Intelligent Motion Tracking by Combining Specialized Algorithms

Matthias Weber

FGAN e.V., FKIE, Neuenahrer Str. 20, 53343 Wachtberg-Werthhoven, Germany
m.weber@fgan.de
<http://www.fgan.com/fkie>

Abstract. Motion Capture is a widely accepted approach to capture natural human motion, usually utilizing markers to track certain anthropological points on the participant's body. Unfortunately, these markers do not carry any identification information. Furthermore, marker data can be noisy. To address these problems this work suggests a hybrid approach, i.e. an approach using several experts to solve easier, less complex subproblems. Currently, the presented hybrid approach is built upon three methods, two for identification and one for tracking purposes. For identification of an initial posture, a PCA-based technique for aligning a skeleton model as well as a tree-based optimization comparing anthropometric and tracking data are introduced. To complement the hybrid computation pipeline a neural network algorithm based on self-organizing maps tracks the markers on subsequent frames.

Keywords: Motion Capture, Marker Identification, Neural Networks.

1 Introduction

Analyzing life-like human motion is often a tedious and exerting task. The underlying motion data is usually captured with a Motion Capture (MoCap) system. These systems produce a lot of data by recording many data frames at a certain rate. In addition, most systems track positional data of certain anthropological points on the participant's body. As this positional data is recorded several times per second for all anthropological points, a lot of data is gathered that has to be analyzed.

Furthermore, for capturing human motion, MoCap systems should be used that do not hinder natural movement. Most systems utilize markers to track certain anthropological points on the participant's body. Markers are sensors, emitters or reflectors which provide positional information to the system. Currently, passive optical systems promise the least hindering of natural movement as they do not require cables or big markers. They use reflectors as markers which are very small but do not carry any identification information. Such optical MoCap systems additionally have to cope with hidden markers or suddenly appearing ghost markers that do not exist in reality. Therefore, the gathered MoCap data can be noisy.

As a consequence the MoCap data first has to be processed to circumvent these problems, leading to mostly polished data which in turn makes later analysis easier. Several algorithms have been proposed to cope with unlabeled and noisy data. Usually these approaches still require certain, often manual steps to be taken. Additionally, they are often very specialized, like dealing with identification or hidden markers only. Approaches that use several experts to solve subproblems, seem to be non-existent in current research. Such a separation into subproblems can yield several advantages. It is often much easier and less complex to solve subproblems. The resulting solutions provide a good basis for solving the whole problem. This work presents such a hybrid approach, built upon three methods, two for identification and one for tracking purposes. It is presented by first giving an overview on related work, followed by the methods for initializing the internal model and for tracking the markers on consecutive MoCap frames. Finally, the approach will be evaluated, leading to a conclusion.

2 Related Work

As this work concentrates on marker-based MoCap techniques only related work relevant to marker-based tracking is described in this section. Research in this field can be separated in the two research topics occlusion free marker tracking and skeleton fitting. Dorfmueller-Ulhaas [1] extended the Kalman filter in conjunction with a motion model based on exponential maps to deal with occlusions. Filtering can be seen as a preprocessing step for a later estimation of the human skeleton configuration from the tracked point cloud. While providing higher accuracy, such filtering techniques can increase the computational demands.

For skeleton fitting a least-squares method, is often used to fit a reference skeleton model into a MoCap data cloud, see e.g. O'Brien et al. [2]. Silaghi et al. [3] present a combination of global and local techniques for skeleton fitting. Global techniques consider the whole skeleton at once, while local techniques adapt only a limited number of bones. Using a closed form alternative to the least-squares methods reduces the computational complexity of skeleton fitting, as shown by Knight and Semwal [4]. Zordan and Horst [5] proposed a physically based approach to skeleton fitting where the marker points were used to compute external forces acting on a skeleton model.

An integrated approach for both robust marker tracking and skeleton fitting is presented by Hornung and Sar-Dessai [6]. In this work clique-based recognition was used to identify indistinguishable markers, as described by Ringer and Lasenby [7]. To solve the problem of missing markers, inverse kinematics, continuity assumptions and other heuristics were used. In general this approach successfully tackles some fundamental problems of MoCap. However, as it has to use several markers to build the cliques which might restrict the mobility of the tracked person, it is not applicable in all environments.

A hybrid approach as presented in this work seems to be a unique investigation in the field of motion tracking. Other work mostly focuses on enhancing tracking performance by combining different tracking technologies. Combining different algorithms for performance enhancement is seldomly used.

3 Algorithm Description

Hybrid methods encompassing several algorithms promise higher stability in terms of recognition rate. They try to combine the advantages of several methods while avoiding their different disadvantages. In addition, they can combine several algorithms in a pipeline, i.e. certain algorithms pre-process the data for other algorithms. This either enhances the processed data or it transforms the data to meet certain requirements for later processing algorithms.

The hybrid tracking approach explained in this paper is composed of two main steps, organized in a pipeline. First (Sect. 4), an internal model is calculated that represents and identifies the markers in an initial MoCap posture. Second (Sect. 5), neural network learning is used to subsequently adapt the skeleton to the data points of consecutive MoCap frames through time.

4 Building an Internal Model

For capturing human motion, people are equipped with markers on special points of their bodies. These markers are tracked and captured to a database. The markers are not labeled by the tracking system, they are 3D points without additional information. As a consequence, each posture of a human motion is represented as a point cloud in the database. For marker identification, first an internal model has to be initialized that maps markers to body landmark points out of the point cloud.

4.1 Prerequisite: User-Created Skeleton Model

A skeleton-like structure is used to develop a description of the participant's body, in particular the respective anatomical landmarks. Skeletons are similar to trees, they consist of joints (like nodes) and segments (edges, connecting the joints). Therefore, they form branches that consist of joints that are linked in a chain, e.g. arms and legs. For the skeleton model these branches will be called subchains.

Fig. 1. The user-created skeleton model used in this work

As a prerequisite, the user has to create a skeleton structure first. This can be done once as the skeleton will be used for all MoCap data captured with similar marker configurations. The provided skeleton does not need to precisely reproduce the human skeleton in size or proportions. Rather, it needs to specify an approximated link structure of the marker configuration on the subject's body. Fig. 1 shows the skeleton model used in this work. It consists of 19 joints, organized in 4 subchains, left and right arm and left and right leg including left and right markers at the pelvis. This simple model is sufficient enough to conduct our experiments as well as many other human motion experiments. More complex models, e.g. including the spine, could be used as well, though requiring more complex calculations afterwards.

4.2 Aligning a Skeleton Model

Acquiring a properly aligned internal model is a helpful first step to identify markers and therefore reconstruct the postures done by the test subject in each frame. To configure the internal model, a body posture out of the MoCap data is used that has to accompany the user-created skeleton, such as standing with the arms reaching out (T-Pose), as it is used in this work.

First, the skeleton model is created with normalized extensions, i.e. coordinates are in the range $0 - 1$. This makes it easier to adopt the model later to the initial MoCap posture. Joints in this skeleton are labeled according to the corresponding anatomical landmarks or joints of the body.

When querying the initial posture from the database, the returned point cloud is arbitrarily positioned and oriented. To be able to fit the normalized skeleton to this point cloud the two main axes of the body are calculated using the principal components analysis (PCA). The PCA transforms the data by projecting it onto a set of orthogonal axes which indicate the directions of greatest variance in the data. In our case, the point cloud is projected onto a 2-dimensional space, spanned by the two main PCA axes. This leads to a properly aligned and flattened point cloud, which is oriented such that the arms reach out in the direction of the x axis. The feet and head are oriented along the y axis.

As there is nothing known about the orientation of the two main axes of the PCA, the left and right side of the body could be confused. Therefore, the orientation of the head (it is equipped with a 6DOF body target) is multiplied with the vector of the PCA x axis. The result of this scalar product reveals parallelism (result > 0) or anti-parallelism (result < 0). If they are anti-parallel, the orientation of the x axis is swapped to its opposite direction yielding correct PCA orientation.

After this, the normalized skeleton model is scaled to the extensions on the x and y axis. The skeleton model is thereby properly aligned to the projected point cloud in PCA space. This scaled skeleton model is retransformed into the normal 3-dimensional space and moved to the median point of the test subject's point cloud. As a result, the model is now aligned to the subject's initial posture.

Fig. 2 shows the initialization steps for an initial model. After the initial model was created, it is trained to the initial posture. For this, the skeleton structure

Fig. 2. Left: Loaded MoCap data from database (spheres) and calculated PCA (coordinate system), right: initial model (line)

is considered as a SOM (see Sect. 5) with the joints being the neurons. After training with the initial posture, the neurons correspond to the skeleton joints and the joints correspond to the markers. This makes it possible to map neurons aka skeleton joints (which are named) to markers, therefore identifying them.

4.3 Tree-Based Optimization for Skeleton Model Initialization

Often T-Pose postures are not available. Therefore it might be more feasible to use an optimization algorithm to fit the skeleton directly to an initial, more complex posture. The idea is to use anthropometric data of a person to compare this data to possible marker configurations and choose the best match. Again, a user-created skeleton is used (see Sect. 4.1) with the difference that only the hierarchy of the skeleton is needed.

Anthropometric Data. Anthropometric data contains fundamental information about the participant's body characteristics. Anthropometry is often part of MoCap experiments, usually as an initial step when marking anatomic landmarks. Segment lengths in the skeleton model can be computed out of anthropometric data. To accomplish this the following anthropometric values are used: Body length, acromial, iliocristal height, width of shoulder, pelvic width, height of merion laterale, height of supratarsale fibulare, hand length, forward range of hand, forward grasping range of hand, foot length, length of upper arm, length of lower arm. All of them are measured on the subject's right side.

The skeleton segment lengths are computed mainly using the theorem of Pythagoras. Lower and upper arm do not need to be calculated as they are already given. The resulting segment lengths are used as reference values for comparison with generated segment lengths, as explained in the following.

Constructing the Tree. To find an optimal mapping between markers and anthropological points, a tree is constructed that contains all possible mappings. For the optimization purpose it is additionally annotated with costs,

denoting discrepancies between the reference segment lengths and the currently constructed segment lengths. Nodes are created in three dimensions, one dimension for the depth of a tree, one for joint names and one for the markers. The maximum number of tree nodes is therefore:

$$\sum_{i=0}^{depth-1} number(joints) \cdot number(markers - i) \quad (1)$$

For the tree construction, all reference segment lengths and all unlabeled markers, as well as the skeleton model are needed. After construction the tree contains nodes for every possible marker to joint mapping, annotated with costs. Branches in the tree represent possible marker configurations corresponding to subchains in the skeleton.

Initial postures must contain all the markers that correspond to the joints in the skeleton model. The tree construction algorithm starts from the root node which corresponds to the head in this work. Nodes are created in all three dimensions of the tree: Joint names are considered that correspond to the skeleton model and nodes are created for all markers. Each node is used as the parent for the next hierarchical level in the skeleton model aka all child joints. In consequence, all possible marker configurations are constructed for each subchain in the skeleton structure. Leaf nodes are the nodes created for the end joints of a subchain.

Costs are calculated for each node. The previously computed segment lengths are compared to distances between the nodes' and parent nodes' associated markers. These costs are accumulated, i.e. each node's costs are added with the accumulated costs of the node's parent. The root node initially contains a value of 0. Additionally, this leads to a possible optimization: Branches in the tree that contain high accumulated costs can be abandoned as the difference between their marker configurations and the reference segment lengths is already high.

An example for such a tree is shown in Fig. 3. It contains a head marker node as the root of the tree. For each child of the head in the predefined skeleton, a set of nodes is created, which are named `l_shoulder`, `r_shoulder`, `l_pelvis` and `r_pelvis`. Child nodes of the head are created in each set. These nodes represent all tracked markers and contain the first discrepancies between the reference segment lengths and differences between any markers' positions and the head marker's position, annotated as costs. Then, children sets for every node in the `l_shoulder` set are constructed, here these are only `l_elbow` sets. Again nodes for all markers are created and connected to the respective parent node. The costs for these nodes are calculated and added to the costs of the parent node. This process is continued until the leaf nodes of the tree have been created.

To sum up, all possible marker configurations are denoted in a 3-dimensional tree resembling the predefined hierarchical skeleton structure. To facilitate determining the best-fitting configurations, the nodes have been annotated with accumulated costs. After the tree has been built, the best leaf has to be found for each branch that corresponds to a subchain in the skeleton model (here for each arm and leg) using the accumulated costs. For each end joint name of the

Fig. 3. Partial view on a constructed 3D tree, containing costs and node connections

skeleton model the corresponding leaf with the least accumulated costs is searched by comparing all costs of leaves corresponding to this joint name.

5 Neural Network for Adapting the Model

When initialization is finished, the SOM is adapted to each frame of the MoCap data. During this process the SOM is used to map joint names to the markers, therefore labeling the markers. Adaptation steps are computed several times for each frame in the MoCap data, finishing when certain abort criteria, like distances between markers and neurons being low or reaching a certain number of iterations, are achieved.

Initialization: Creation of the SOM and a Reference Model. First, the SOM has to be created. After the initial model was generated, the neurons are linked to the joints of this model. This reference model is used to keep the neural net consistent with regards to the skeleton structure. Whenever markers are occluded it is used for keeping the distances between joints approximately the same. However, the following usual SOM learning rules apply.

Step 1: Stimulus and Response. For the current frame, the position of each tracked marker is presented to the SOM, with $\mathbf{x}$ being the training vector. Distances to all prototype vectors are then computed, using Euclidean distance measure. The neuron with its prototype closest to $\mathbf{x}$ is the winning neuron b . The prototype vectors were set in the initialization step (see Sect. 5), when the SOM was created out of the initial skeleton.

$$\|\mathbf{x} - \mathbf{m}_b\| = \min_i \{\|\mathbf{x} - \mathbf{m}_i\|\} \quad (2)$$

Step 2: Adaptation. For adaptation, first the winner neuron is moved towards the current MoCap point, with a certain learning rate ϵ and learning radius σ . Each other neuron is also adapted towards this point but, in contrast to the usually applied learning rules, with much lower learning parameters. Such lower learning parameters seem to achieve a good compromise between generalization and specialization.

The prototype vectors are updated according to the following update rule:

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + \epsilon H_{bi}(t) (\mathbf{x} - \mathbf{m}_i(t)) \quad (3)$$

t is the current iteration. $H_{bi}(t)$ is a neighborhood kernel centered at the winning neuron vector:

$$H_{bi}(t) = e^{\left(-\frac{\|\mathbf{m}_b(t) - \mathbf{m}_i(t)\|^2}{2\sigma^2}\right)} \quad (4)$$

Step 3: Identifying Markers. Steps 1 to 3 are usually computed several times to adapt properly to a posture. After adaptation is finished, the markers have to be labeled. This can be done by using any nearest neighbors computation algorithm. The nearest marker of a neuron gets the name of this neuron. After this the model is fully adapted to the markers and the markers are labeled according to the names of the nearest neurons. Fig. 4 shows an example tracking for several frames.

Fig. 4. Tracking of several frames

6 Evaluation

For capturing motion data, we used a passive optical tracking system with 8 cameras operating at 50Hz made by ART GmbH, Germany. Such systems are nowadays widely used in the field of real-time interaction with virtual environments. An experiment with 30 participant was conducted. Each participant was equipped with one 6DOF body above the head as a reference point. For tracking the complete body, the participant was additionally equipped with 18 3DOF markers on the anatomic body landmarks. The joints tracked are the same as the joints of our skeleton model (see Sect. 1). The tracking cameras span a measure volume of $2.8m \times 3.8m$ wide and $2.2m$ high.

A participant performed several motions. It first moved to the initial pose (see Sect. 4). After that it performed some arm movement and body movement. Additionally, the initial pose was performed on several different locations with different orientations. Finally, movement while using an interaction device was captured.

6.1 PCA-Based Aligning

The participant was orientated in arbitrary directions to check the method's capabilities to cope with such situations. Especially the PCA algorithm, as it is used here to calculate position and orientation of the participant, was evaluated. The algorithm works 100% correctly for T-Pose postures and for frames where all markers are present.

6.2 Tree-Based Optimization

Anthropometric dimensions were measured for each participant. After MoCap, some postures out of the MoCap data were chosen as test input data. After presenting the data to the system, the correct assignment of the recognized marker configuration was checked manually. 80% of the frames were recognized correctly. The rest was recognized partially, e.g. arms were correct but the legs were mixed. One reason for this could be the accuracy of the participant's anthropometric data. Using additional constraints to guide the algorithm yielded a 100% recognition rate.

6.3 Neural Network

To test the algorithm, a part of the data set, that contained only few marker occlusions was used for training and testing. This data was additionally obfuscated with one marker being occluded for a longer time. As it is not easy to measure the error (there is no reference here) it can only be quantitatively stated that the SOM performed very well for this situation.

Finally, the rest of the data, where a lot of markers were occluded (each once a while up to only four markers were seen), was used for adapting the SOM. This data even contained situations where the whole left arm data was missing for approximately 2 seconds, being a hard condition for the algorithm. Markers just disappear and pop up on completely different positions. Unfortunately, the SOM is not very well fitted to track a lot of missing markers over a longer period of time, and, for human motion, even 2 seconds are a long period of time. During this time the SOM tries to adapt to the other markers and can completely lose track on its corresponding, but missing marker. It can even happen that other neurons adapt strongly to a reappearing marker, because in the meanwhile it moved to their position and they also lost track to their corresponding markers.

7 Conclusion and Future Work

Many tracking algorithms require time-consuming manual initialization processes, at least to gather initial information for a subsequent automatic tracking of the markers. This work presents a mostly automatic and robust approach to identify and track markers on a human subject. An initial pose out of Motion Capture data is used to generate an internal model of this pose. This internal model is then represented as a self-organizing map which can adapt easily to new subsequent poses out of the MoCap data.

Even though such a hybrid approach achieves quite robust tracking, it still has some manual actions left, namely setting parameters for the algorithms and search for a good start posture. It might be interesting to integrate other methods to further increase robustness and automatic processing. Integrating other motion prediction methods might also help the tracking process and guide the SOM.

References

1. Dorfmueller-Ulhaas, K.: Robust Optical User Motion Tracking Using a Kalman Filter. Technical Report 2003-6, University of Augsburg, Institut für Informatik, Universitätsstr. 2, D-86159 Augsburg (May 2003)
2. O'Brien, J.F., Bodenheimer, R., Brostow, G., Hodgins, J.K.: Automatic Joint Parameter Estimation from Magnetic Motion Capture Data. In: Graphics Interface, pp. 53–60 (2000)
3. Silaghi, M.C., Plänkers, R., Boulic, R., Fua, P., Thalmann, D.: Local and Global Skeleton Fitting Techniques for Optical Motion Capture. In: Magnenat-Thalmann, N., Thalmann, D. (eds.) CAPTECH 1998. LNCS, vol. 1537, pp. 26–40. Springer, Heidelberg (1998)
4. Knight, J.K., Semwal, S.: Fast Skeleton Estimation from Motion Capture Data using Generalized Delogne-Kasa Method. In: Skala, V. (ed.) WSCG Conference Proceedings, WSCG (January 2007)
5. Zordan, V.B., Horst, N.C.V.D.: Mapping Optical Motion Capture Data to Skeletal Motion using a Physical Model. In: SCA 2003: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation, Aire-la-Ville, Switzerland, Switzerland, Eurographics Association, pp. 245–250 (2003)
6. Hornung, A., Sar-Dessai, S.: Self-Calibrating Optical Motion Tracking for Articulated Bodies. In: VR 2005: Proceedings of the, IEEE Conference, on Virtual Reality, Washington, DC, USA, pp. 75–82. IEEE Computer Society Press, Los Alamitos (2005)
7. Ringer, M., Lasenby, J.: A Procedure for Automatically Estimating Model Parameters in Optical Motion Capture. *Image and Vision Computing* 22(10), 843–850 (2004)