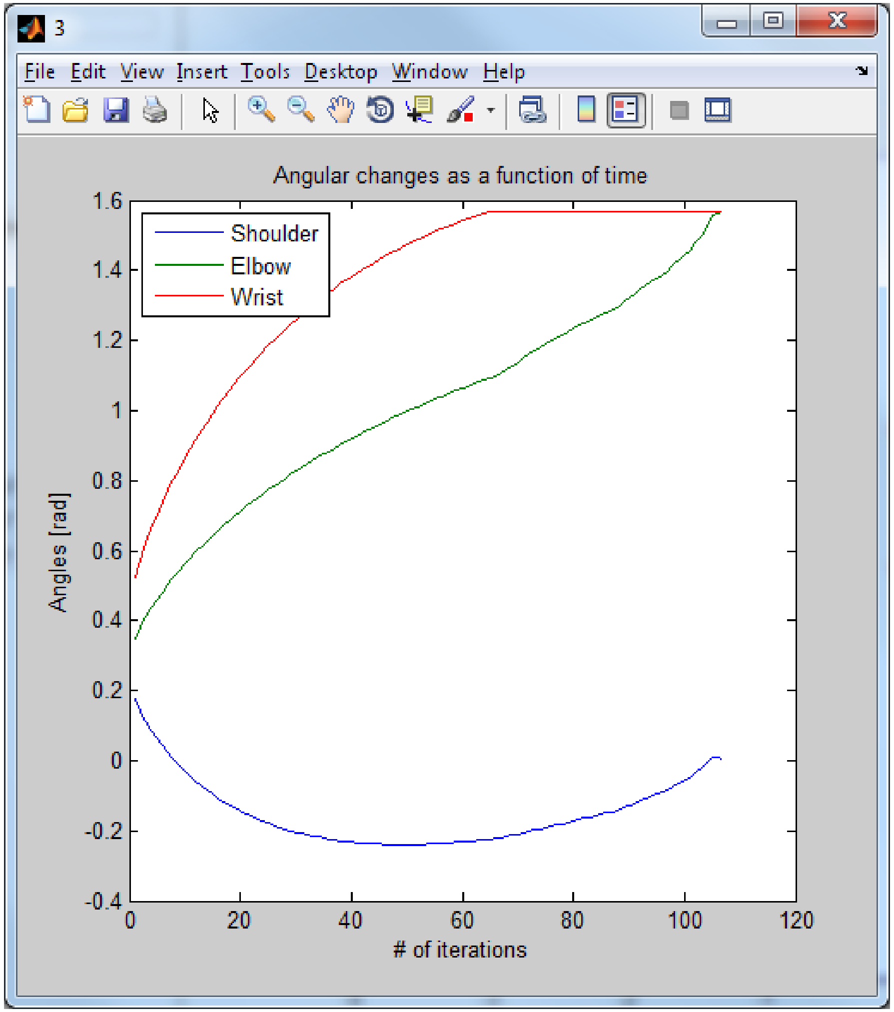


ArmAnimation
File Edit View Insert Tools Desktop Window Help
100,80,60
(0,0,10)
1,1
(0,1)
Inicialization
204.0513
114.1317
90.0018
Plot Distance
Plot Angles
Unreachable area
Workspace
5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6

Neuromorph Movement Control:
Solving the issue of Inverse Kinematics



- **Decreasing distance between $e_{current}$ and e_{goal}**
  - E.g.:reaching and object
- **Nearly linear trajectory**
- **An error can be minimized by choosing smaller steps in iterations**
  - Important to choose small enough tolerance value



Angular changes as a function of movement time (number of iterations)

- **In the next slides:**
  - Upper limb (reaching e_{goal} from arbitrary location within reachable distance)
    - 2 examples
      1. For a succesfull reaching
      2. For a sinularity
  - In the lecture we present realtime animation
- **Sign of singularity:**
  - **In realtime animations:** the amount of angular changes is decreasing continously and target is not found
  - Infinite running time

Reaching a given point in the 2D space

Introducing a 2D reaching task

Initial outer joint angles in degrees:

$a_{Shoulder}$: 10^o
a_{Elbow}: 20^o
a_{Wris}t: 30^o
$Target_{x,y}$: 0.5,0.5

Inter segmental joint angles in degrees (at a discrete time instant):

$a_{Shoulder}$: 227.74^o
a_{Elbow}: 100.21^o
a_{Wris}t: 98.89^o

Final inter segmental joint angles in degrees:

$a_{Shoulder}$: 245.02^o
a_{Elbow}: 73.36^o
a_{Wris}t: 90^o

Introducing the singularity

Reaching a given point in the 2D space

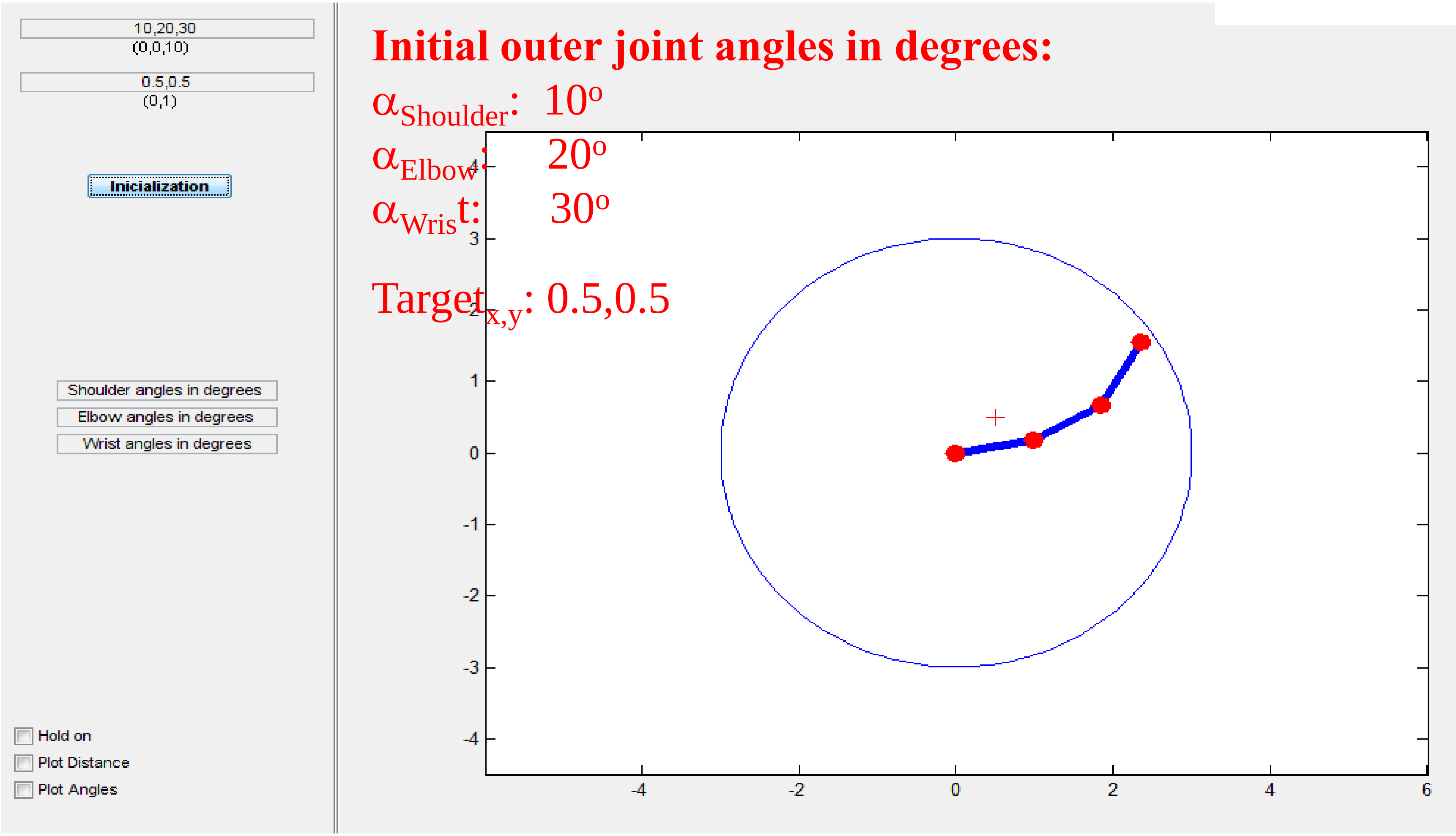
Initial outer joint angles in degrees:

$\alpha_{Shoulder}$: 10^o

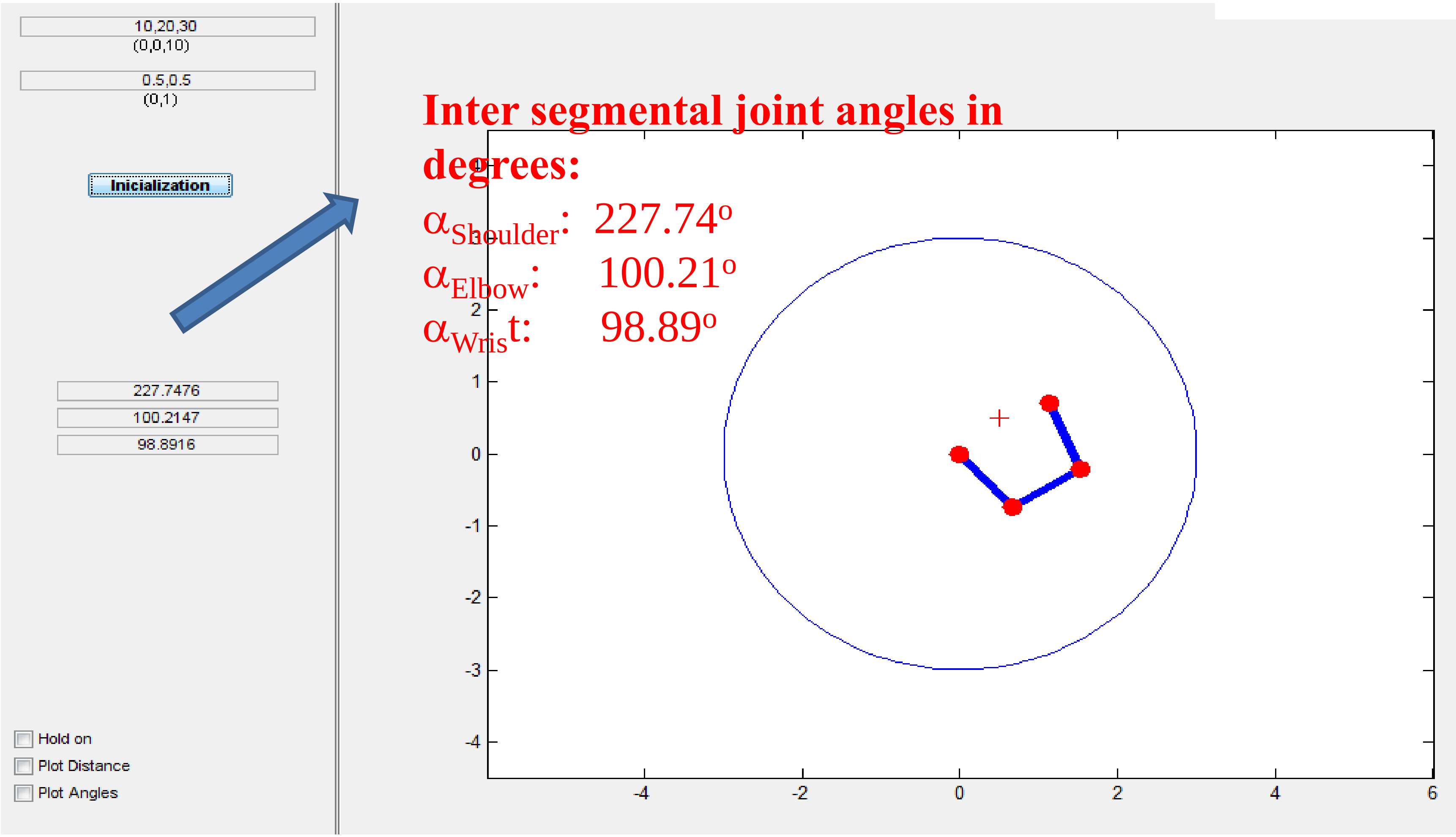
α_{Elbow}: 20^o

α_{Wrist}: 30^o

$Target_{x,y}$: 0.5,0.5



Reaching a given point in the 2D space



Inter segmental joint angles in degrees:

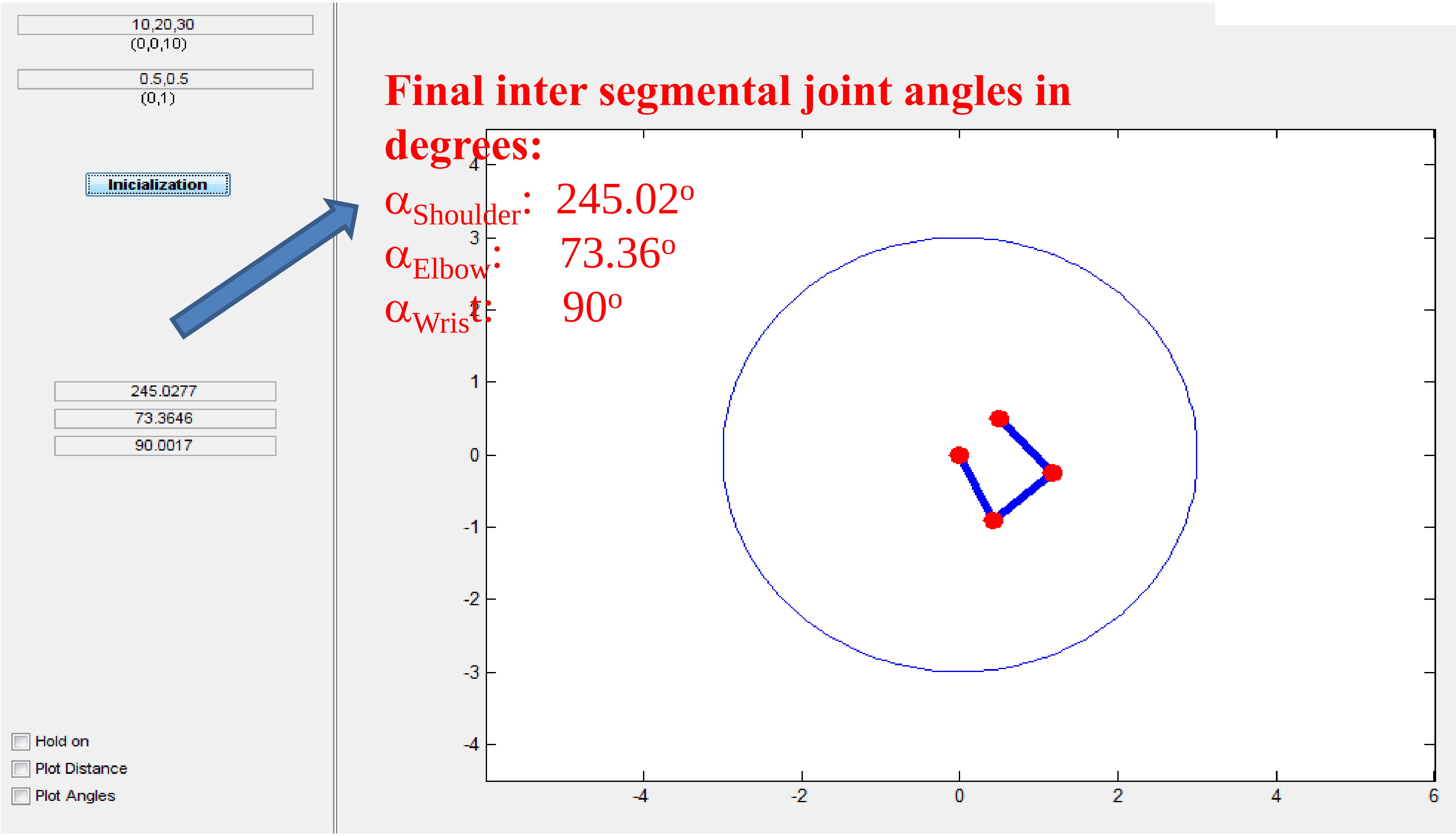
$\alpha_{Shoulder}$: 227.74°

α_{Elbow}: 100.21°

α_{Wrist}: 98.89°

Neuromorph Movement Control:
Solving the issue of Inverse Kinematics

Reaching a given point in the 2D space



Final inter segmental joint angles in degrees:

$\alpha_{Shoulder}$: 245.02°

α_{Elbow}: 73.36°

α_{Wrist}: 90°

Reaching a given point in the 2D space - Singularity

Initial outer joint angles in degrees are the same as in the previous example:

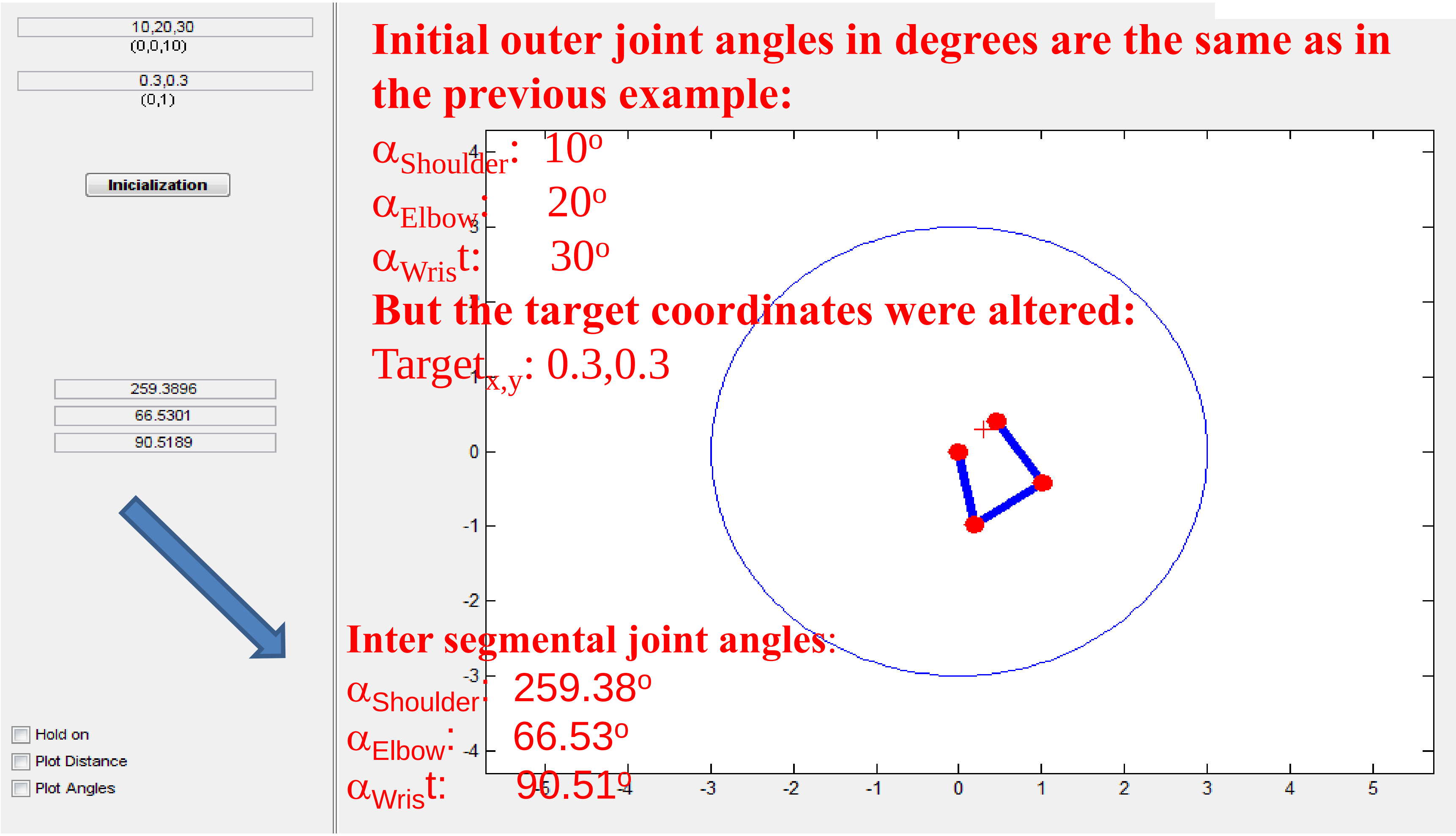
$\alpha_{Shoulder}$: 10°

α_{Elbow}: 20°

α_{Wrist}: 30°

But the target coordinates were altered:

$Target_{x,y}$: 0.3,0.3



Inter segmental joint angles:

$\alpha_{Shoulder}$: 259.38°

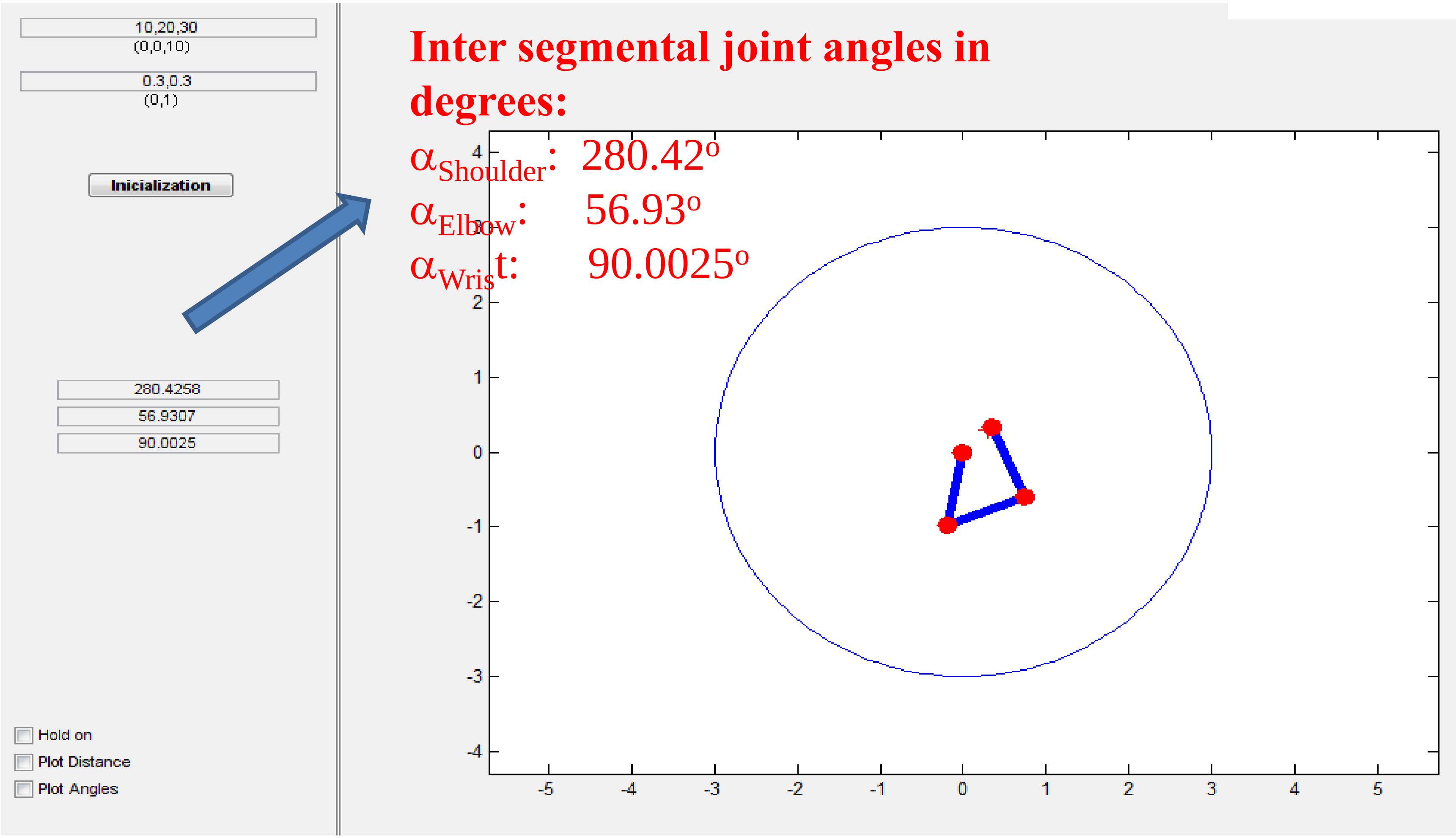
α_{Elbow}: 66.53°

α_{Wrist}: 90.51°

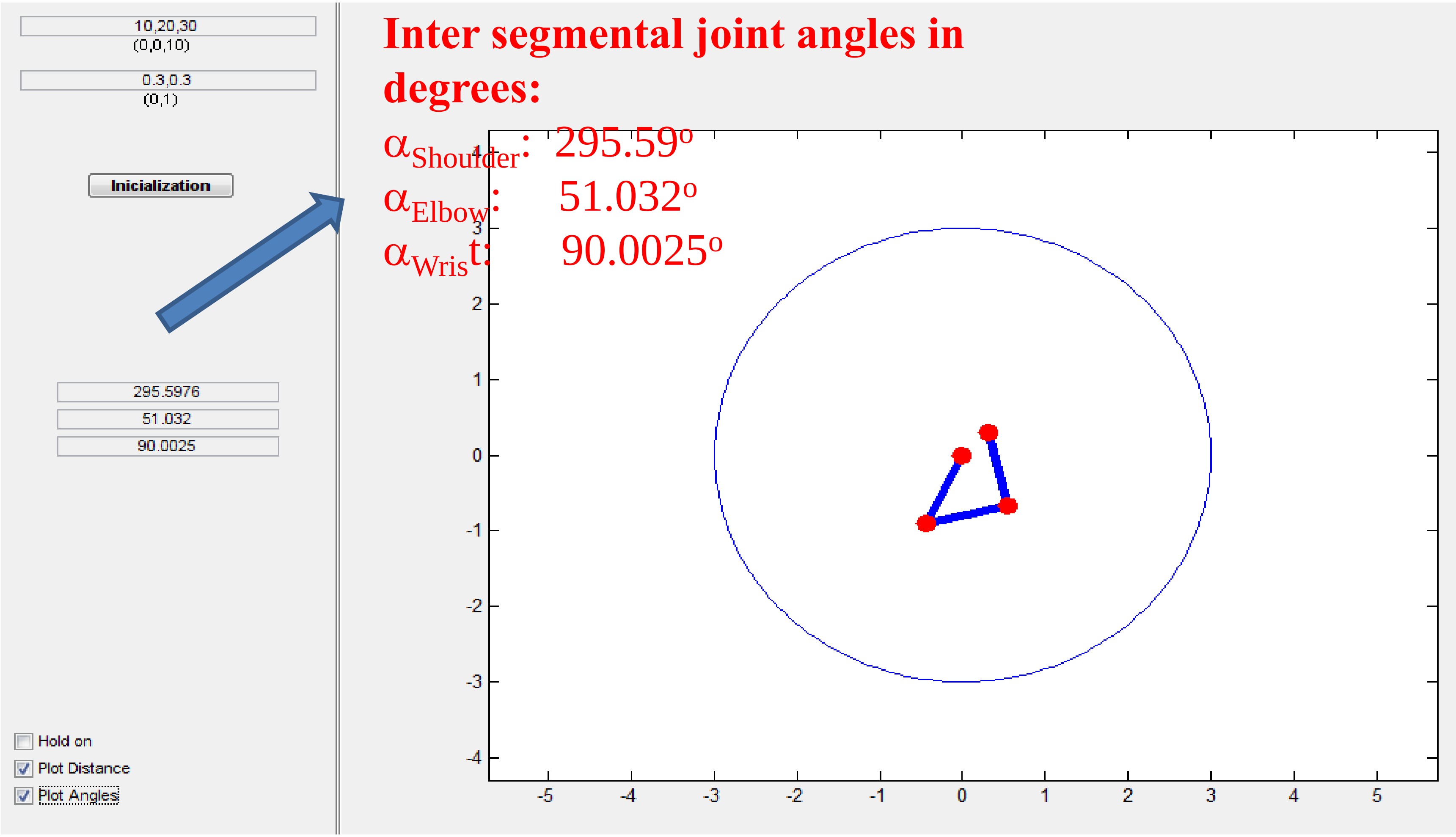
Reaching a given point in the 2D space - Singularity

Inter segmental joint angles in degrees:

$\alpha_{Shoulder}$: 280.42°
α_{Elbow}: 56.93°
α_{Wrist}: 90.0025°



Reaching a given point in the 2D space - Singularity



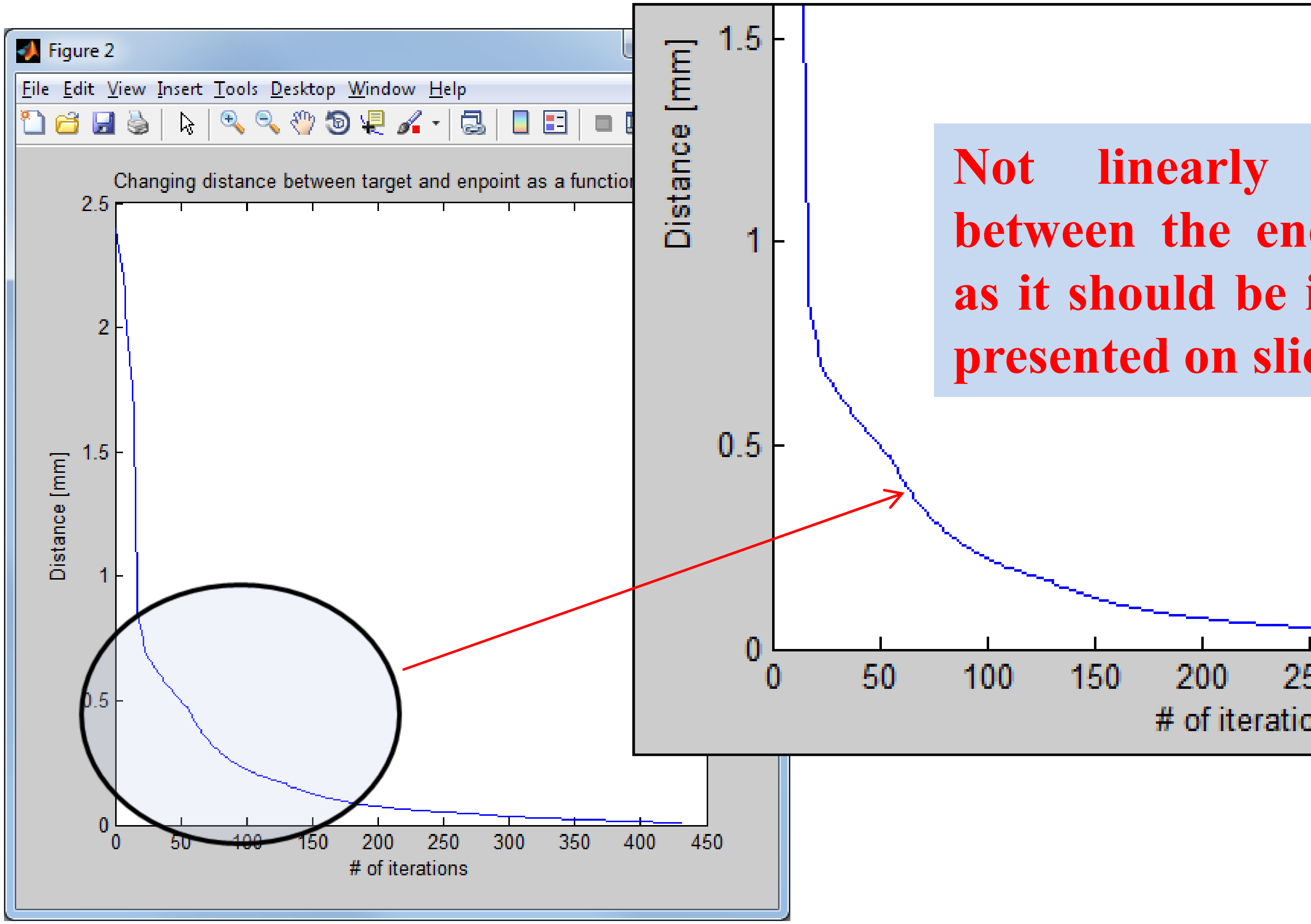
Inter segmental joint angles in degrees:

$\alpha_{Shoulder}$: 295.59°

α_{Elbow}: 51.032°

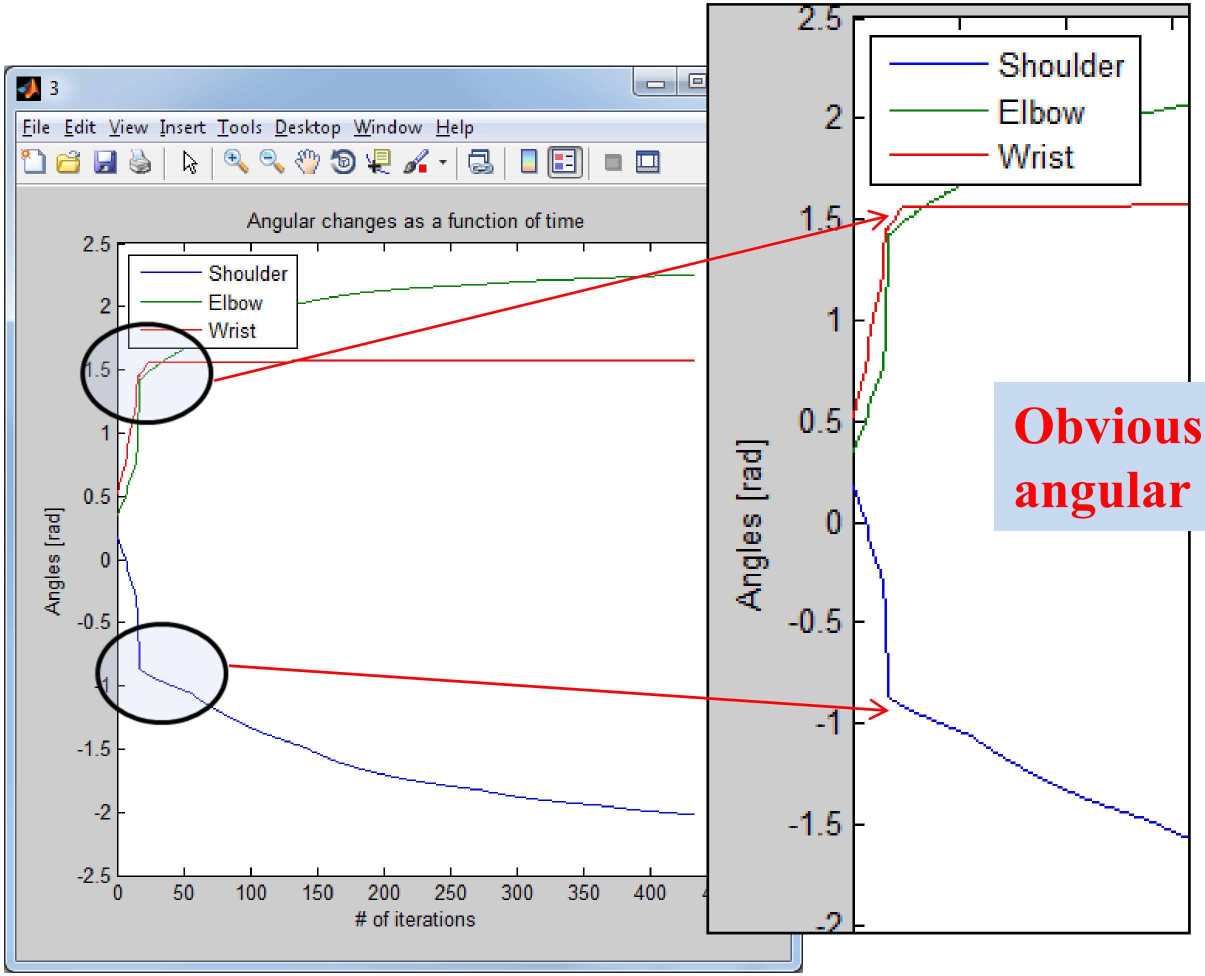
α_{Wrist}: 90.0025°

Reaching a given point in the 2D space – Singularity (Distance)



Not linearly decreasing distance between the endpoint and the target as it should be in succesful reacing as presented on slide #26

Reaching a given point in the 2D space – Singularity (Angular changes)



Obvious signs of singularity in angular changes

Summary

- **Inverse Kinematic problem: If the position of the endpoint of a given kinematic chain is given (with segment lengths) then compute the set of intersegmental joint angles.**
  - **The Jacobian matrix: to every endpoint position the Jacobian can be given**

- **In the IK problem solution one of the most important question is whether J is invertable or not.**
  - **Generally: J is not invertable**
  - **Thus: let's find the pseudo inverse of J which approximates a quasi inverse of J if the „error" is small enough.**

Suggested literature

- http://billbaxter.com/courses/290/html/index.htm
- http://freespace.virgin.net/hugo.elias/models/m_ik.htm
- http://www.learnaboutrobots.com/inverseKinematics.htm

- D.Tolani, A. Goswami, MI. Badler (2000), Real-Time Inverse Kinematics Techniques for Anthropomorphic Limbs, Graphical models 62(5),353-388

- JPA Dewald, V Sheshadri, ML Dawson, RF Beer (2004), Upper-Limb Discoordination in Hemiparetic Stroke: Implications for Neurorehabilitation, Stroke Rehabilitation, 1-12

Suggested literature

- Siciliano, B. (1999). "The Tricept robot: Inverse kinematics, manipulability analysis and closed-loop direct kinematics algorithm." Robotica 17: 437-445.
- Jiang, L., D. Sun, et al. (2009). "An Inverse-Kinematics Table-Based Solution of a Humanoid Robot Finger With Nonlinearly Coupled Joints." Ieee-Asme Transactions on Mechatronics 14(3): 273-281.
- Saglia, J. A., N. G. Tsagarakis, et al. (2009). "Inverse-kinematics-based control of a redundantly actuated platform for rehabilitation." Proceedings of the Institution of Mechanical Engineers Part I-Journal of Systems and Control Engineering 223(I1): 53-70.
- Yahya, S., M. Moghavvemi, et al. (2011). "Geometrical approach of planar hyper-redundant manipulators. Inverse kinematics, path planning and workspace." Simulation Modelling Practice and Theory 19(1): 406-422.

Suggested literature

- Unzueta, L., M. Peinado, et al. (2008). "Full-body performance animation with Sequential Inverse Kinematics." Graphical Models 70: 87-104.
- de Angulo, V. R. and C. Torras (2008). "Learning Inverse Kinematics: Reduced Sampling Through Decomposition Into Virtual Robots." Ieee Transactions on Systems Man and Cybernetics Part B-Cybernetics 38(6): 1571-1577.
- Neppalli, S., M. A. Csencsits, et al. (2009). "Closed-Form Inverse Kinematics for Continuum Manipulators." Advanced Robotics 23(15): 2077-2091.
- Van Henten, E. J., E. J. Schenk, et al. (2010). "Collision-free inverse kinematics of the redundant seven-link manipulator used in a cucumber picking robot." Biosystems Engineering 106(2): 112-124.