Inverse Kinematics Solutions using Conformal Geometric Algebra

Andreas Aristidou[†] and Joan Lasenby[‡]
Signal Processing and Communications Group,
Department of Engineering,
University of Cambridge
email: [†]aa462@cam.ac.uk and [‡]jl221@cam.ac.uk

June 24, 2010

Table of contents

Introduction

Inverse Kinematics (IK) is defined as the problem of determining a set of appropriate joint configurations for which the end effectors move to desired positions as smoothly, rapidly and as accurately as possible.

It finds application in:

Introduction

Inverse Kinematics (IK) is defined as the problem of determining a set of appropriate joint configurations for which the end effectors move to desired positions as smoothly, rapidly and as accurately as possible.

It finds application in:

- Robotics,

Introduction

Inverse Kinematics (IK) is defined as the problem of determining a set of appropriate joint configurations for which the end effectors move to desired positions as smoothly, rapidly and as accurately as possible.

It finds application in:

- Robotics,
- Computer animation,

Introduction

Inverse Kinematics (IK) is defined as the problem of determining a set of appropriate joint configurations for which the end effectors move to desired positions as smoothly, rapidly and as accurately as possible.

It finds application in:

- Robotics,
- Computer animation,
- Ergonomics,

Introduction

Inverse Kinematics (IK) is defined as the problem of determining a set of appropriate joint configurations for which the end effectors move to desired positions as smoothly, rapidly and as accurately as possible.

It finds application in:

- Robotics,
- Computer animation,
- Ergonomics,
- Computer games industry,

Introduction

Inverse Kinematics (IK) is defined as the problem of determining a set of appropriate joint configurations for which the end effectors move to desired positions as smoothly, rapidly and as accurately as possible.

It finds application in:

- Robotics,
- Computer animation,
- Ergonomics,
- Computer games industry,
- Human Computer Interaction. (HCI)

Introduction

Inverse Kinematics (IK) is defined as the problem of determining a set of appropriate joint configurations for which the end effectors move to desired positions as smoothly, rapidly and as accurately as possible.

It finds application in:

- Robotics,
- Computer animation,
- Ergonomics,
- Computer games industry,
- Human Computer Interaction. (HCI)

Related Work

The most popular Inverse Kinematic solvers are:

- Inverse Jacobian methods (Pseudo-inverse, SVD, DLS-SVD, SDLS),

Related Work

The most popular Inverse Kinematic solvers are:

- Inverse Jacobian methods (Pseudo-inverse, SVD, DLS-SVD, SDLS),
- Newton Methods,

Related Work

The most popular Inverse Kinematic solvers are:

- Inverse Jacobian methods (Pseudo-inverse, SVD, DLS-SVD, SDLS),
- Newton Methods,
- Mesh based methods (Learning Space),

Related Work

The most popular Inverse Kinematic solvers are:

- Inverse Jacobian methods (Pseudo-inverse, SVD, DLS-SVD, SDLS),
- Newton Methods,
- Mesh based methods (Learning Space),
- Statistical Methods (SMCM, Particle Filtering, IK from a control prospective),

Related Work

The most popular Inverse Kinematic solvers are:

- Inverse Jacobian methods (Pseudo-inverse, SVD, DLS-SVD, SDLS),
- Newton Methods,
- Mesh based methods (Learning Space),
- Statistical Methods (SMCM, Particle Filtering, IK from a control prospective),
- Heuristic Methods (CCD).

Related Work

The most popular Inverse Kinematic solvers are:

- Inverse Jacobian methods (Pseudo-inverse, SVD, DLS-SVD, SDLS),
- Newton Methods,
- Mesh based methods (Learning Space),
- Statistical Methods (SMCM, Particle Filtering, IK from a control prospective),
- Heuristic Methods (CCD).

Related Work in Conformal Geometric Algebra

- Dorst et al, [1], gives a simple framework solution for a robot arm,
- Corrochano and Kähler [2] used a language of points, lines and planes (which are later replaced by spheres in [3]) to solve the IK problem of a robot arm within a framework,

Related Work in Conformal Geometric Algebra

- Dorst et al, [1], gives a simple framework solution for a robot arm,
- Corrochano and Kähler [2] used a language of points, lines and planes (which are later replaced by spheres in [3]) to solve the IK problem of a robot arm within a framework,
- [4, 5, 6] used CGA to deal with forward kinematics, dynamics and projective geometry problems,

Related Work in Conformal Geometric Algebra

- Dorst et al, [1], gives a simple framework solution for a robot arm,
- Corrochano and Kähler [2] used a language of points, lines and planes (which are later replaced by spheres in [3]) to solve the IK problem of a robot arm within a framework,
- [4, 5, 6] used CGA to deal with forward kinematics, dynamics and projective geometry problems,
- Tanev, in [7], described an application of CGA to the analysis of a parallel manipulator with limited mobility,

Related Work in Conformal Geometric Algebra

- Dorst et al, [1], gives a simple framework solution for a robot arm,
- Corrochano and Kähler [2] used a language of points, lines and planes (which are later replaced by spheres in [3]) to solve the IK problem of a robot arm within a framework,
- [4, 5, 6] used CGA to deal with forward kinematics, dynamics and projective geometry problems,
- Tanev, in [7], described an application of CGA to the analysis of a parallel manipulator with limited mobility,
- Hildenbrand et al, [8], gives a synopsis of different IK framework solutions and grasping processes of a robot arm.

Related Work in Conformal Geometric Algebra

- Dorst et al, [1], gives a simple framework solution for a robot arm,
- Corrochano and Kähler [2] used a language of points, lines and planes (which are later replaced by spheres in [3]) to solve the IK problem of a robot arm within a framework,
- [4, 5, 6] used CGA to deal with forward kinematics, dynamics and projective geometry problems,
- Tanev, in [7], described an application of CGA to the analysis of a parallel manipulator with limited mobility,
- Hildenbrand et al, [8], gives a synopsis of different IK framework solutions and grasping processes of a robot arm.

However, most of the literature:

- solves simple kinematic problems of a robot arm with 5 degrees of freedom (DoF)

Related Work in Conformal Geometric Algebra

- Dorst et al, [1], gives a simple framework solution for a robot arm,
- Corrochano and Kähler [2] used a language of points, lines and planes (which are later replaced by spheres in [3]) to solve the IK problem of a robot arm within a framework,
- [4, 5, 6] used CGA to deal with forward kinematics, dynamics and projective geometry problems,
- Tanev, in [7], described an application of CGA to the analysis of a parallel manipulator with limited mobility,
- Hildenbrand et al, [8], gives a synopsis of different IK framework solutions and grasping processes of a robot arm.

However, most of the literature:

- solves simple kinematic problems of a robot arm with 5 degrees of freedom (DoF)
- describes how to constrain the movement of a manipulator to a feasible set within a framework rather than describing a solver itself.

Related Work in Conformal Geometric Algebra

- Dorst et al, [1], gives a simple framework solution for a robot arm,
- Corrochano and Kähler [2] used a language of points, lines and planes (which are later replaced by spheres in [3]) to solve the IK problem of a robot arm within a framework,
- [4, 5, 6] used CGA to deal with forward kinematics, dynamics and projective geometry problems,
- Tanev, in [7], described an application of CGA to the analysis of a parallel manipulator with limited mobility,
- Hildenbrand et al, [8], gives a synopsis of different IK framework solutions and grasping processes of a robot arm.

However, most of the literature:

- solves simple kinematic problems of a robot arm with 5 degrees of freedom (DoF)
- describes how to constrain the movement of a manipulator to a feasible set within a framework rather than describing a solver itself.

FABRIK

FABRIK (Forward And Backward Reaching Inverse Kinematics):

- Solves the IK problem in an iterative fashion,
- Divides the problem into 2 phases, a forward and a backward reaching approach,

FABRIK

FABRIK (Forward And Backward Reaching Inverse Kinematics):

- Solves the IK problem in an iterative fashion,
- Divides the problem into 2 phases, a forward and a backward reaching approach,
- Uses points, lines and spheres to solve the IK problem instead of rotational angles.

FABRIK

FABRIK (Forward And Backward Reaching Inverse Kinematics):

- Solves the IK problem in an iterative fashion,
- Divides the problem into 2 phases, a forward and a backward reaching approach,
- Uses points, lines and spheres to solve the IK problem instead of rotational angles.

FABRIK

- Able to treat most of the joint types,
- Supports biomechanical constraints on chains with both single and multiple end effectors,

FABRIK

- Able to treat most of the joint types,
- Supports biomechanical constraints on chains with both single and multiple end effectors,
- Low computational cost $\rightarrow$ Real-Time implementable,

FABRIK

- Able to treat most of the joint types,
- Supports biomechanical constraints on chains with both single and multiple end effectors,
- Low computational cost $\rightarrow$ Real-Time implementable,
- Tracks the target smoothly without oscillations or discontinuities,

FABRIK

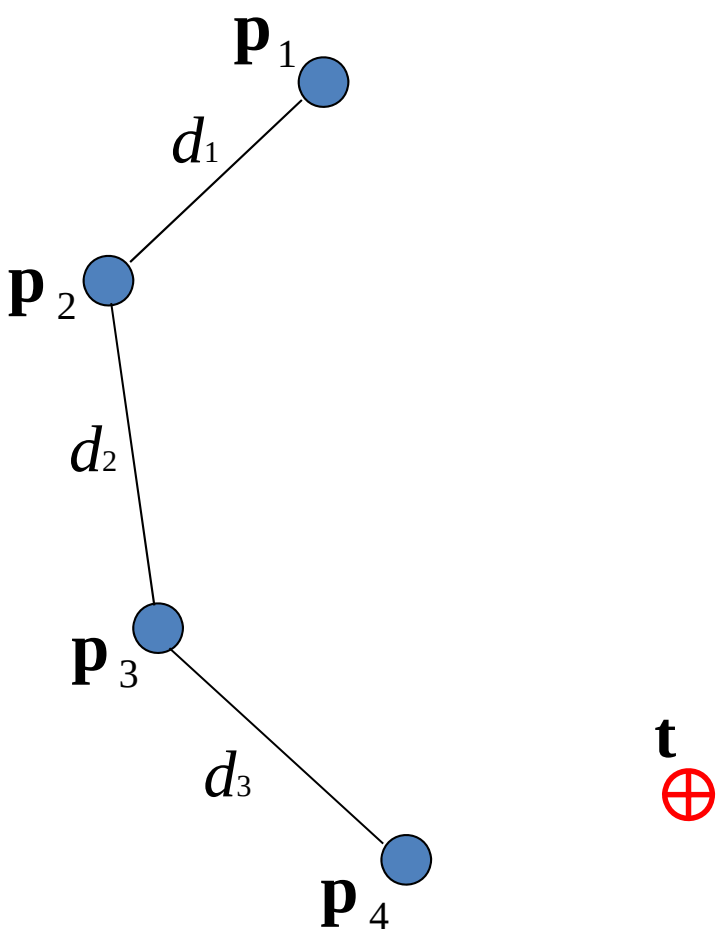
- Able to treat most of the joint types,
- Supports biomechanical constraints on chains with both single and multiple end effectors,
- Low computational cost $\rightarrow$ Real-Time implementable,
- Tracks the target smoothly without oscillations or discontinuities,
- Produce natural poses.

FABRIK

- Able to treat most of the joint types,
- Supports biomechanical constraints on chains with both single and multiple end effectors,
- Low computational cost $\rightarrow$ Real-Time implementable,
- Tracks the target smoothly without oscillations or discontinuities,
- Produce natural poses.

FABRIK: The Algorithm

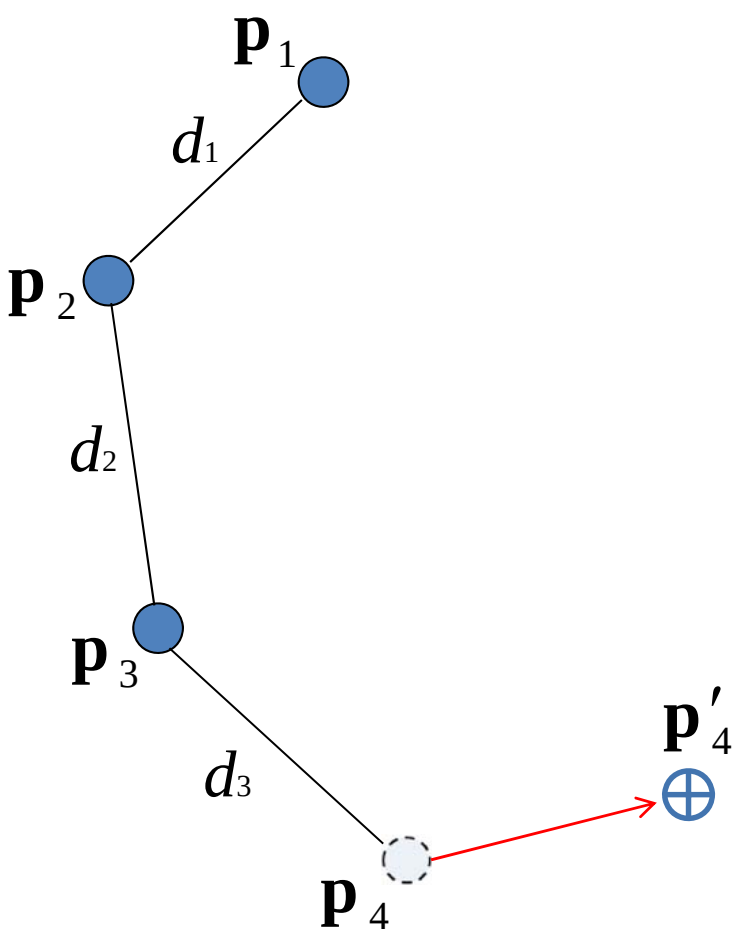
Forward Reaching Step



The initial position of the manipulator and the target.

FABRIK: The Algorithm

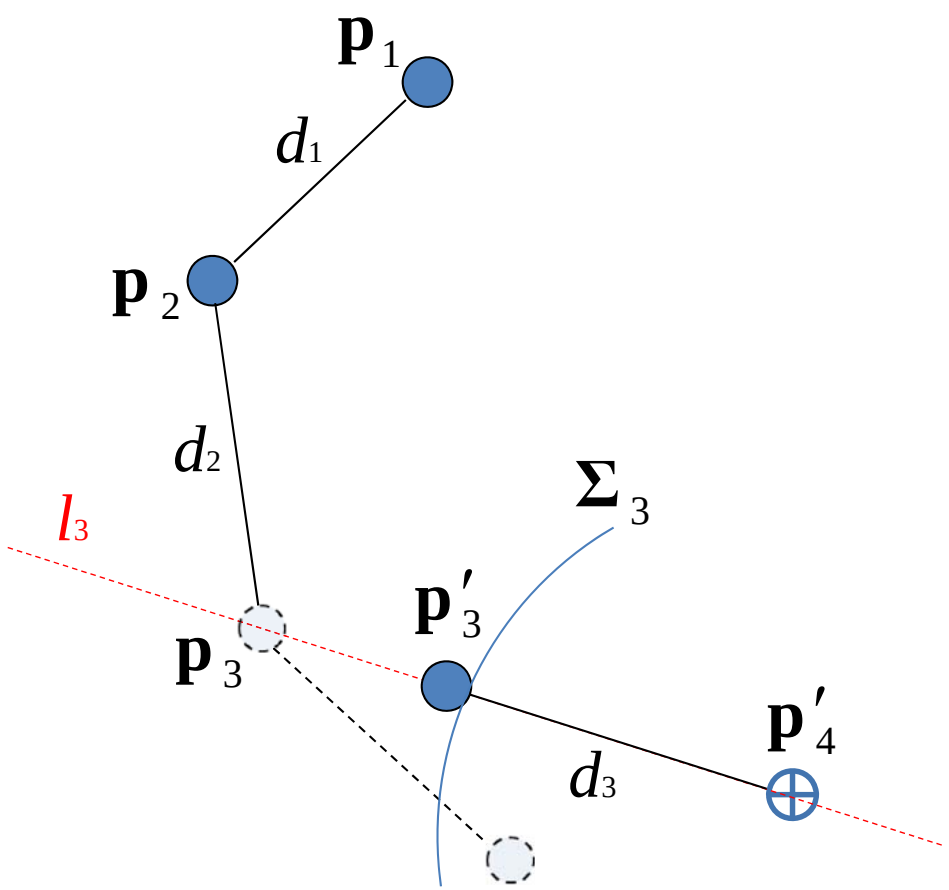
Forward Reaching Step



Move the end effector $\mathbf{p}_4$ to the target.

FABRIK: The Algorithm

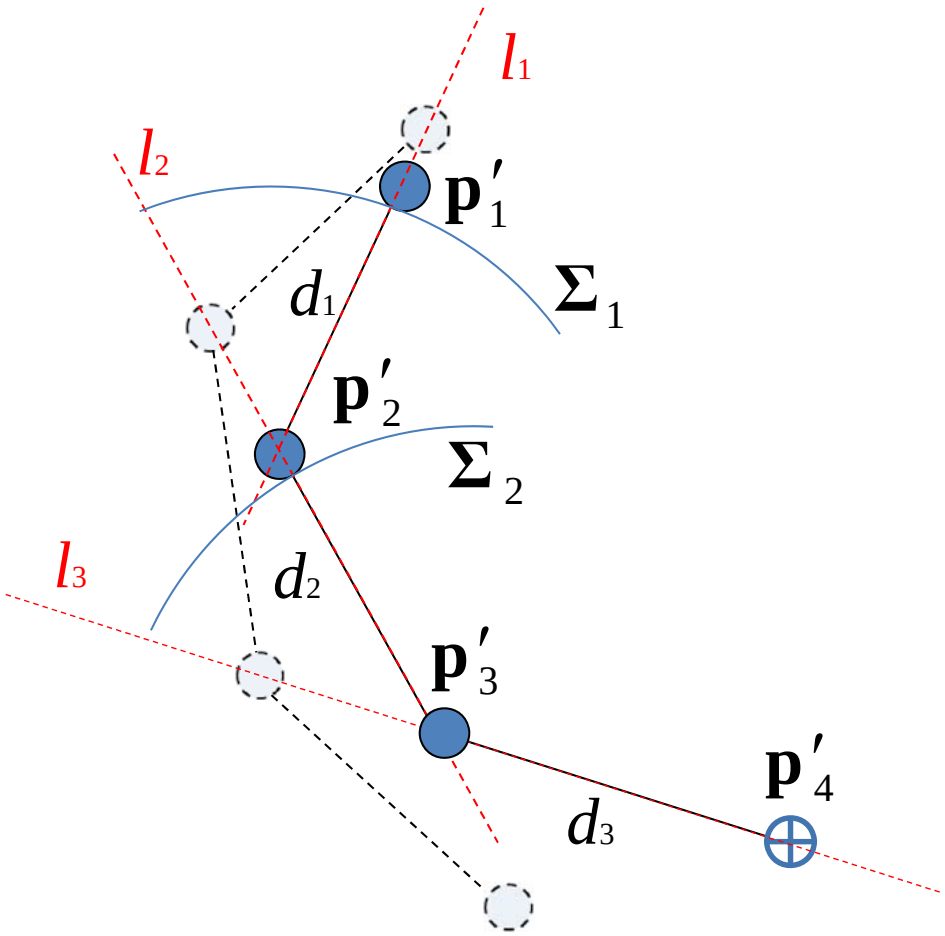
Forward Reaching Step



Find the joint $\mathbf{p}'_3$ which is the intersection of the sphere Σ_3 and the line l_3 which passes through the points $\mathbf{p}'_4$ and $\mathbf{p}_3$.

FABRIK: The Algorithm

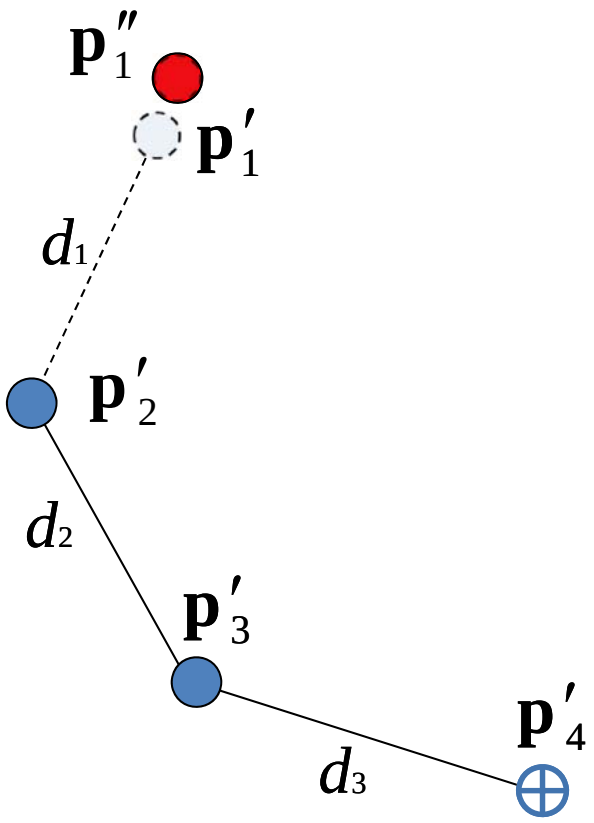
Forward Reaching Step



Continue the algorithm for the rest of the joints.

FABRIK: The Algorithm

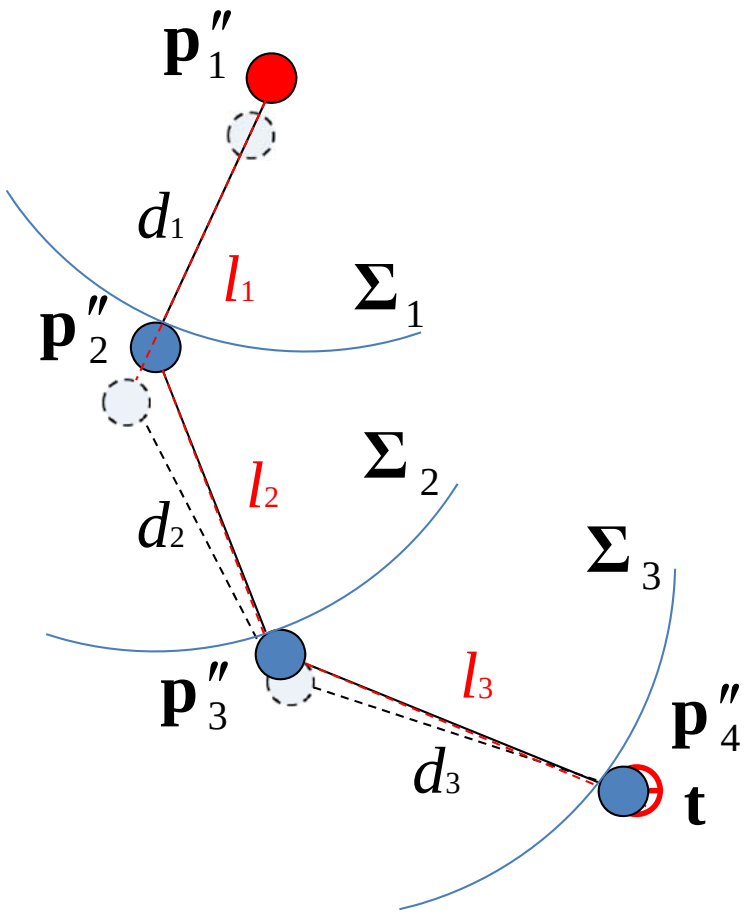
Backward Reaching Step



The second stage of the algorithm: move the root joint $\mathbf{p}'_1$ to its initial position.

FABRIK: The Algorithm

Backward Reaching Step

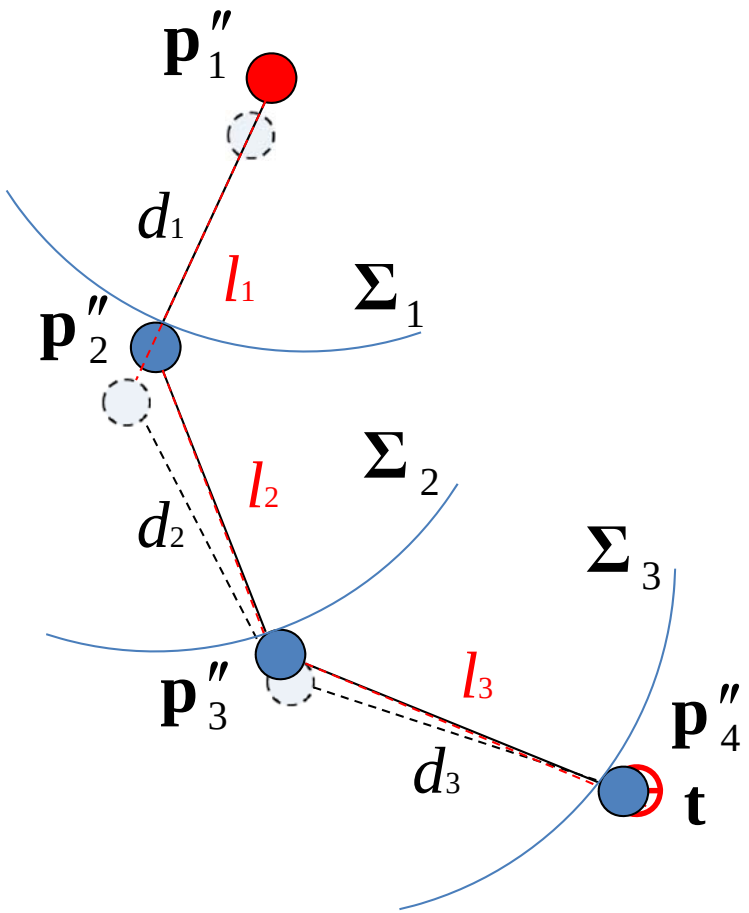


Repeat the same procedure but this time start from the base and move outwards to the end effector.

The algorithm is repeated until the position of the end effector reaches the target or gets sufficiently close.

FABRIK: The Algorithm

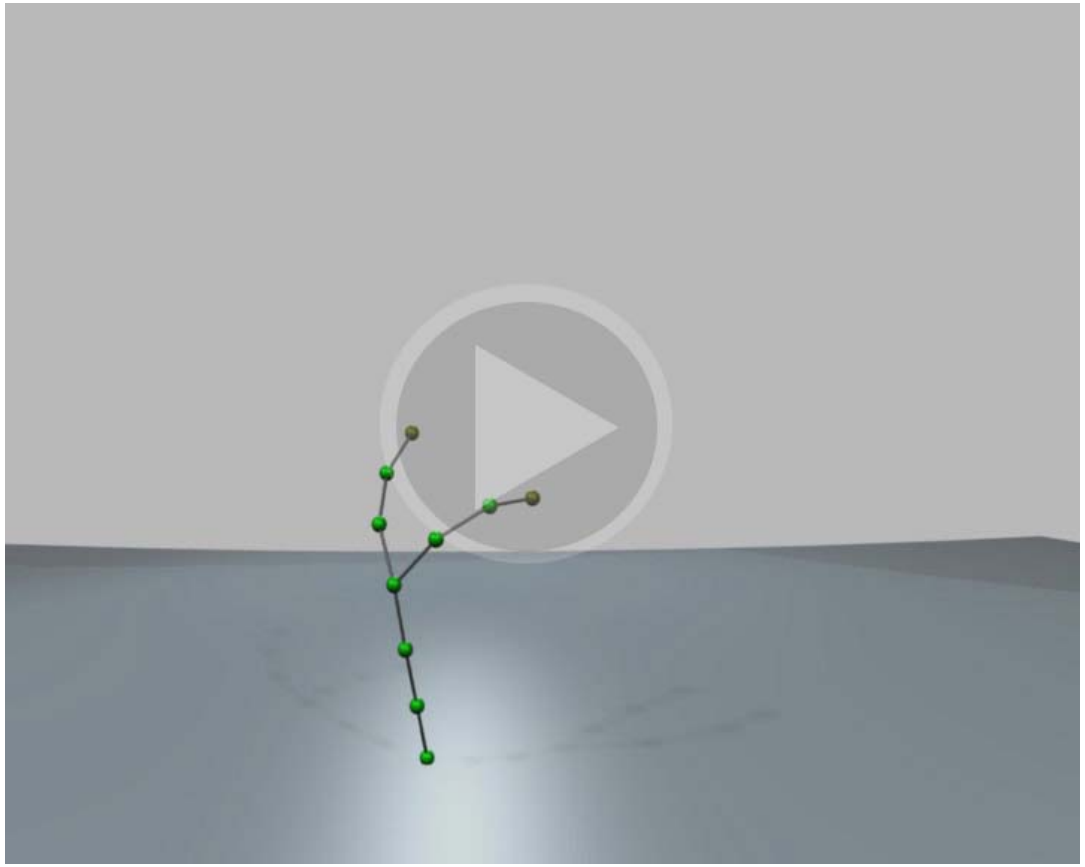
Backward Reaching Step



Repeat the same procedure but this time start from the base and move outwards to the end effector.

The algorithm is repeated until the position of the end effector reaches the target or gets sufficiently close.

FABRIK video example



Hand tracking in optical motion capture

An application of the proposed algorithm that demonstrates all of FABRIK's advantages.

- Hand Tracking and reconstruction using optical motion capture,

Hand tracking in optical motion capture

An application of the proposed algorithm that demonstrates all of FABRIK's advantages.

- Hand Tracking and reconstruction using optical motion capture,
- The hand is constrained using Conformal Geometric Algebra,

Hand tracking in optical motion capture

An application of the proposed algorithm that demonstrates all of FABRIK's advantages.

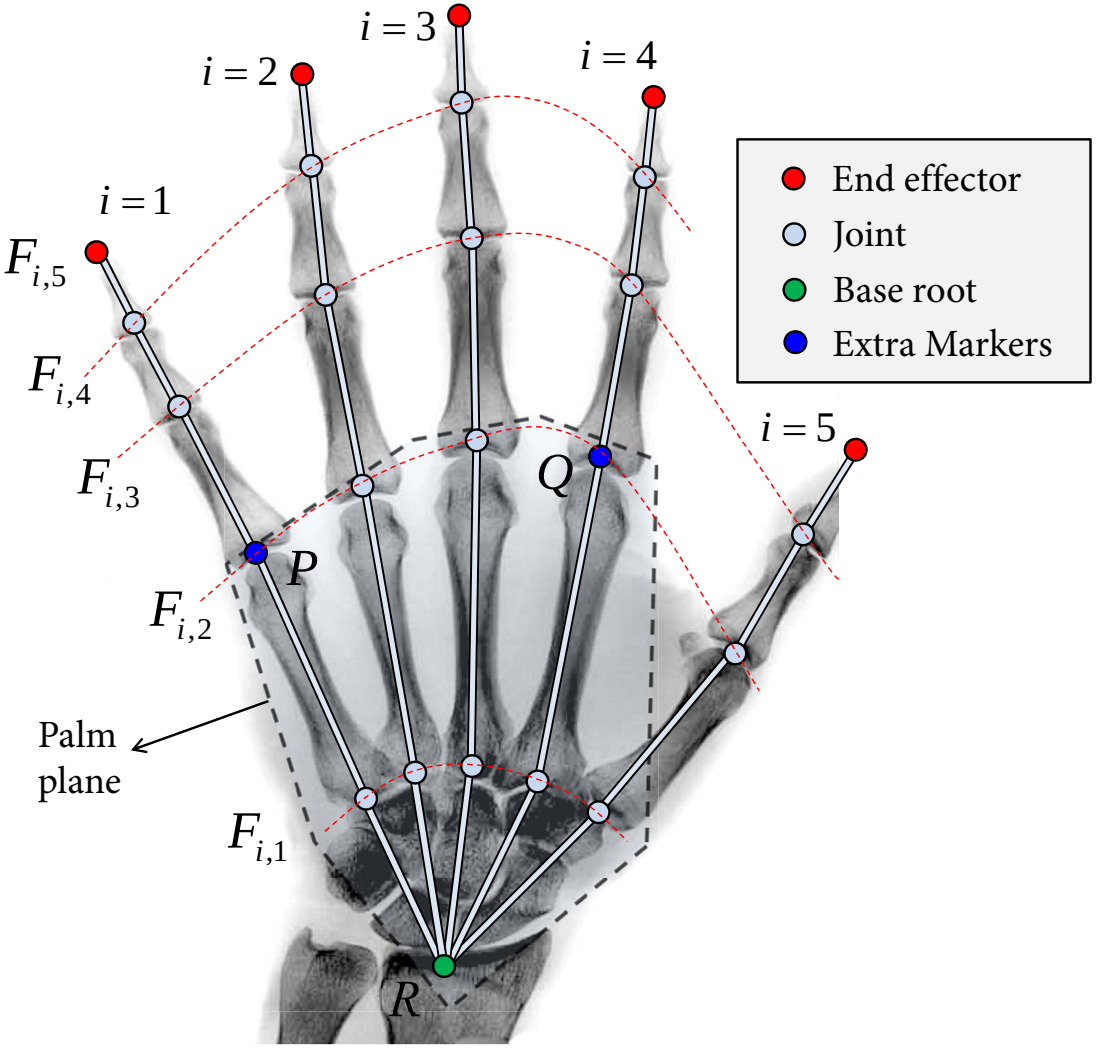
- Hand Tracking and reconstruction using optical motion capture,
- The hand is constrained using Conformal Geometric Algebra,
- The resulting poses are produced in real-time.

Hand tracking in optical motion capture

An application of the proposed algorithm that demonstrates all of FABRIK's advantages.

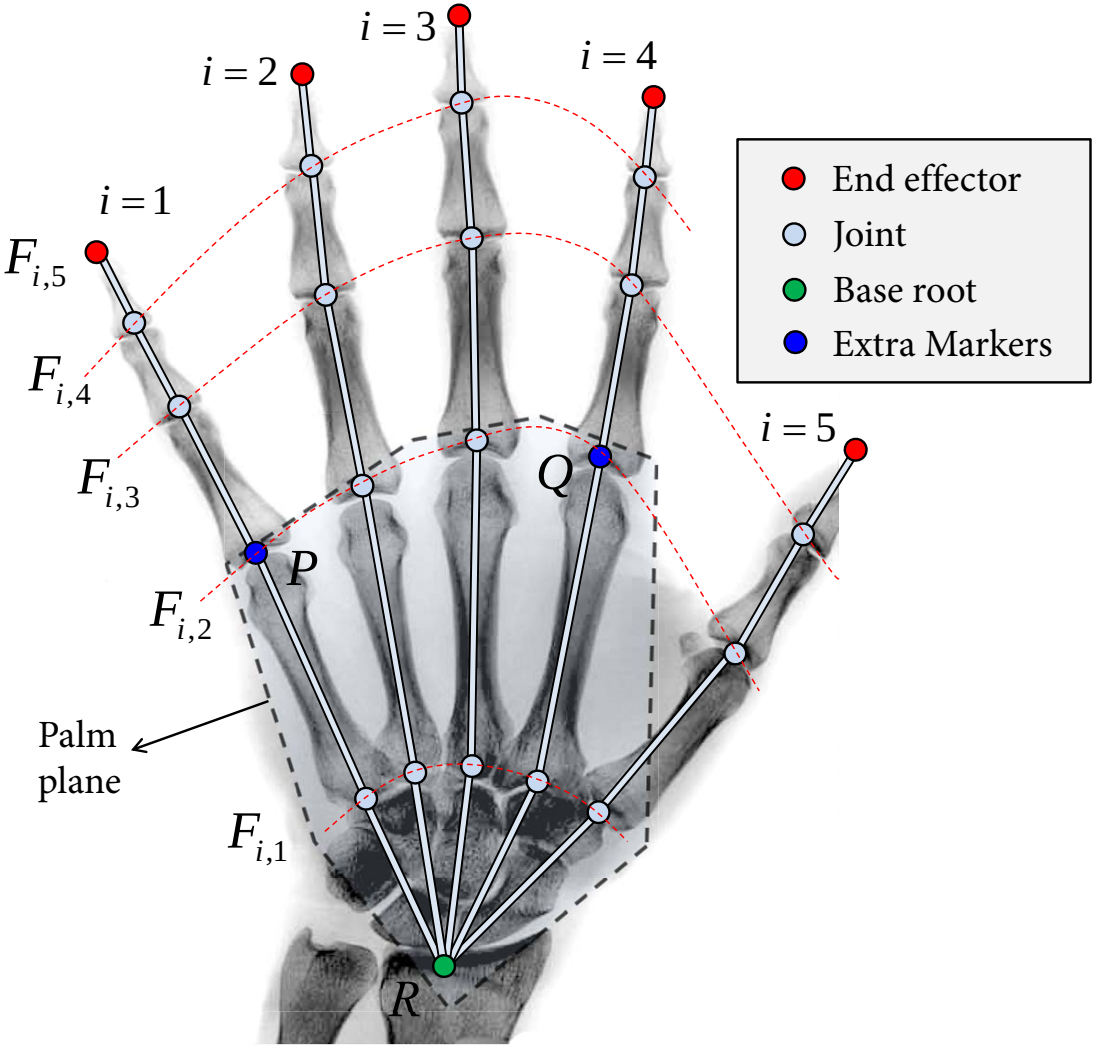
- Hand Tracking and reconstruction using optical motion capture,
- The hand is constrained using Conformal Geometric Algebra,
- The resulting poses are produced in real-time.

The Hand Model



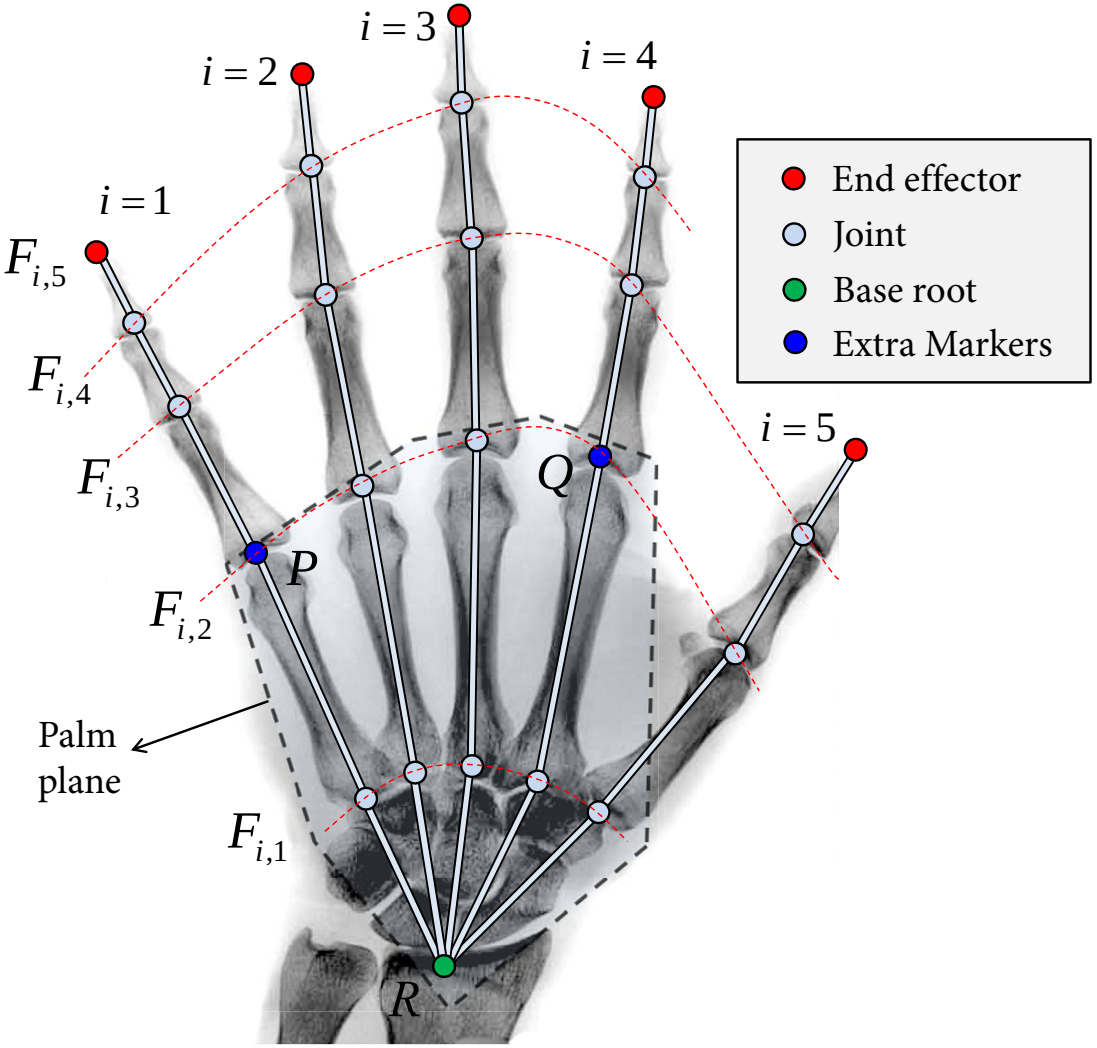
- 5 markers are attached at the end of each finger (end effectors),
- 1 marker is attached at the hand root,

The Hand Model



- 5 markers are attached at the end of each finger (end effectors),
- 1 marker is attached at the hand root,
- 2 markers are attached at strategic points on the palm to help us find the hand orientation.

The Hand Model



- 5 markers are attached at the end of each finger (end effectors),
- 1 marker is attached at the hand root,
- 2 markers are attached at strategic points on the palm to help us find the hand orientation.

The Hand Model

Assumptions:

- The hand orientation is given by the palm,

The Hand Model

Assumptions:

- The hand orientation is given by the palm,
- The palm plane is flat $\rightarrow$ the inter-joint distances are constant over time,

The Hand Model

Assumptions:

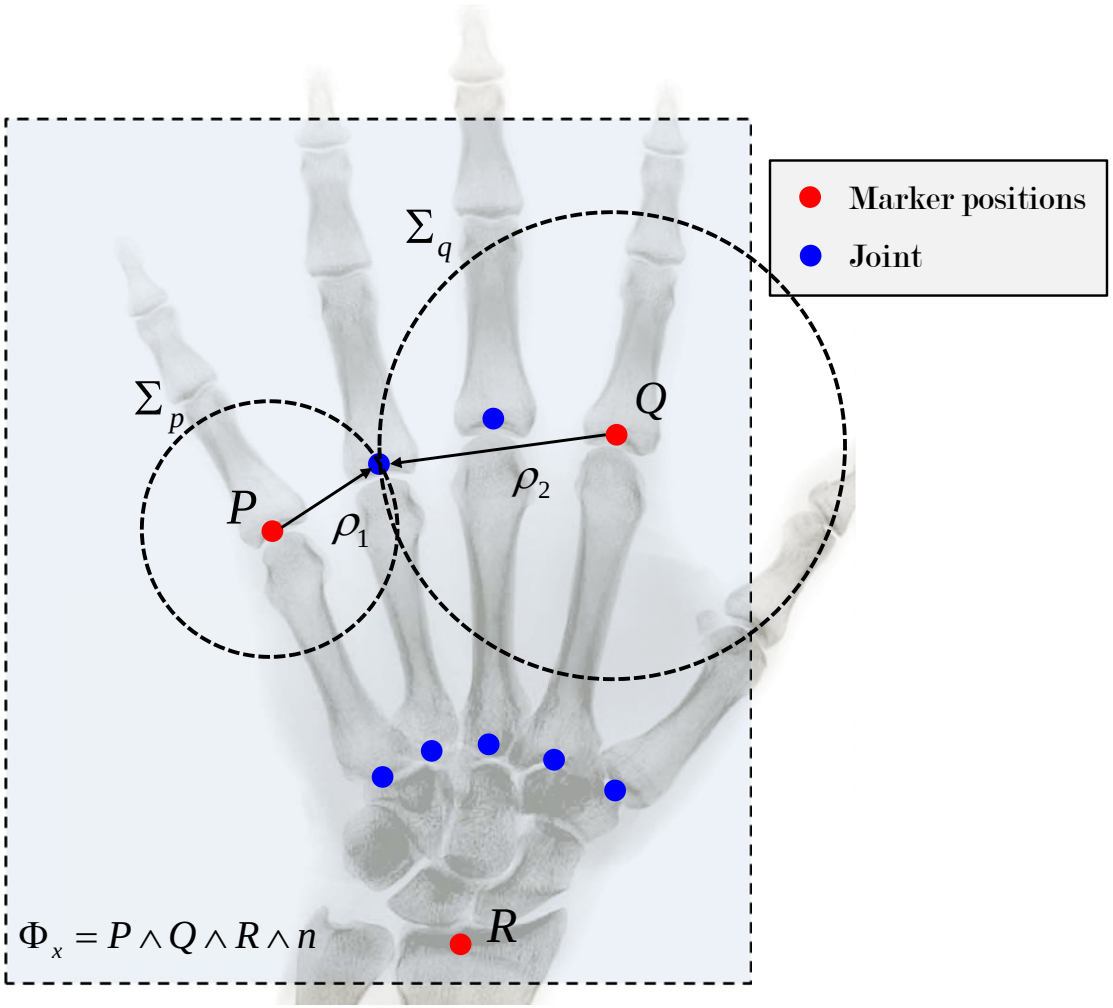
- The hand orientation is given by the palm,
- The palm plane is flat $\rightarrow$ the inter-joint distances are constant over time,
- Finger planes are perpendicular to the palm plane.

The Hand Model

Assumptions:

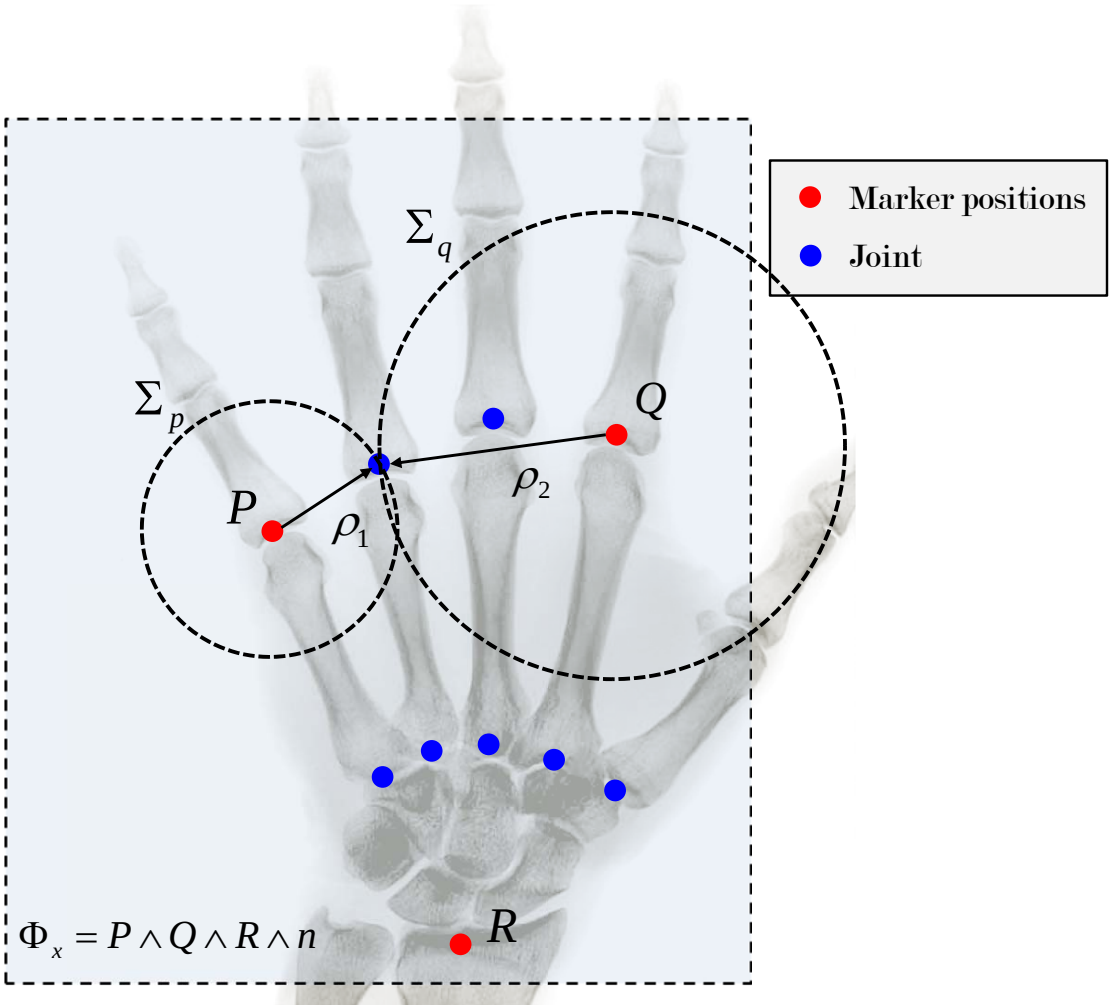
- The hand orientation is given by the palm,
- The palm plane is flat $\rightarrow$ the inter-joint distances are constant over time,
- Finger planes are perpendicular to the palm plane.

Calculating the palm joints



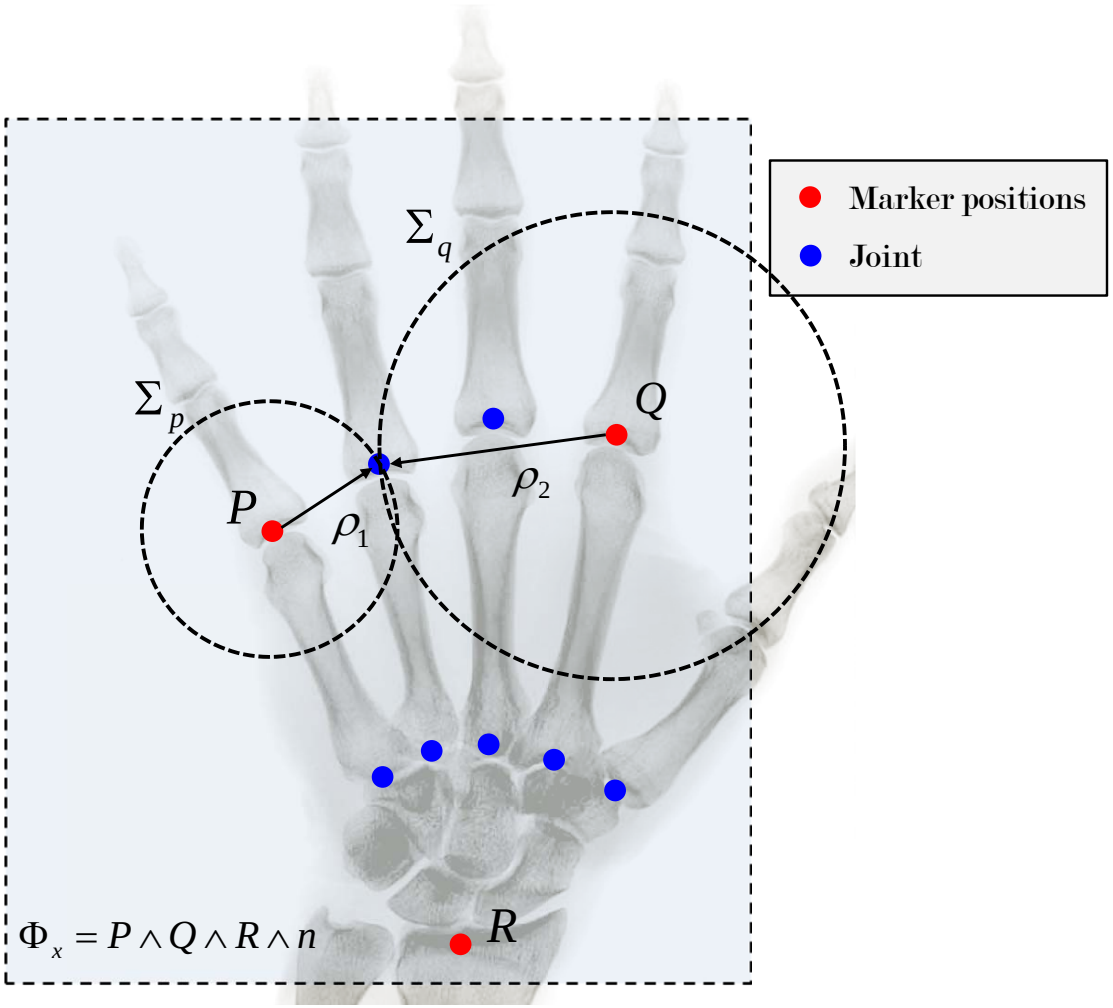
- Find the Spheres Σ_p and Σ_q,
- Find the intersection of the spheres $C = \Sigma_p \vee \Sigma_q$,

Calculating the palm joints



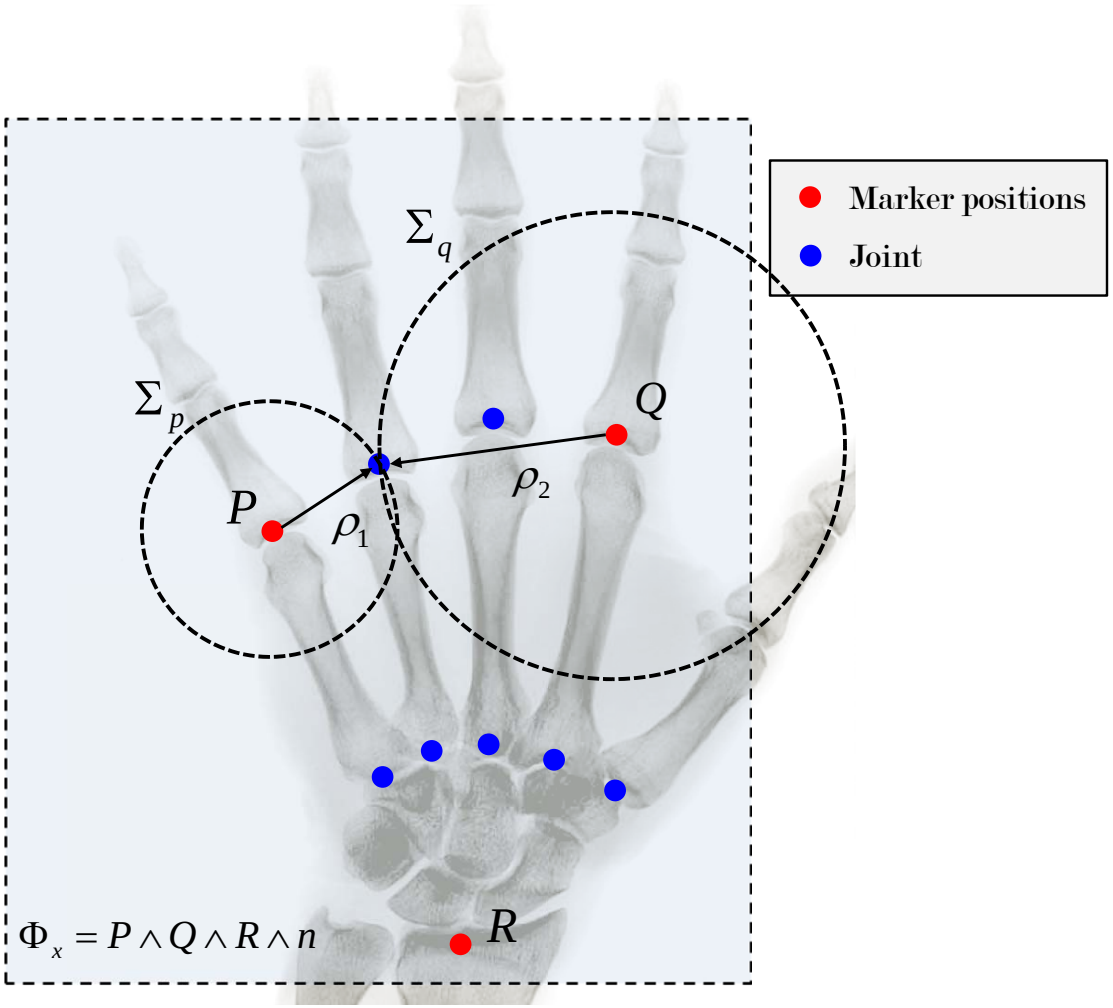
- Find the Spheres Σ_p and Σ_q,
- Find the intersection of the spheres $C = \Sigma_p \vee \Sigma_q$,
- Find the intersection between the entity C and the palm plane Φ_x, $B = C \vee \Phi_x$,

Calculating the palm joints



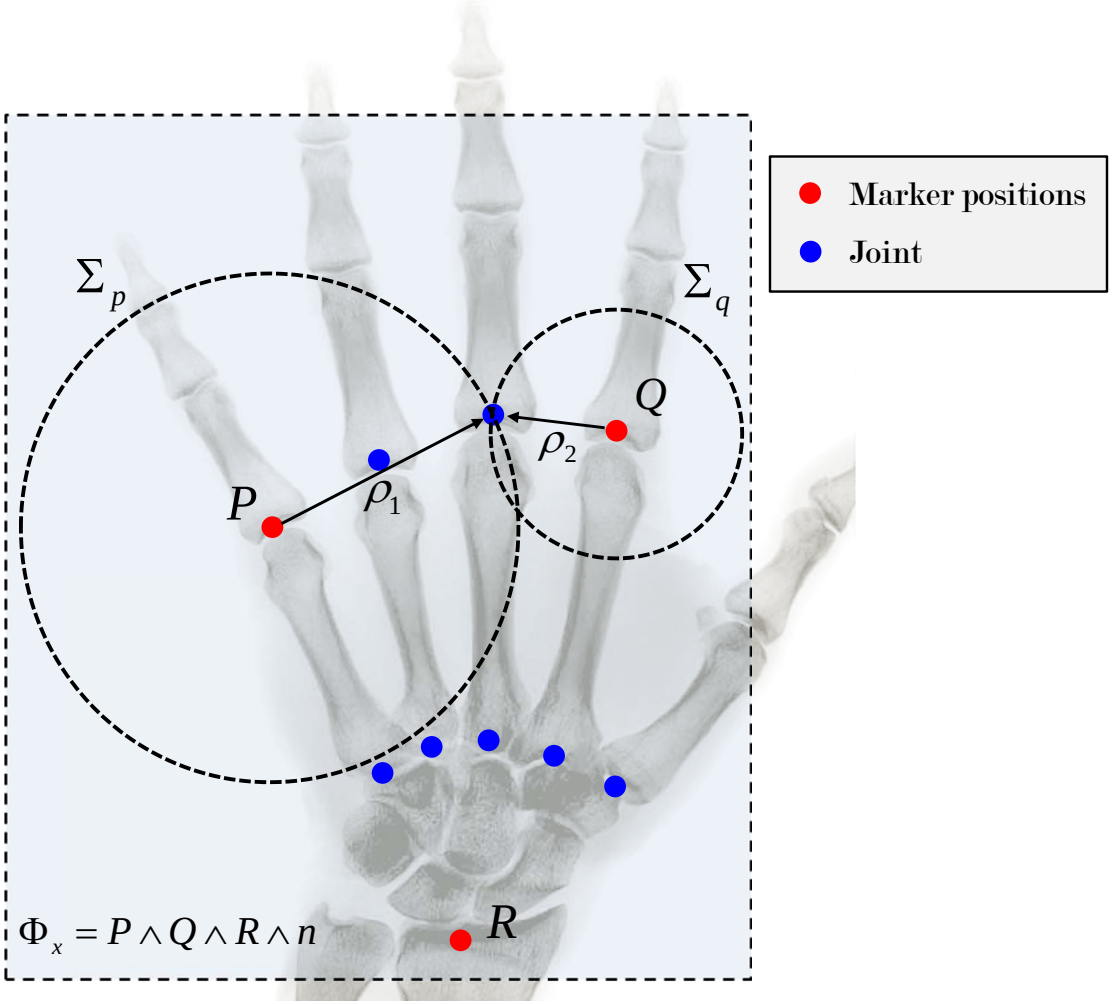
- Find the Spheres Σ_p and Σ_q,
- Find the intersection of the spheres $C = \Sigma_p \vee \Sigma_q$,
- Find the intersection between the entity C and the palm plane Φ_x, $B = C \vee \Phi_x$,
- Choose the point that is closer to its previous position.

Calculating the palm joints



- Find the Spheres Σ_p and Σ_q,
- Find the intersection of the spheres $C = \Sigma_p \vee \Sigma_q$,
- Find the intersection between the entity C and the palm plane Φ_x, $B = C \vee \Phi_x$,
- Choose the point that is closer to its previous position.

Calculating the palm joints



- Similarly for all the remaining palm joints

Calculating the finger joints

Find the finger planes Φ_i, for $i = 1, \ldots, 4$

- Φ_i can be calculated using the known joint positions $F_{i,2}$, the marker positions $F_{i,5}$ and by assuming that the finger planes are perpendicular to the palm plane Φ_x (apart from the thumb plane Φ_5),

Calculating the finger joints

Find the finger planes Φ_i, for $i = 1, \ldots, 4$

- Φ_i can be calculated using the known joint positions $F_{i,2}$, the marker positions $F_{i,5}$ and by assuming that the finger planes are perpendicular to the palm plane Φ_x (apart from the thumb plane Φ_5),
- Both points from each finger are known (the motion capture system tracks the end effector positions $F_{i,5}$ and the finger roots $F_{i,2}$ lie on the palm plane),

Calculating the finger joints

Find the finger planes Φ_i, for $i = 1, \ldots, 4$

- Φ_i can be calculated using the known joint positions $F_{i,2}$, the marker positions $F_{i,5}$ and by assuming that the finger planes are perpendicular to the palm plane Φ_x (apart from the thumb plane Φ_5),
- Both points from each finger are known (the motion capture system tracks the end effector positions $F_{i,5}$ and the finger roots $F_{i,2}$ lie on the palm plane),
- The vector that is perpendicular to the hand plane Φ_x, $\hat{n} = \Phi_x^* - \frac{1}{2}\left(\Phi_x^* \cdot \bar{n}\right) n$,

Calculating the finger joints

Find the finger planes Φ_i, for $i = 1, \ldots, 4$

- Φ_i can be calculated using the known joint positions $F_{i,2}$, the marker positions $F_{i,5}$ and by assuming that the finger planes are perpendicular to the palm plane Φ_x (apart from the thumb plane Φ_5),
- Both points from each finger are known (the motion capture system tracks the end effector positions $F_{i,5}$ and the finger roots $F_{i,2}$ lie on the palm plane),
- The vector that is perpendicular to the hand plane Φ_x, $\hat{n} = \Phi_x^* - \frac{1}{2}\left(\Phi_x^* \cdot \bar{n}\right) n$,
- $\Phi_i = F_{i,2} \wedge F_{i,5} \wedge \hat{n} \wedge n$ for $i = 1, \ldots, 4$.

Calculating the finger joints

Find the finger planes Φ_i, for $i = 1, \ldots, 4$

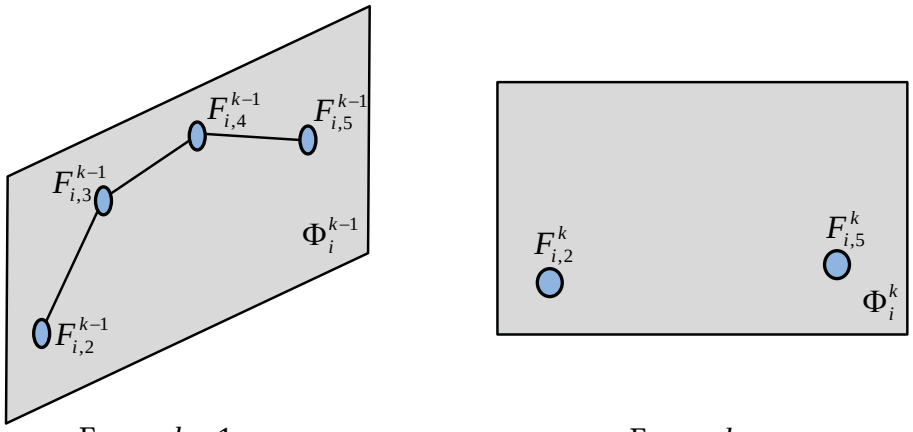
- Φ_i can be calculated using the known joint positions $F_{i,2}$, the marker positions $F_{i,5}$ and by assuming that the finger planes are perpendicular to the palm plane Φ_x (apart from the thumb plane Φ_5),
- Both points from each finger are known (the motion capture system tracks the end effector positions $F_{i,5}$ and the finger roots $F_{i,2}$ lie on the palm plane),
- The vector that is perpendicular to the hand plane Φ_x, $\hat{n} = \Phi_x^* - \frac{1}{2}\left(\Phi_x^* \cdot \bar{n}\right) n$,
- $\Phi_i = F_{i,2} \wedge F_{i,5} \wedge \hat{n} \wedge n$ for $i = 1, ..., 4$.

Calculating the finger joints

- The thumb orientation Φ_5 can be estimated using the marker position $F_{5,4}$, and the joints positions $F_{1,2}$ and $F_{5,2}$ that lie on the palm, assuming that when the thumb bends to the ventral side of the palm, it always points at the joint $F_{1,2}$.

Calculating the finger joints

When the finger planes are known

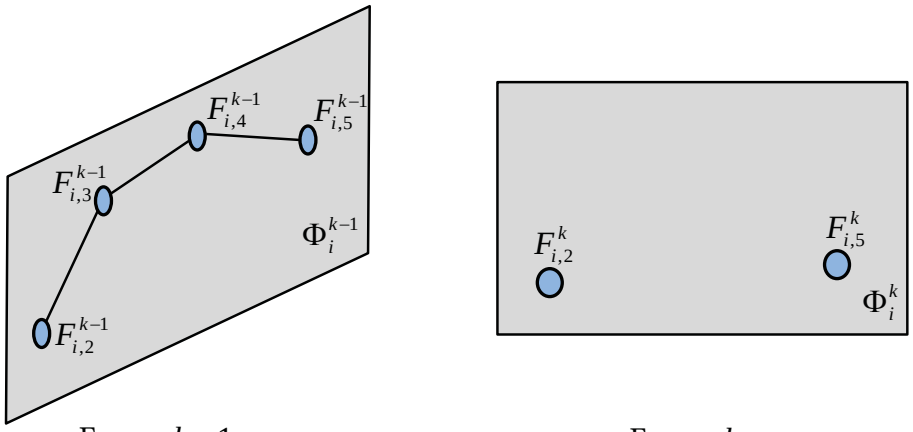


Frame $k-1$ Frame k

- Find the translation T from the previous to the current time,

Calculating the finger joints

When the finger planes are known

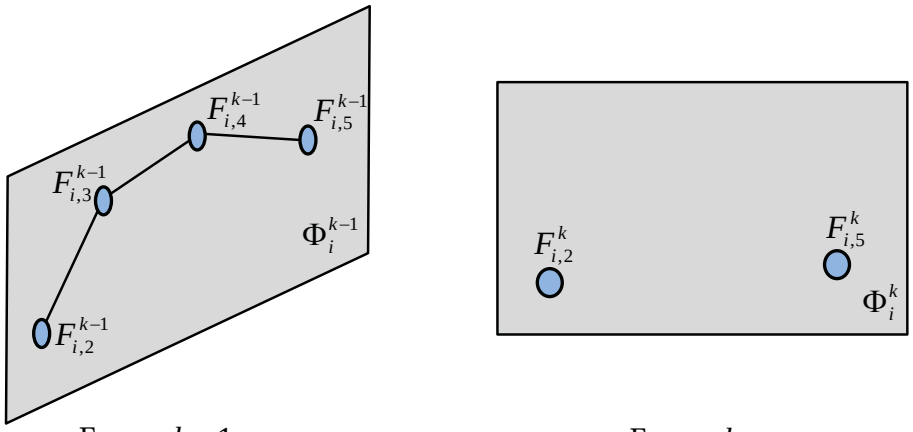


Frame $k-1$ Frame k

- Find the translation T from the previous to the current time,
- Find the rotor R that expresses the rotations between the previous and the current time,

Calculating the finger joints

When the finger planes are known

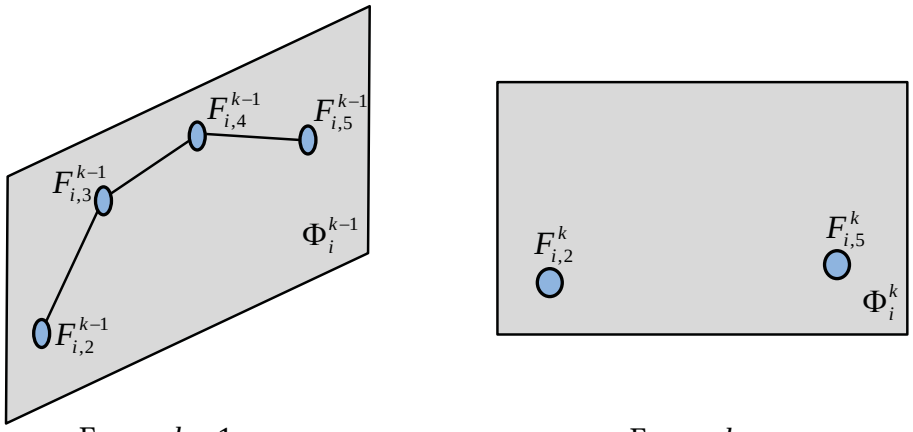


Frame $k-1$ Frame k

- Find the translation T from the previous to the current time,
- Find the rotor R that expresses the rotations between the previous and the current time,
- Translate and rotate each finger joint from the previous to the current time.

Calculating the finger joints

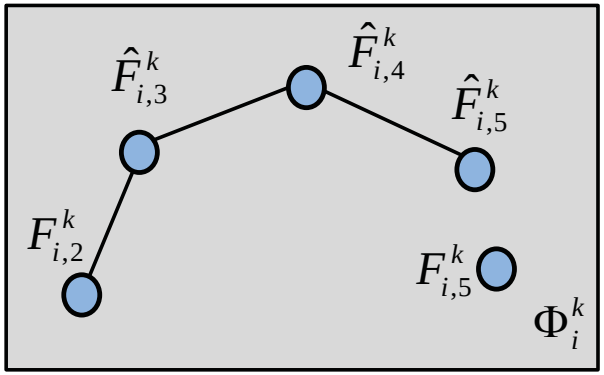
When the finger planes are known



Frame $k-1$ Frame k

- Find the translation T from the previous to the current time,
- Find the rotor R that expresses the rotations between the previous and the current time,
- Translate and rotate each finger joint from the previous to the current time.

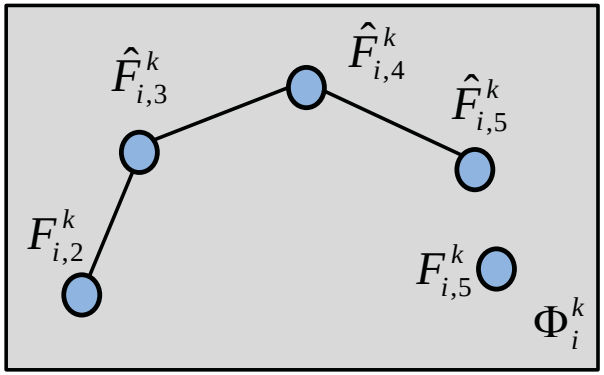
Calculating the finger joints



Frame k

- $\hat{F}^k_{i,j} = TRF^{k-1}_{i,j}\tilde{R}\tilde{T}$ where $i = 1, \ldots, 4$ and $j = 3, 4, 5$ (except for thumb $i = 5$ and $j = 2, 3, 4$),
- All joints now lie on plane Φ^k_i,

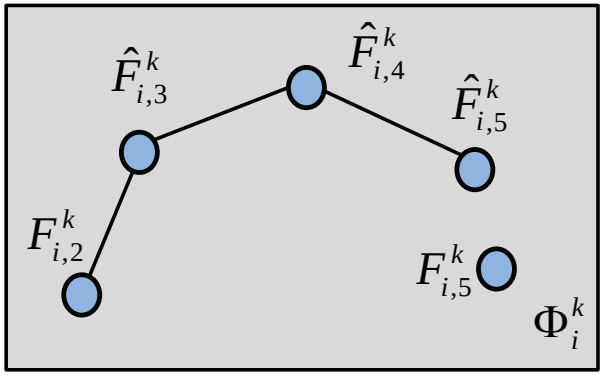
Calculating the finger joints



Frame k

- $\hat{F}^k_{i,j} = TRF^{k-1}_{i,j}\tilde{R}\tilde{T}$ where $i = 1, \ldots, 4$ and $j = 3, 4, 5$ (except for thumb $i = 5$ and $j = 2, 3, 4$),
- All joints now lie on plane Φ^k_i,
- FABRIK is applied to each finger chain, assuming that the root of the chain is $F^k_{i,2}$, the end effector is the rotated point $\hat{F}^k_{i,5}$ and the target is the current marker position $F^k_{i,5}$.

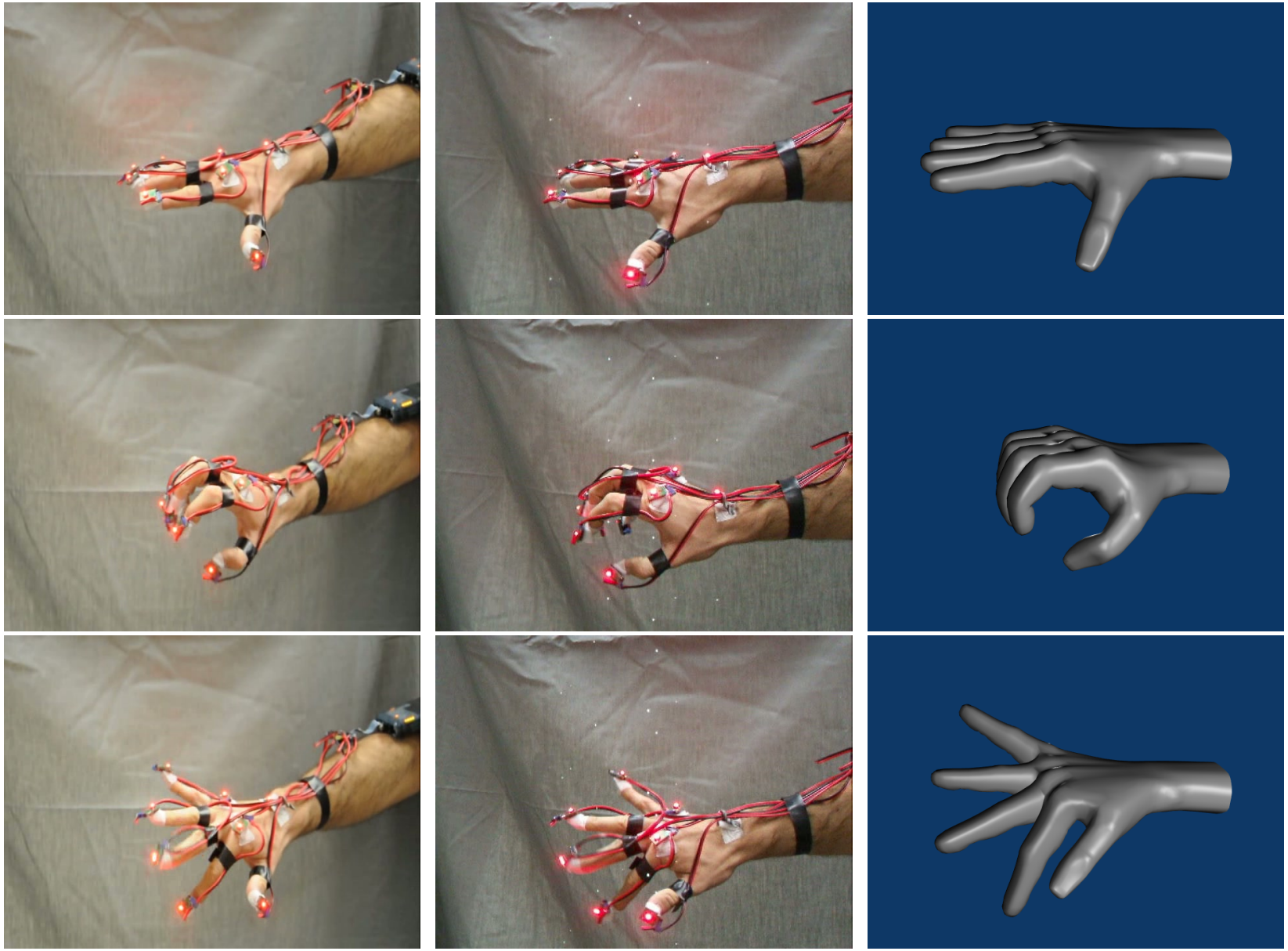
Calculating the finger joints



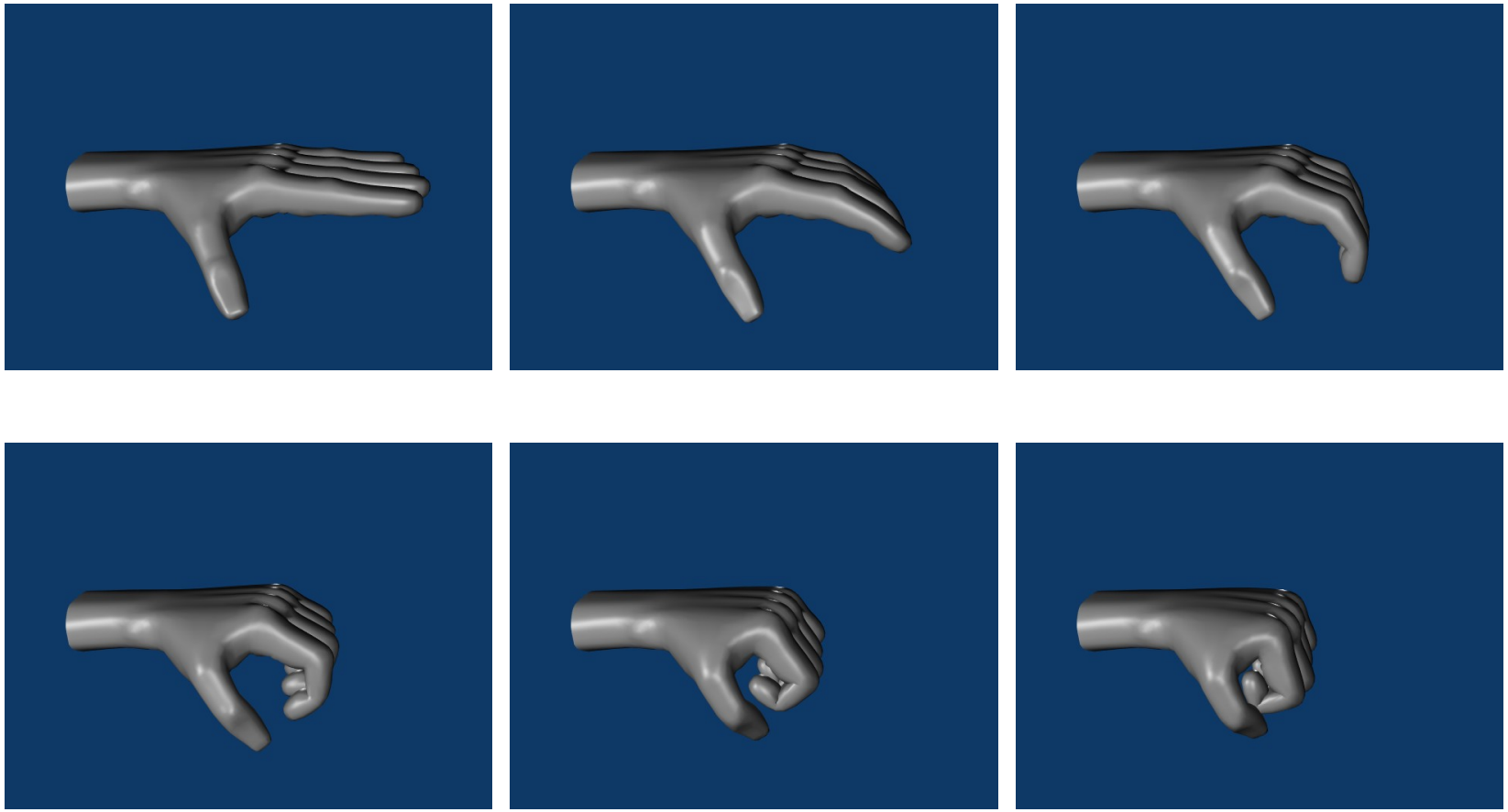
Frame k

- $\hat{F}^k_{i,j} = TRF^{k-1}_{i,j}\tilde{R}\tilde{T}$ where $i = 1, \ldots, 4$ and $j = 3, 4, 5$ (except for thumb $i = 5$ and $j = 2, 3, 4$),
- All joints now lie on plane Φ^k_i,
- FABRIK is applied to each finger chain, assuming that the root of the chain is $F^k_{i,2}$, the end effector is the rotated point $\hat{F}^k_{i,5}$ and the target is the current marker position $F^k_{i,5}$.

Results - Example 1



Results - Example 2



Results - Video

Conclusions

FABRIK:

- is real-time implementable and produces visually natural poses comparable with highly sophisticated analytic solutions,

Conclusions

FABRIK:

- is real-time implementable and produces visually natural poses comparable with highly sophisticated analytic solutions,
- is able to solve problems with multiple end effectors and targets,

Conclusions

FABRIK:

- is real-time implementable and produces visually natural poses comparable with highly sophisticated analytic solutions,
- is able to solve problems with multiple end effectors and targets,
- supports most joint types (apart from prismatic, slicing or shifting joints),

Conclusions

FABRIK:

- is real-time implementable and produces visually natural poses comparable with highly sophisticated analytic solutions,
- is able to solve problems with multiple end effectors and targets,
- supports most joint types (apart from prismatic, slicing or shifting joints),
- tracks the target without oscillations or discontinuities.

Conclusions

FABRIK:

- is real-time implementable and produces visually natural poses comparable with highly sophisticated analytic solutions,
- is able to solve problems with multiple end effectors and targets,
- supports most joint types (apart from prismatic, slicing or shifting joints),
- tracks the target without oscillations or discontinuities.

Conclusions

The proposed hand model:

- is real-time implementable,

Conclusions

The proposed hand model:

- is real-time implementable,
- produces natural movements within a feasible set,

Conclusions

The proposed hand model:

- is real-time implementable,
- produces natural movements within a feasible set,
- tracks the target without oscillations or discontinuities.

Conclusions

The proposed hand model:

- is real-time implementable,
- produces natural movements within a feasible set,
- tracks the target without oscillations or discontinuities.

Future Work

Future work will see the introduction of physiological constraints subject to the hand's anatomy:

1. Constraints related to:
   - finger's intradigital correlation,

Future Work

Future work will see the introduction of physiological constraints subject to the hand's anatomy:

1. Constraints related to:
   - finger's intradigital correlation,
   - finger's transdigital correlation,

Future Work

Future work will see the introduction of physiological constraints subject to the hand's anatomy:

1. Constraints related to:
   - finger's intradigital correlation,
   - finger's transdigital correlation,
2. Skin related constraints,

Future Work

Future work will see the introduction of physiological constraints subject to the hand's anatomy:

1. Constraints related to:
   - finger's intradigital correlation,
   - finger's transdigital correlation,
2. Skin related constraints,
3. Self collisions.

Future Work

Future work will see the introduction of physiological constraints subject to the hand's anatomy:

1. Constraints related to:
   - finger's intradigital correlation,
   - finger's transdigital correlation,
2. Skin related constraints,
3. Self collisions.

Questions

Thank you! Any questions?

References

[1] L. Dorst, D. Fontijne and S. Mann, *Geometric Algebra for Computer Science: An Object-Oriented Approach to Geometry*, Morgan Kaufmann Publ. Inc., 2009.

[2] E. Bayro-Corrochano and D. Kähler, Motor algebra approach for computing the kinematics of robot manipulators, *J. of Robotics Systems*, 17(9): 495-516, 2000.

[3] E. Bayro-Corrochano, Robot perception and action using conformal geometric algebra. In *Handbook of Geometric Computing, Chapter 14*, Springer-Verlag, Heidelberg, pages 405-457, 2005.

[4] D. Hildenbrand, Geometric computing in computer graphics using CGA, *Computers and Graphics*, 29(5): 795-803, 2005.

[5] J. Zamora and E. Bayro-Corrochano, Kinematics and Grasping using CGA, In *Advances in Robot Kinematics*, Springer, pages 473-480, 2006.

[6] D. Hildenbrand, J. Zamora and E. Bayro-Corrochano, Inverse Kinematics computation in computer graphics and robotics using CGA, In *Advances in Applied Clifford Algebras*, volume 18, pages 699-713, 2008.

[7] T. K. Tanev, Geometric Algebra Approach to Singularity of Parallel Manipulators with Limited Mobility, In *Advances in Robot Kinematics*, Springer, pages 39-48, 2008.

[8] D. Hildenbrand, H. Lange, F. Stock and A. Koch, Efficient Inverse Kinematics algorithm based on CGA (Using reconfigurable hardware), 2007.