University of Cyprus

Technical Report

MARKER PREDICTION AND SKELETAL RECONSTRUCTION IN MOTION CAPTURE TECHNOLOGY

Andreas Aristidou

UCY-CS-TR-13-2

August 2013

Table of Contents

Abstract

Optical motion capture systems suffer from marker occlusions resulting in loss of useful information. This technical report addresses the problem of real-time joint localisation of human skeletons in the presence of such missing data; at least three markers are placed at strategic positions on each limb segment. The data is assumed to be labelled 3d marker positions from an 8-cameras PhaseSpace Impulse X2 motion capture system. An integrated framework is implemented using MATLAB which predicts the occluded marker positions using a Variable Turn Model within an Unscented Kalman filter. Inferred information from neighbouring markers is used as observation states; these constraints are efficient, simple, and real-time implementable. This work also takes advantage of the common case that missing markers are still visible to a single camera, by combining predictions with under-determined positions, resulting in more accurate predictions. An Inverse Kinematics (IK) technique is then applied ensuring that the bone lengths remain constant over time; the human skeleton has been structured hierarchically and the IK solver has been applied sequentially to the kinematic chains, maintaining a continuous data-flow. The marker and Centre of Rotation (CoR) positions can be calculated with high accuracy even in cases where markers are occluded for a long period of time. Results demonstrate the efficiency of both the proposed methodology and the implemented algorithms in MATLAB.

1. Introduction

Optical motion capture is a technology used to turn the observations of a moving subject (taken from a number of cameras) into 3D position and orientation information about that subject. It is commonly used to better analyse techniques for sports training and performance; for observation of asymmetries and abnormalities in rehabilitation medicine; in biomechanics labs (prosthetics, ergonomics); and for animating virtual characters for films and computer games.

Our lab is equipped with the new PhaseSpace Impulse X2 motion capture system with active LEDs and a 3-wall immersive virtual reality set-up (see **Figure *1.1***). The PhaseSpace Impulse X2 system uses eight cameras that are able to capture 3D motion using modulated LEDs. These cameras contain a pair of linear scanner arrays operating at high frequency each of which can capture the position of any number of bright spots of light as generated by the LEDs. The system offers a fast rate of capture (up to 960Hz) and allows the individual markers to be identified by combining the information from several frames and hence identifying the marker from its unique modulation.

In this work it is assumed that 3 markers must be available on each limb segment at all times, in order to achieve accurate skeletal reconstruction. However, even with many cameras, there are instances where occlusion of markers by elements of the scene leads to missing data. In order to unambiguously establish each marker's position, it must be visible by at least two cameras (using PhaseSpace X2 system, markers must be visible to one and a half of the double cameras) in each frame.

In this technical report we have implemented the methodology described in [AL13] for real-time marker prediction and joint localisation; the algorithms have been adjusted and adapted in the PhaseSpace Impulse X2 system. It is important to note that the proposed approach does not require a T-pose or any training sessions prior the capturing session. In this work, a new version of the FABRIK Inverse Kinematics solver has been incorporated for bone length correction and human skeletal control, and it is presented in section 4.2. The human skeleton has been structured hierarchically from the outset and FABRIK has been applied sequentially to manipulate the character posture. Therefore, from this time forth the PhaseSpace Inc. motion capture products will be able to reconstruct the human skeleton in real-time, without facing marker occlusion problems or unrealistic shapes.

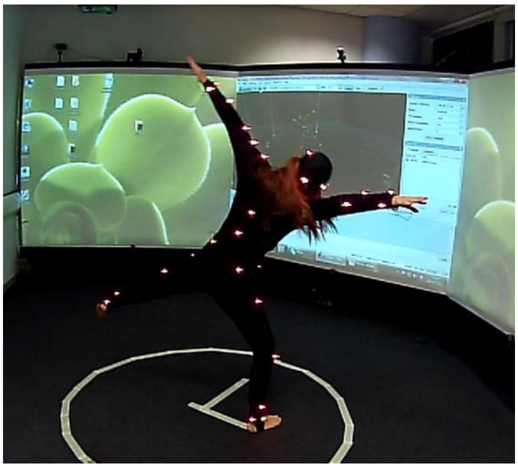


Figure 1.1: *Snapshot of a dance performance using PhaseSpace Impulse X2 motion capture system in our laboratory*

1.1. Mathematical Background

Please note that, in this technical report and for the MATLAB implementation, the mathematical background used is Geometric Algebra [HS84]; you can find more information regarding the Geometric Algebra solutions discussed in this report in [AL10]. Nevertheless, the MATLAB implementation provides all the necessary algorithms for transforming Geometric Algebra to Euclidean and vice versa.

1.2. Data Description

This report is accompanied with the following supplementary materials:

C3D file: stored 3D coordinate information, analog data and associated information as it is recorded from the motion capture system. You can find enclosed the *Panayiotis_All_markers.C3D* file of the experimental data.

RPD file: The PhaseSpace .RPD format that is stored 3D coordinate information, analog data and associated information (including cameras, rigid bodies etc) as it is recorded from the PhaseSpace motion capture system. You can find enclosed the *Panayiotis_All_markers.RPD* file of the experimental data.

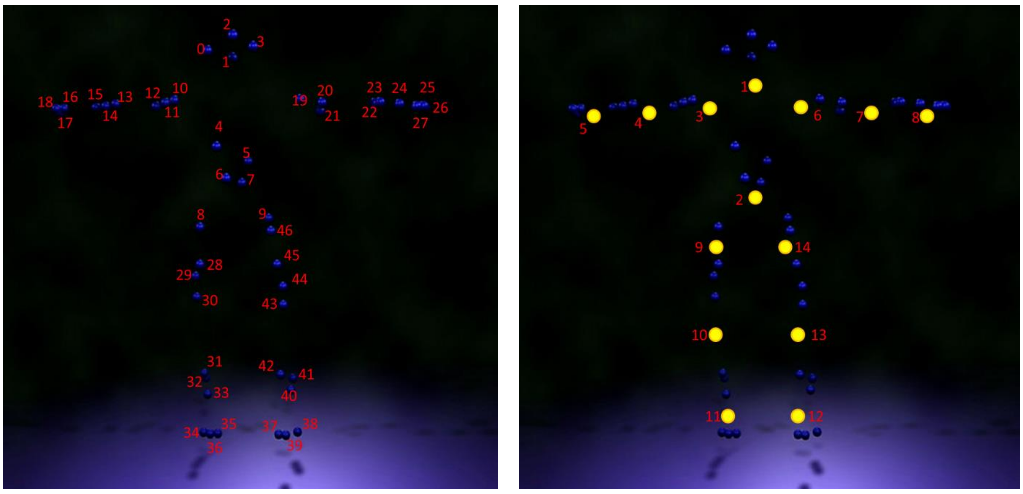
Text files: There are four folders with text data:

A. The **Marker** folder includes the marker positions Y1-Y47, where Y1.txt represents the positions of marker 0, Y2.txt represents the positions of marker 1, Y3.txt represents the positions of marker 2 etc. The marker configuration is presented in **Figure *1.2*** (a).

B. The **Joints, VTM-UKF** folder includes the joint positions J1-J14, where missing markers have been predicted using a VTM-UKF model and marker constraints; the CoRs have been calculated using the marker predicted positions. The joint configuration is presented in **Figure *1.2*** (b).
C. The **Joints, FABRIK** folder includes the joint positions J1-J14, that have been calculated similarly to case B, plus that the CoR positions have been corrected using the FABRIK algorithm and the data have been smoothed using a moving average window.
D. The **Joints, FABRIK start 1170** folder includes the joint positions J1-J14, that have been calculated similarly to C, but the algorithm has been applied at the 1170 frame (T-pose has not being used).

MATLAB code: You can find the MATLAB files that implement the proposed algorithm.

Videos: You can find enclosed two videos with the results of the proposed methodologies.



(a) (b)

Figure 1.2: *The (a) Marker configuration and (b) joint configuration of the skeletal model used in our implementation*

2. Marker prediction

In this MATLAB implementation, a real-time integrated framework has been developed for missing marker prediction, as described in [AL13]; a Variable Turn Model within an Unscented Kalman Filter has been used to predict the occluded marker positions over time. In addition, assuming that the inter-marker distance is constant over time, inferred information from the approximate rigid body assumption is used for constraining the markers positions within a feasible set. This method is automatic and scalable, without requiring any parameters to be set by the user. It considers all the cases of marker occlusion within a limb resulting in accurate predictions even in cases where all markers on a limb segment are missing for an extended period of time. This is the first method that takes advantage of the special, but common, case where missing markers are visible to just one camera, reducing the marker estimation error significantly. If a marker is partially visible to only one camera results in a line starting from the camera and passing through the marker position; the proposed framework uses this information for better approximation of the missing marker position. For the purpose of this work, [AL13] has been adjusted to work with PhaseSpace Inc. Impulse motion capture system; moreover, we have modified the marker configuration of the PhaseSpace mocap suite in order to ensure 3 markers will be attached at each limb segment, as per [AL13] assumption.

3. Centre of Rotation Estimation

With a continuous stream of accurate labelled 3D data, we can perform real-time CoR estimation. Please note that, during capture, markers must be carefully placed on the body in order to obtain good results. Results using markers placed too close to the CoR are more susceptible to errors since a small error may cause large deviations in the estimated rotation, leading to erroneous calculation of the model parameters. In addition, markers must be placed as far as possible from each other for better estimation of the bone rotation.

The data discussed here are labelled by PhaseSpace Implulse X2 active motion capture system where no tracking is necessary. The PhaseSpace Impulse system identifies and tracks individual markers from their unique modulation and in this report problems related to marker inversion are not therefore considered. In general, 3 markers per bone segment are required to estimate the CoR for joints with 3 Degrees of Freedom (DoF); for simpler problems having fewer DoF, such as knees and elbows, the CoR can be calculated with fewer markers [CP07]. In this report, we consider the general case of joints with 3 DoF, since no prior knowledge of the model or joint-type is assumed.

Locating the CoRs is a crucial step in acquiring a skeleton from raw motion capture data. To calculate the joints between two sets of markers, it is helpful to have the rotation of a limb at any given time. The orientation of a limb at time k relative to a reference frame can be estimated using the Procrustes formulation [Horn87]. The location of the joints can be calculated using the approach described in [CL05], which has been amended in [AL13] to respond in presence of missing marker data; both of these methods take advantage of the approximation that all markers on a segment are attached to a rigid body.

4. Centre of Rotation Correction

The CoR positions are thereafter corrected via a real-time Inverse Kinematics (IK) technique which guarantees that the inter-joint pairwise distances remain constant over time. The human skeleton has been structured hierarchically in different kinematic chains and then the FABRIK Inverse Kinematic solver [AL11a] has been applied sequentially in order to maintain fixed bone lengths over time.

4.1. Forward And Backward Reaching Inverse Kinematics

Forward And Backward Reaching Inverse Kinematics (FABRIK) is a recent, real-time IK solver which returns smooth postures in an iterative fashion. Instead of using angle rotations, FABRIK treats finding the joint locations as a problem of finding a point on a line; the algorithm trace back step by step to different positions of the joint of a chain, crossing the chain and back into a finite number of iterations. FABRIK supports all the rotational joint limits and joint orientations in an iterative fashion by repositioning and re-orienting the target at each step. It does not suffer from singularity problems, it is fast and computationally efficient. The main interests of the FABRIK approach lies in its simplicity, its low computational cost and its ability to control multiple end effectors, making it ideal for applications on timeliness systems.

Although FABRIK is a recent algorithm, it has become one of the most popular IK solvers; many researchers and game developers have implemented or extended FABRIK due to its efficiency and simplicity. For instance, FABRIK has been used for hand skeleton reconstruction [AL11b], [Alv11], and for skeletal control under marker occlusion in motion capture technology [AL13]; Liu implemented FABRIK for robot manipulation [Liu12], Munshi for robot simulation [Mun12], while Lo and Xie [LX13] used FABRIK in a redundant 4-revolute (4R) spherical wrist mechanism for an active shoulder exoskeleton. Besides, different variations of FABRIK are currently available; Ramachandran and Nigel, [RN13], solve the IK problem using an alternate of FABRIK with intersection of circles, while Naour et al. [NCG12] uses FABRIK within a global iterative optimisation process. Furthermore, Huang and Pelachaud, [HP12], use a variation of FABRIK in order to solve the Inverse Kinematics problem from an energy transfer perspective. They used a mass-spring model to adjust the joint positions by minimising the force energy which is conserved in springs. Recently, Moya and Colloud, in [MC13], proved that FABRIK can manage with target priorities, adjusting the initial algorithm to deal with joints that have more than two segments. The flexibility of the algorithm to be easily adapted into different problems, its easy configuration, its low computational cost, and the algorithm's performance in closed-loops or problems with multiple end effectors, make FABRIK a popular and efficient IK solver.

4.2. FABRIK implementation for skeletal control

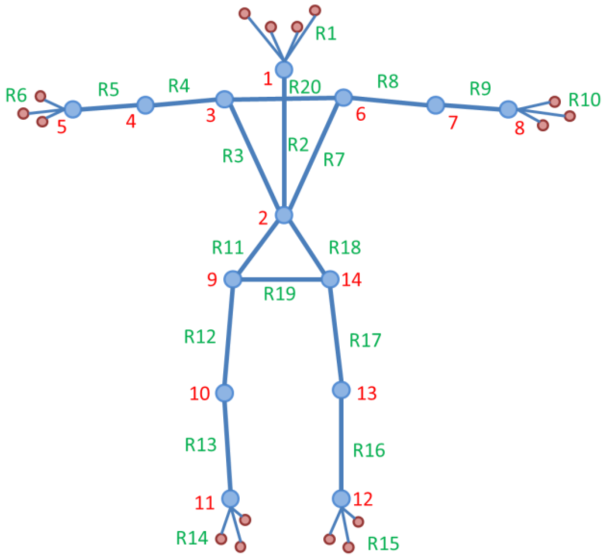


Figure 4.1: *The skeletal configuration used in our implementation, showing the naming of the CoRs and the inter-joint dista-nce vectors (R). Joints are coloured in blue while end-effectors in red; end-effectors are assumed to be the marker positions.*

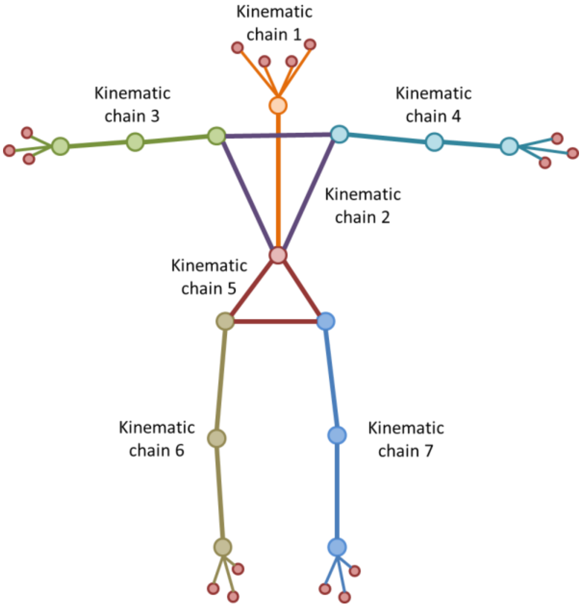


Figure 4.2: *The skeletal hierarchy and the seven different kinematic chains of our implementation.*

In this section we analyse the human model structure of our implementation using FABRIK, in order to control the anatomy of the tracking character. It is important to note that in this implementation the inter-joint distances are assumed to be constant (or close to constant) over time. Hence, the first step is to calculate the distances between the indicated joints (Rs); for stability reasons, there is a need to have at least 15 frames with the true joint positions in order to be able to accurately calculate the bone lengths. The configuration and naming of these distances in MATLAB can be seen in **Figure *4.1***. The bone lengths are computed as the average distance between joints over time from the first till the current frame (or if preferred, the average distance of the last 200 frames). In that manner it is ensured that any noise in the data will not affect the anatomy of the character. In addition, since the CoRs are calculated dynamically and there are cases with body flexion and extension, this approach allows a slight violation of the constant inter-joint constraints that is depended on the estimated joint positions (the joint positions that have been calculated using predicted marker positions). Moreover, in order to construct a hierarchical solution, we have divided the problem into smaller steps, e.g. kinematic chains, for easier manipulation. **Figure *4.2*** shows the human skeleton and the seven different kinematic chains that have been used for skeletal control.

In the proposed model, the skeleton is constructed by defining joint 2 as the root joint of the system. Looking at the marker configuration, as shown in **Figure *1.2*** (a), joint 2 is calculated using markers 4, 5, 6, 7 and 8, 9, 46. In addition, joint 2 is connected to five joints, creating the main torso of the human skeleton, as shown in **Figure *4.3***.

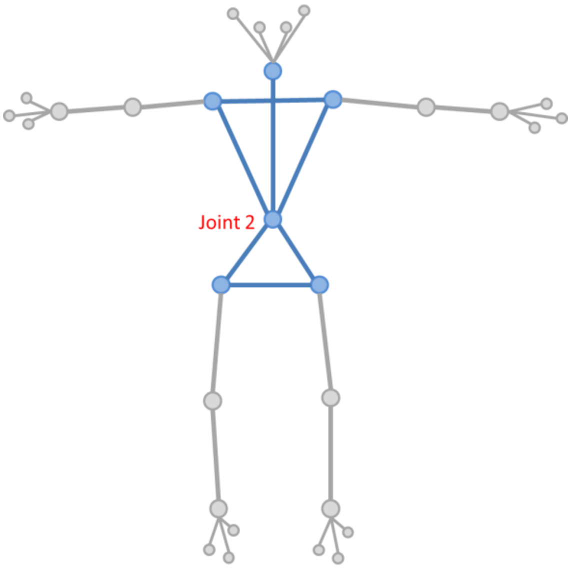
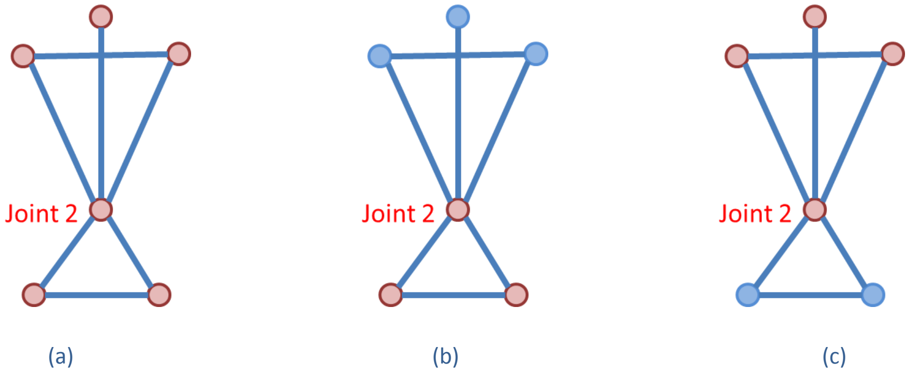


Figure 4.3: *The root joint and the torso of the human skeleton.*

4.2.1. Correcting the root joint

The first step of the proposed methodology requires a guarantee that the position of joint 2, which is defined as the root joint, corresponds as closely as possible to its true position. Obviously, if the position of joint 2 has been determined using true marker positions, then the step of locating the root joint is considered to be solved. Nevertheless, there are three cases where the root joint is calculated using predicted markers positions, as shown in **Figure *4.4***. The first case is when none joint of the body torso is calculated using true marker positions, as shown in **Figure *4.4*** (a); in the second and third case additional bone length constraints can be incorporated since there are other joints of the torso that have been calculated using true marker positions, as shown in **Figure *4.4*** (b) and (c).



(a) (b) (c)

Figure 4.4: *The three cases where joint 2 has been estimated using predicted marker positions. (a) case where no other joint of the torso has been estimated using true marker position, (b) and (c) cases where other joints of the torso have been calculated using true marker positions.*

For the first case (**Figure *4.4*** (a)), since no additional information can be used for joint correction, it is considered that the estimated position of joint 2 is the true joint position. In the second and third cases (**Figure *4.4*** (b) and (c)), since there is additional information from the other visible joints[1] of the torso, the root joint can be further constrained to meet the inter-joint distance restrictions. The proposed solution for both of these cases has similar procedure, which is a simple variant of the FABRIK Inverse Kinematics algorithm. The procedure involves an iterative process until the current and the previous joint positions (of the corresponding joint) do not differ or their difference is less than a minimum accepted error. Starting from one of the visible joints, FABRIK is applied assuming that there are only two joints, where one is a visible joint and the other is the end-effector (non-visible joint[2]), as shown in **Figure *4.5***. In that manner, joint 2 (the non-visible joint) is re-positioned in an intermediate position (step 1); the intermediate position meets the inter-joint distance restrictions from the first visible joint, but not from the other. Thus, the same procedure is applied but now from the other visible joint position (step 2), where end-effector is assumed to be the intermediate position of joint 2.

---

[1] Visible joint is assumed to be the joint that has been calculated using true marker positions.

[2] Non-visible joints are assumed to be the joints that have been calculated using predicted marker positions.

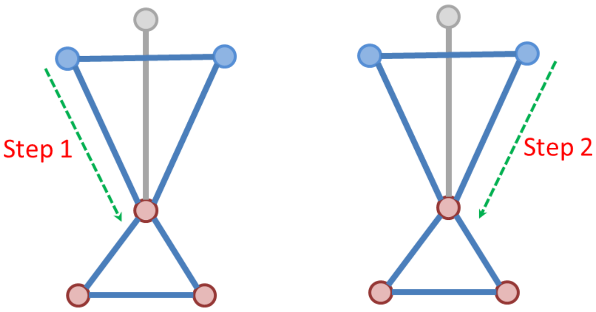


Figure 4.5: *The FABRIK variation of a triangle Inverse Kinematics problem when two joints are visible and one is non-visible.*

The procedure is repeated until the difference in the position of joint 2 in the current and previous iteration is less than an acceptable error. MATLAB functions `triangle1FABRIK.m` and/or `triangle1FABRIK2.m` solve the Inverse Kinematics problem for the aforementioned case.

4.2.2. Controlling the torso

Having the true or the corrected position of the root joint, means that the position of the remaining skeleton joints can be calculated; the rest of the skeleton has been structured hierarchically, starting from the root joint and moving to the end-effectors (as shown in **Figure *4.1***). In the proposed skeletal model there are five end-effectors, the two hands, the two feet and the head.

The skeletal model has been divided into three main groups, as shown in **Figure *4.6***: the upper body, the lower body and the head. As can be seen, the upper and the lower body have exactly the same structure, consisting of three kinematic chains, one root and two end-effectors. Thus, solving the Inverse Kinematics problem for the upper body means that the lower body problem is assumed to be solved as well. In addition, the third group, meaning the head, consists of just one kinematic chain and it can be easily structured using the simple version of the FABRIK algorithm with multiple end-effectors, as it is described in [AL11].

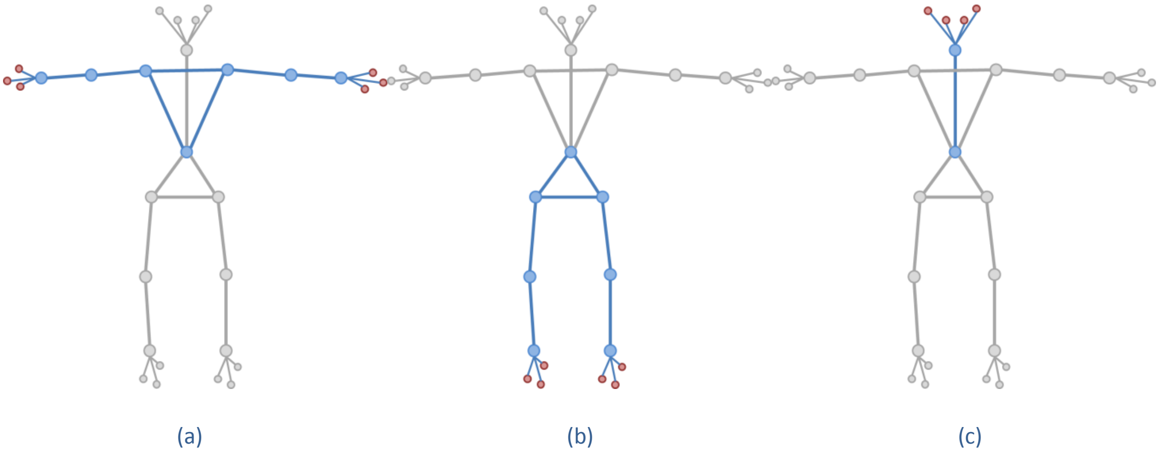


Figure 4.6: *The three main human skeleton groups: (a) upper body, (b) lower body, and (c) head.*

In this paragraph, we describe the IK solution for the upper body using FABRIK. It is important to remind that this group consists of three kinematic chains and two end-effectors. Again, joint 2 is assumed to be the root joint of the human model structure.

The first step of this procedure is to control the joint positions that belong to the torso, meaning joints 3 and 6 (as shown in **Figure *4.1***). Obviously, if the joints have been calculated using true marker positions, there is no need for joint correction. However, if there are non-visible joints, it is a necessity to ensure that the bone length restrictions are satisfied. There are four cases of non-visible joints in the upper body torso, as it is shown in **Figure *4.7***. In order to secure that the inter-joint distance restrictions are met over time, we can take advantage of the triangle that has been created between joints 2, 3 and 6.

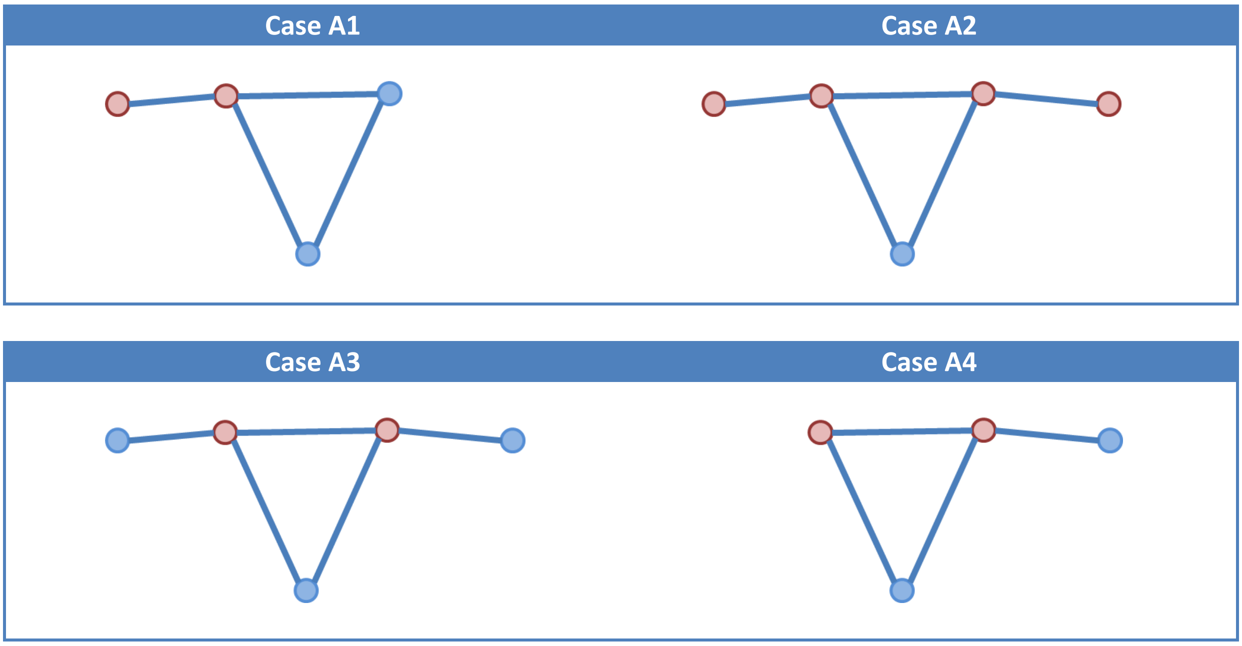


Figure 4.7: *The four cases of non-visible joints in the upper body torso*

Case A1 is the simplest case for re-position of the torso joints; the procedure used is similar to the root joint correction, where one non-visible and two visible joints exist (see **Figure *4.8***). The MATLAB function used for that case is again the `triangle1FABRIK2.m`.

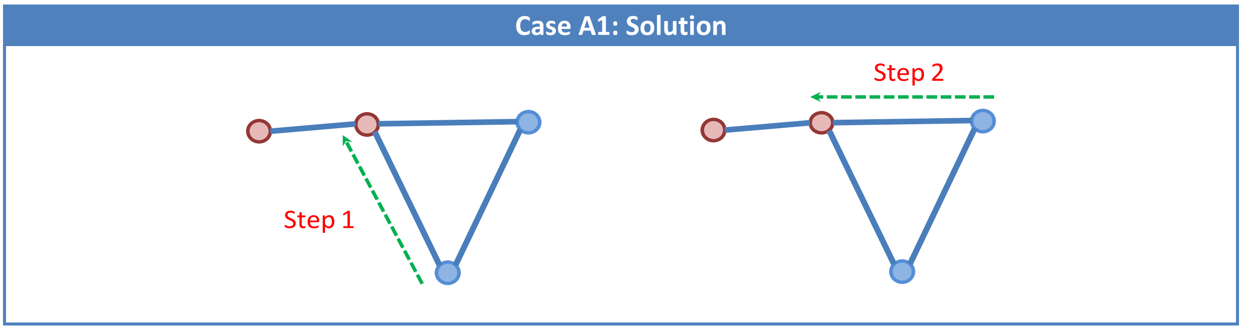


Figure 4.8: *The solution for the simplest torso joint correction, where one non-visible and two visible joints exist.*

On the other hand, in Case A2 there are two non-visible and only one visible joint (joint 2), as it is shown in **Figure *4.7***. The joints in that case cannot be hard restricted since there is not sufficient information for further restriction. The proposed solution is divided into two phases, as it is presented in **Figure *4.9***. In this case the FABRIK algorithm is applied in a circular form, attempting to correct the non-visible joints to meet the distance limitations. Thus, starting from the visible joint, the algorithm re-positions the non-visible joints (step 1 and step 2), giving them an intermediate value. Subsequently, the algorithm gives a new temporal value for the visible joint (in step 3). The first phase is completed in step 4, as shown in **Figure *4.9***. In the second phase of the algorithm, the visible joint returns to its original position (which in fact is the true position), and the algorithm is applied from the other direction, as shown in step 5 and step 6. This procedure is repeated until the positions of the non-visible joints in the current and previous iteration are identical or their difference is smaller than an acceptable error. The MATLAB function for this solution is `triangle2FABRIK.m`.

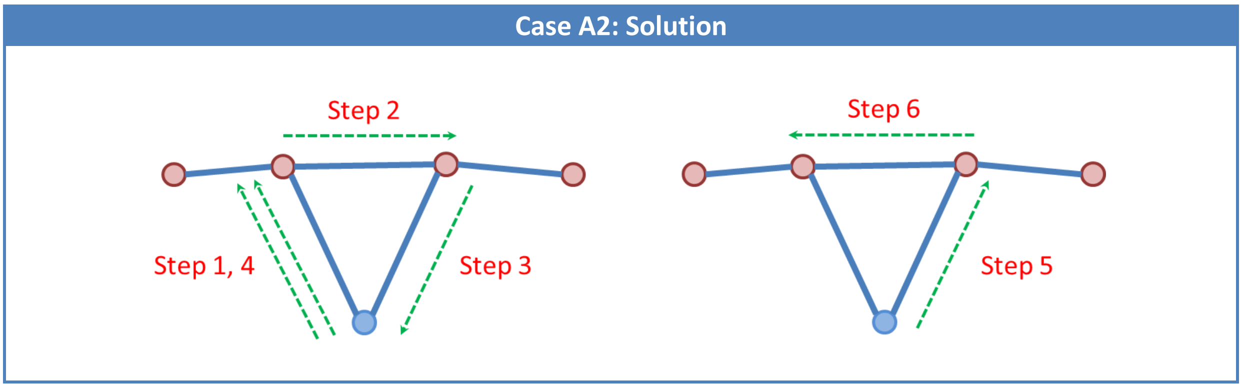


Figure 4.9: *The solution of Case A2, where torso has two non-visible joints and one visible joint and no additional information is available.*

In cases A3 and A4 there are two non-visible joints in the torso; in addition, visible joints from the rest of the body can be used for inter-joint distance restrictions and skeletal control of the torso joints. Both cases have a similar solution, as presented in **Figure *4.10***; in A3 there are two joints from the rest of the body that contributes in the skeletal control, whereas in case A4 there is only one joint. The solution of the IK problem in cases A3 and A4 is a combination of the solutions in A1 and A2; thus, the procedure for A3 is as follows: the first six steps are the same as the solution given in case A2, the steps 7, 8 and steps 9, 10 are similar to case A1, but implemented twice, and finally, the last six steps are again similar to case A2. A3 solution is implemented in MATLAB as `MultipleFABRIK.m`, while A4 as `MultipleFABRIK2.m`.

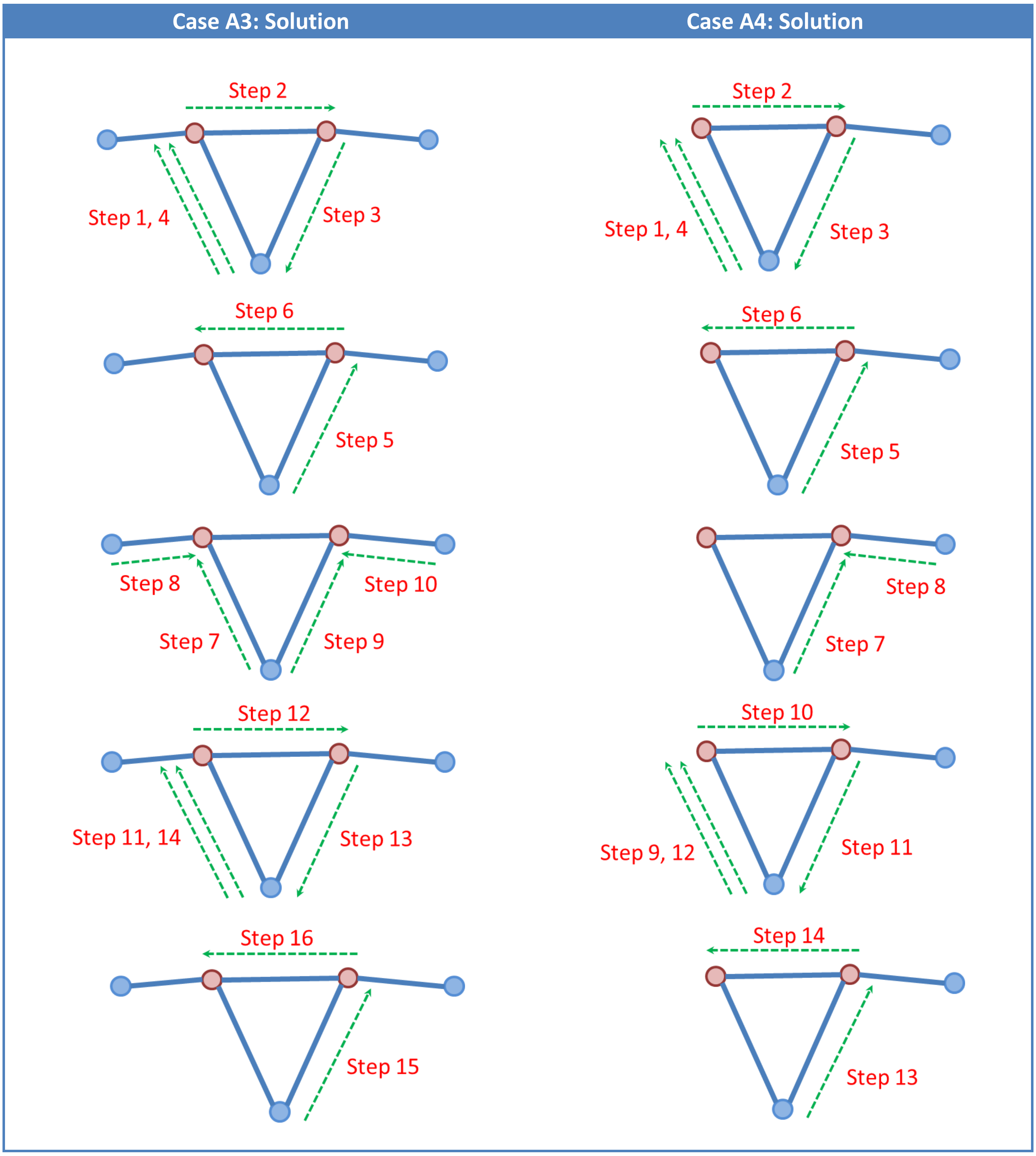


Figure 4.10: *The Inverse Kinematics solutions for cases A3 and A4, where two non-visible joints exist in the torso triangle; additional information from the rest of the body can be used for correcting the non-visible joints.*

Thereafter, since it is secured that the joints of the torso have the appropriate inter-joint distance, we can calculate the joint positions of the rest of the body. In the proposed model, there are five different kinematic chains: kinematic chain 1 for the head, 3 and 4 for the upper body, and finally 6 and 7 for the lower body. The proposed IK solution is similar for all the aforementioned kinematic chains; thus, in this technical report we present the solution only for the upper body.

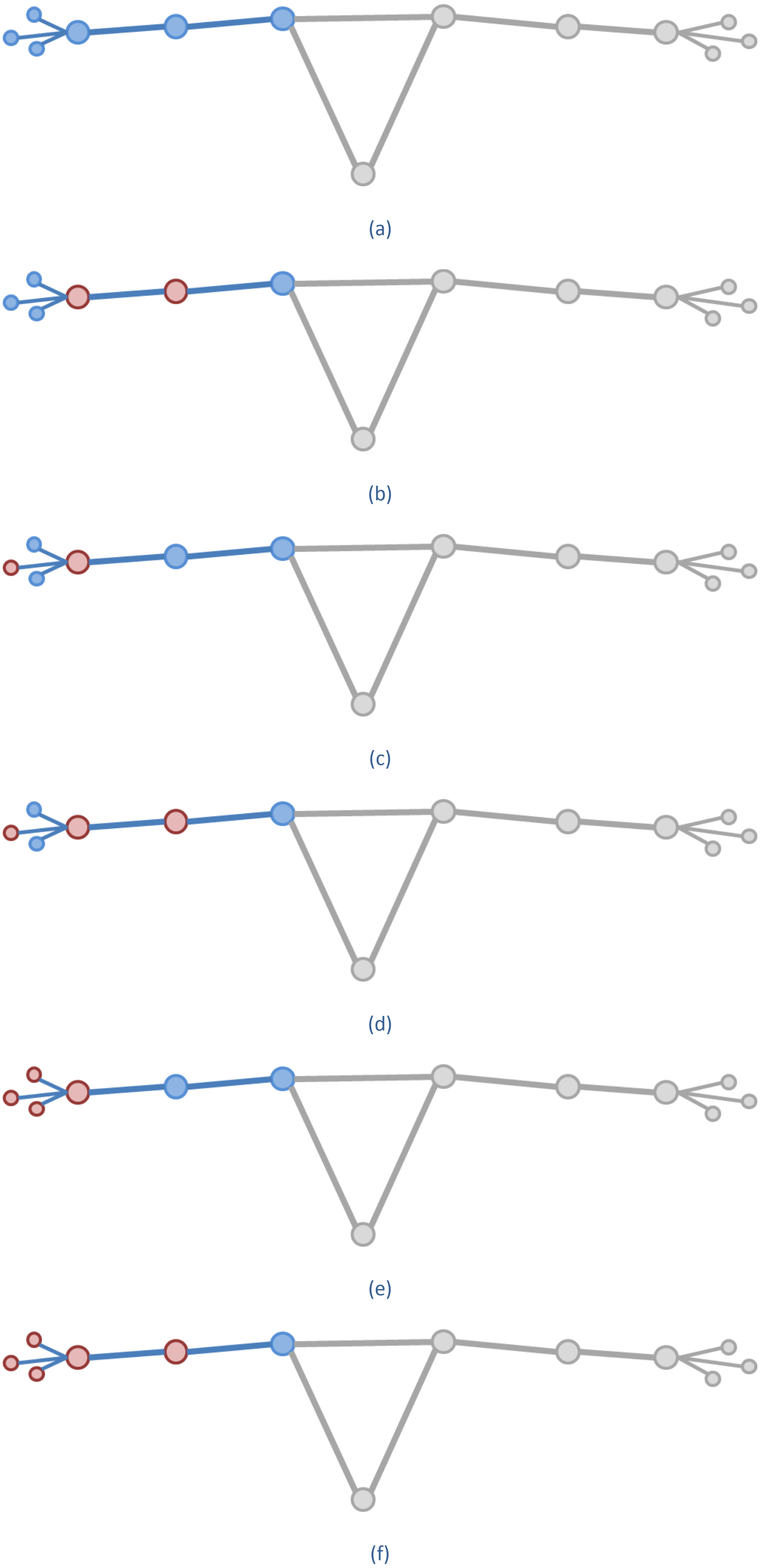


Figure 4.11: *Six different cases of occlusion in the end-limbs of the upper body. Visible joints are coloured in blue, while non-visible in red.*

The kinematic chains are distinguished in six different joint occlusion cases, as shown in **Figure *4.11***. However, the solution can be simplified in only two main cases; the case where there are end-effectors visible and the case where all end-effectors are non-visible. The procedure used in these cases is described in [AL13]; when visible end-effectors exist, an iterative procedure of the FABRIK algorithm is applied. On the other hand, when there is no visible end-effector, the algorithm is not iterative but only the forward reaching approach of the FABRIK algorithm in applied. The solutions are presented in **Figure *4.12***, where case B1 is the solution when visible end-effectors exist and case B2 when there are no visible end-effectors. The MATLAB function for these cases is `FABRIK1Marker.m`.

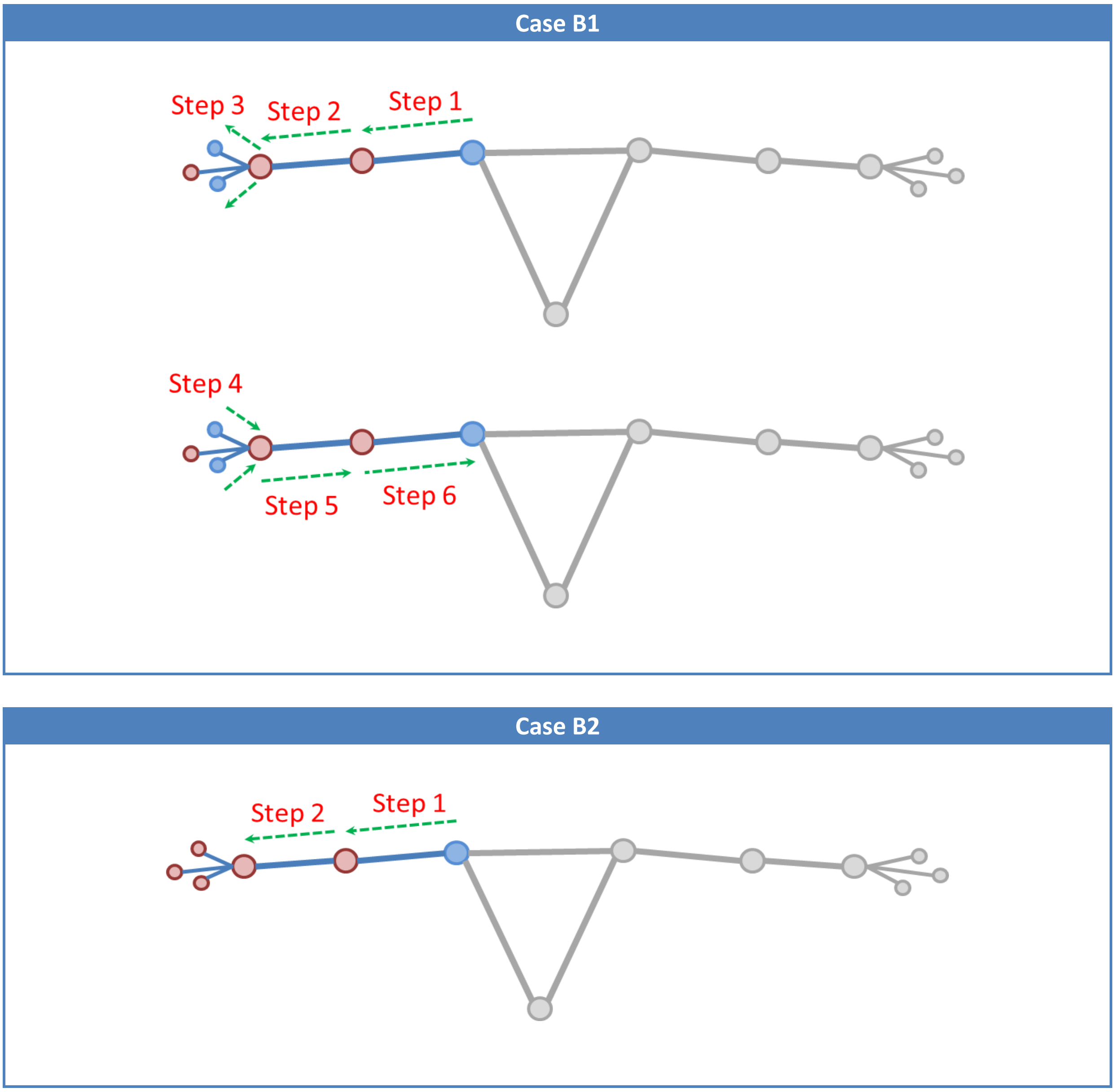


Figure 4.12 *The Inverse Kinematics solution for the two main occlusion cases of the end-limbs of the body.*

4.3. Smoothing the CoRs for noise reduction

Since data may become noisy due to continual and extended marker occlusions, we have applied a real-time window average filter to overcome possible oscillations in CoR positions. The implemented average filter (MATLAB function: `CoRAveraging.m`) uses only previous values (real-time applicable); however, the implemented filter needs further investigation since it causes delays and the CoR positions appear displaced by 20-50 frames.

5. Results

The results of the proposed methodology are clearly demonstrated on the enclosed videos, proving that the proposed methodology can predict the missing markers and efficiently estimate the CoR positions, driving to real-time human skeletal reconstruction. **Figure *5.1*** shows a snapshot of the results, where on the left side is the skeletal reconstruction without applying FABRIK and on the right side the same solution but after incorporating FABRIK for fix inter-joint distance.

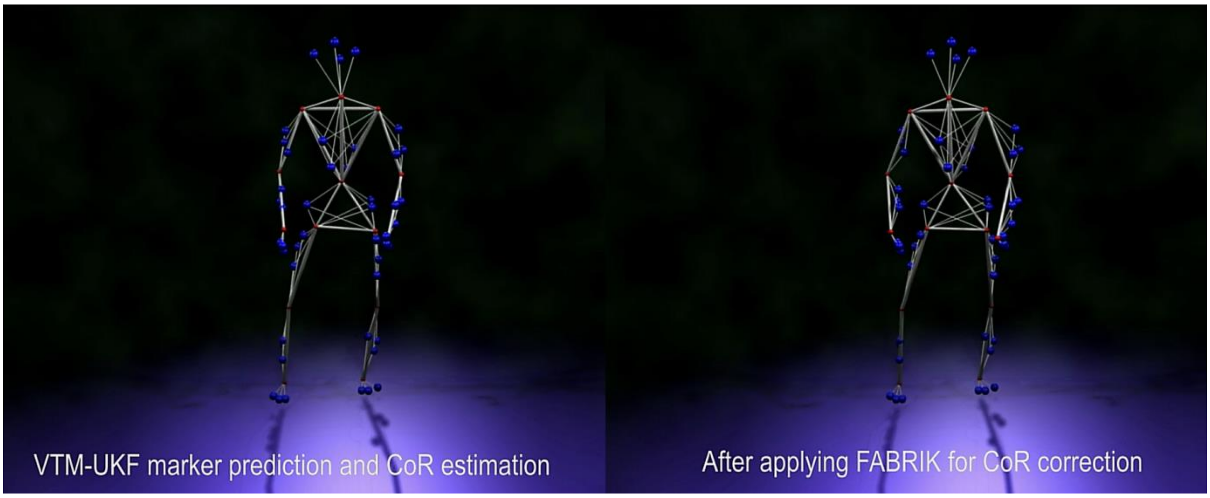
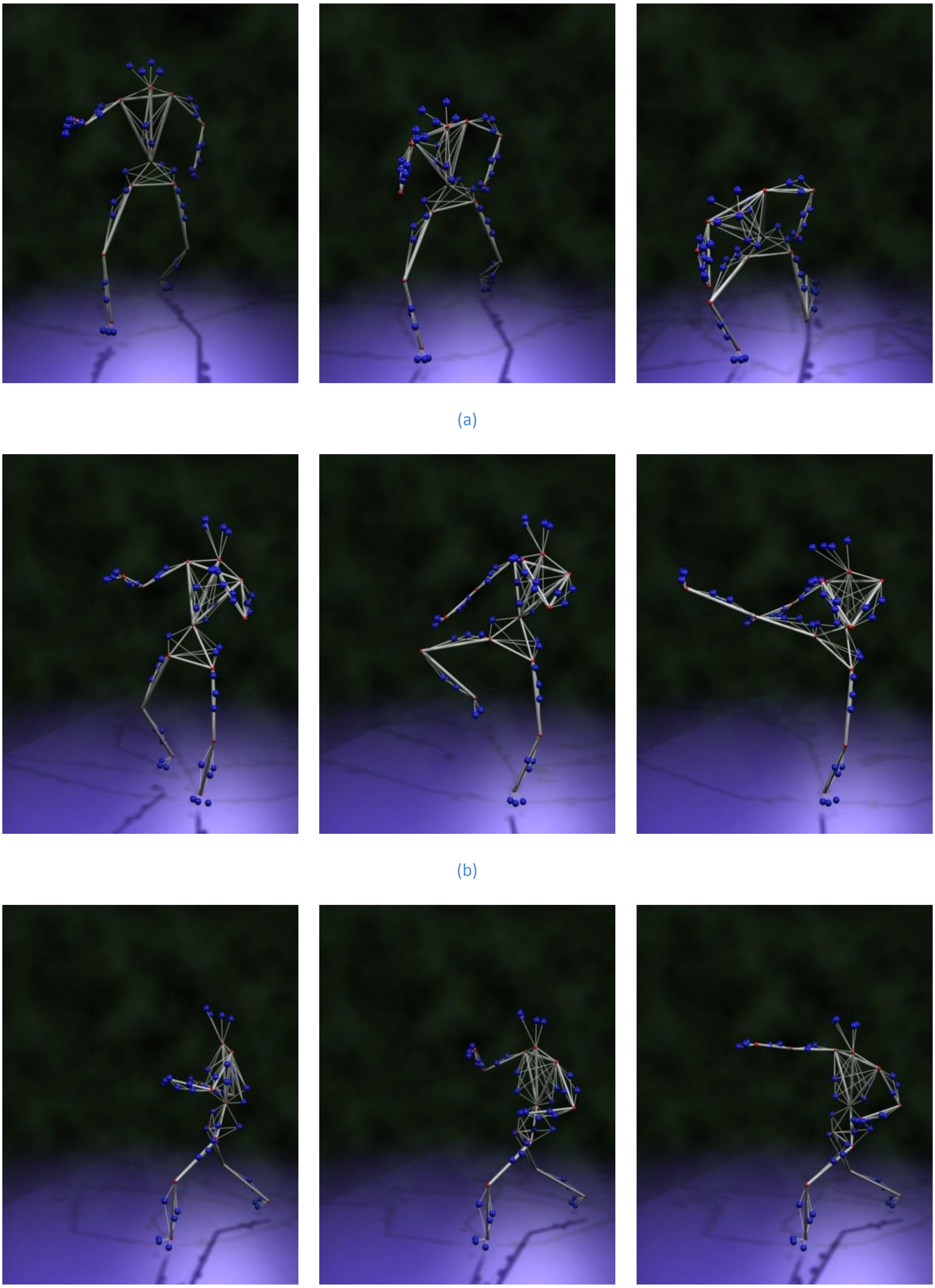


Figure 5.1: *The skeletal reconstruction using the proposed methodology; on the left side, when CoRs have been calculated using the VTM-UKF model, and on the right side when CoRs positions have been corrected using FABRIK.*

Results demonstrate the efficiency of the proposed methodology; even if only limited information about the tracking pose was available, the proposed model returned visually natural solutions that satisfy the user or character constraints. **Figure *5.2*** (a), (b) and (c) show an example where the skeletal reconstruction methodology and the proposed skeletal hierarchy have been incorporated within PhaseSpace Impulse X2 motion capture framework. It shows different examples where the tracking character kneels, gives a kick and a punch respectively. The reconstructed postures are animated in smooth motion without discontinuities, abnormalities or oscillations.



(a)

(b)

(c)

Figure 5.2: *Snapshots of FABRIK implementation within a PhaseSpace motion capture framework; the algorithm was successfully incorporated, ensuring that the inter-joint distance will remain constant over time.*

5.1. Important notes, limitations and further improvements

A. Marker placement: Markers must be placed as far as possible from the Centre of Rotation (CoR) positions and as far as possible from each other on the limb segment. In our experimental data, the marker placement is not the best possible; actually it can be further improved for better results. As can be seen, the upper body works better than the lower body since in the latter the markers have been placed very close to the hip (see **Figure *5.3*** (a)). **Figure *5.3*** (b) shows an example of a lower body implementation where the markers have been placed in appropriate positions, resulting in better estimations of CoR and less noise.

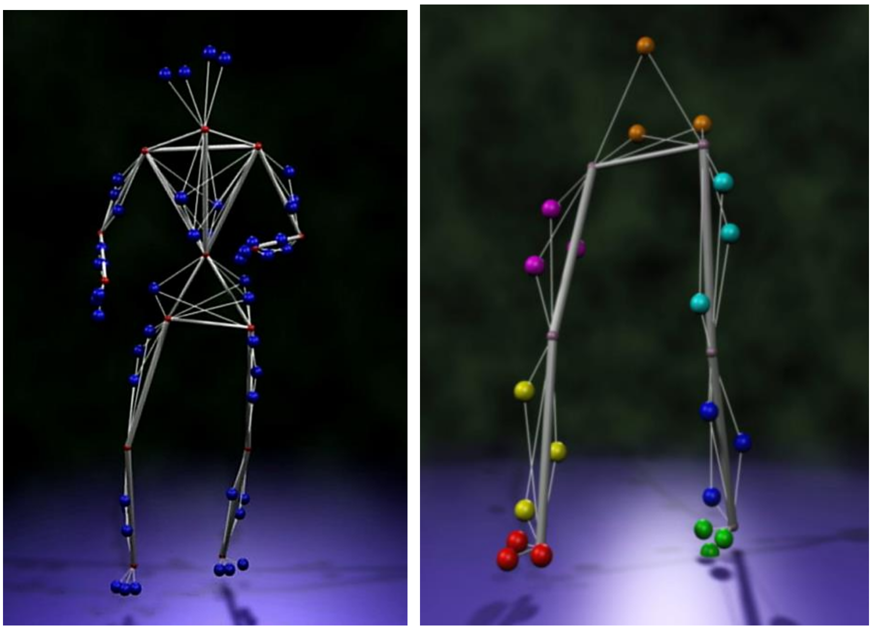


Figure 5.3: *(a) The full body skeletal reconstruction of our experimental data, and (b) the lower body reconstruction data of [AL13].*

B. No need for T-pose or training sessions: The algorithm for CoR calculation is dynamic and changes as the user moves his body segments (kinematic chains) over time. Thus, it is recommended that users will move their body parts for at least 240-480 frames (0.5 – 1 second) for initialisation purposes. In the video example, the algorithm fails in the first 2 seconds because there is no movement from the actor (T-pose). Nevertheless, the algorithm was able to recover fast and have reasonable results just after two seconds. It is important to note that, as the time passes and more data come into the system, the joint positions are getting more stable.

C. All markers must be visible for initialisation purposes: In this implementation, it is a requirement that all markers will be available for initialisation of the system. For instance, in order to apply the CoR estimation and the marker prediction methods, all markers must be visible during the first 3 frames. Furthermore, in order to utilise the FABRIK inter-joint correction algorithm, all markers must be available for the first 15 frames. Thus, before applying the proposed approach, it is suggested to check whether all markers are visible; if they remain available for 15-20 frames, then continue to the skeleton reconstruction, otherwise restart the checking process.

D. Partially visible markers: One of the main novelties and advantages of the proposed methodology for marker prediction is the usage of additional information from partially visible markers. In order to unambiguously establish its position, each marker must be visible to at least two cameras in each frame. However, it is a common phenomenon that markers are visible to a single camera; using these under-determined positions, we can further improve the marker predictions. By relaxing the constraints that the inter-marker distance is constant and accepting that the real position of the marker is on the line starting from the camera and passing through the marker, we obtain a more accurate estimate of the position of the relevant marker. This methodology has been implemented in the MATLAB code, but has not been used in the video example since no such data were available.

E. Reset button: It is recommended to include a “skeleton reset” button in the final system that will allow re-calculating the CoRs and reconstructing the skeleton in case that something fails badly.

F. Marker assignment: It is a necessity to assign the attached markers to rigid bodies (each limb segment is a rigid body with at least 3 markers attached) at the beginning, probably something similar to what PhaseSpace is currently using for Recap or Motion Builder applications. In this example, the marker assignment to rigid bodies is the following:

Rigid body	Makers IDs				Rigid body	Makers IDs		
Head	0	1	2	3	**Right upper leg**	28	29	30
Chest	4	5	6	7	**Right lower leg**	31	32	33
Pelvis	8	9	46		**Right foot**	34	35	36
Right upper arm	10	11	12		**Left upper leg**	37	38	39
Right lower arm	13	14	15		**Left lower leg**	40	41	42
Right hand	16	17	18		**Left foot**	43	44	45
Left upper arm	19	20	21					
Left lower arm	22	23	24					
Left Hand	25	26	27					

Acknowledgements

This project is co-financed by the Office of Naval Research (ONR) Global (N62909-13-1-V090), the European Regional Development Fund and the Republic of Cyprus through the Research Promotion Foundation (DIDAKTOR/0311/73). The author would also like to thank Mr Tracy McSheery and Dr Kan Anant of PhaseSpace Inc. for their valuable help, as well as Mr Panayiotis Charalambous for his help in capturing the experimental data.

References

[AL10] Aristidou, A. “Tracking and Modelling Motion for Biomechanical Analysis”, *PhD thesis*, University of Cambridge, Cambridge, UK, October 2010.

[AL11a] Aristidou, A. and Lasenby, J. “FABRIK: a fast, iterative solver for the Inverse Kinematics problem”, *Graphical Models*, 73(5):243-260, Elsevier 2011.

[AL11b] Aristidou, A. and Lasenby, J. “Inverse kinematics solutions using conformal geometric algebra”, *Guide to Geometric Algebra in Practice*, Springer London, pages 47-62, 2011.

[AL13] Aristidou, A. and Lasenby, J. “Real-Time Marker Prediction and CoR Estimation in Optical Motion Capture”, *The Visual Computer*, Volume 29, Issue 1, pp 7-26, 2013.

[Alv11] Alver, J. “A virtual hand for prosthetic training”, *Master thesis*, Chalmers University of

Technology, Gothenburg, Sweden, 2011.

[CL05] Cameron, J. and Lasenby, J. "A real-time sequential algorithm for human joint localization". *In ACM SIGGRAPH Posters*, p. 107. ACM, New York, 2005.

[CP07] Chang, L.Y., Pollard, N. "Constrained least-squares optimization for robust estimation of center of rotation". *Journal of Biomechanics*, 40(1), 1392–1400, 2007.

[Horn87] Horn, B. "Closed-form solution of absolute orientation using unit quaternions". *Journal of the Optical Society of America*, A 4, 629–642, 1987.

[HP12] Huang, J. and Pelachaud, C., "An efficient energy transfer inverse kinematics solution". *In Proceedings of Motion In Game*, LNCS volume 7660, pages 278-289, Berlin, Heidelberg. Springer, 2012.

[HS84] Hestenes, D. and Sobczyk, G. "Clifford Algebra to Geometric Calculus: A Unified Language for Mathematics and Physics", *Reidel*, Dordrecht, 1984.

[Liu12] Liu, Y.-C. "Gesture and End-Effector Trajectory Planning of Robot Manipulator using Human Motion Imitation", Master thesis, National Taipei University of Technology, China, 2012.

[LX13] Lo, H.S. and Xie, S.S.Q., "Optimization of a redundant 4r robot for a shoulder exoskeleton", *In Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 798–803, Wollongong, Australia, July 2013.

[MC13] Moya, S. and Colloud, F., "A fast geometrically-driven prioritized inverse kinematics solver", *In Proceedings of the XXIV Congress of the International Society of Biomechanics (ISB 2013)*, pages 1–3, August 2013.

[Mun12] Munshi, S. M. "Analysis, Investigation and Design of Flexible Universal Pneumatic Industrial Manipulators Involving Cartesian and Joint Control in the Basis of Economic Feasibility and Appropriate Technology." PhD thesis, University of New South Wales, Australia, 2012.

[NCG12] Le Naour, T., Courty, N. and Gibet, S. "Cinématique guidée par les distances", Revue Électronique Francophone d'Informatique Graphique, Volume 6, Numéro 1, pp. 15–25, 2012

[RN13] Ramachandran, S and Nigel, J. W. "A Fast Inverse Kinematics Solver using Intersection of Circles", In Proceeding of the Eurographics UK Theory and Practice of Computer Graphics conference, 2013.