Neurocomputing 147 (2015) 174–184

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Self-organizing maps for hand and full body tracking

Foti Coleca [a,c,*], Andreea State [a,b], Sascha Klement [c], Erhardt Barth [a], Thomas Martinetz [a]

[a] Institute for Neuro- and Bioinformatics, University of Lübeck, Ratzeburger Allee 160, 23538 Lübeck, Germany[1]
[b] University "POLITEHNICA" of Bucureşti, Splaiul Independenţei 313, 060042 Bucureşti, Romania[2]
[c] gestigon GmbH, Maria-Goeppert Straße 9a, 23562 Lübeck, Germany[3]

ARTICLE INFO

Article history:
Received 10 April 2013
Received in revised form
27 October 2013
Accepted 31 October 2013
Available online 20 June 2014

Keywords:
Body tracking
Hand skeleton tracking
Gestures
Self-organizing maps
Kinect
TOF cameras

ABSTRACT

Touch-free gesture technology opens new avenues for human–machine interaction. We show how self-organizing maps (SOM) can be used for hand and full body tracking. We use a range camera for data acquisition and apply a SOM-learning process for each frame in order to capture the pose. In a next step we introduce an extension of the SOM to 1D and 2D segments for an improved representation and skeleton tracking of body and hand. The proposed SOM based algorithms are very efficient and robust, and produce good tracking results. Their efficiency allows to implement these algorithms on embedded systems, which we demonstrate on an ARM-based embedded platform.

1. Introduction

The challenge of human hand/body tracking and pose estimation has gained much attention during the last years, mainly driven by the mainstream interest toward building usable gestural interfaces for consumer applications. This was seen with the introduction of gaming consoles that can track a user's hand gestures (Nintendo Wii) or body (Microsoft Kinect), which showed that gesture interfaces can be used to create rich interactive experiences. Hand tracking alone can be used in a wide variety of applications and represents a milestone in human–machine interaction. A major catalyst was the introduction of new technologies and devices designed for 3D image acquisition. Depth cameras provide a more favorable framework for tracking algorithms, simplifying the task of three-dimensional model fitting, giving algorithms that use them an edge over 2D image processing techniques.

Nevertheless, these are both difficult problems, especially estimating the hand pose: the hand itself is a complex object, having an extremely large state space due to its 27 degrees of freedom [1]. Because of this complexity, its projection in images often involves self-occlusions which, coupled with the chromatic uniformity of the skin, makes segmentation and feature detection very difficult. With speeds reaching up to 5 ms^{-1} for translation and 300°s^{-1} for wrist rotation [2], consecutive frames of a moving hand can have very little in common (especially with a slow camera frame rate), making it a difficult object to track. Adding to these difficulties, the algorithms have to cope with various backgrounds and lighting conditions.

3D cameras can alleviate some of the difficulties described above, having multiple advantages over standard color image processing. A critical step of any pose estimation algorithm is object segmentation. By having access to the depth map of the scene, objects can be segmented accurately based on their shape and distance to the camera, regardless of texture, skin color or background clutter. With active 3D technologies (such as time-of-flight or structured light), there is even no need for a uniform or consistent scene illumination. This is a very useful feature for real-world applications, where consumer devices are being used by a variety of people in a variety of environments. Our work focuses on building a hand/body pose estimation and tracking algorithm for such a 3D camera, that is both accurate and has low computational costs.

In this article we present an extension of the hand pose estimation method proposed in [3], as well as a practical implementation of the algorithm. It is all based on the original work introduced in [4], a novel approach to pose estimation by the use of self-organizing maps (SOM) [5] to fit a topology of the human

* Corresponding author at: Institute for Neuro- and Bioinformatics, University of Lübeck, Ratzeburger Allee 160, 23538 Lübeck, Germany.
E-mail addresses: coleca@inb.uni-luebeck.de (F. Coleca), state@inb.uni-luebeck.de (A. State), sascha.klement@gestigon.de (S. Klement), barth@inb.uni-luebeck.de (E. Barth), martinetz@inb.uni-luebeck.de (T. Martinetz).
[1] http://www.inb.uni-luebeck.de.
[2] http://www.upb.ro.
[3] http://www.gestigon.de.

http://dx.doi.org/10.1016/j.neucom.2013.10.041
0925-2312/

upper body inside a 3D point cloud. We show that this topology can be successfully extended to a full body as well as a human hand. A further extension of the algorithm is presented, in which the original SOM is extended to include not only nodes but also the segments and planes between the nodes of the topology. This has the advantage of requiring less nodes than the original topology, offering a more realistic representation of the human hand and being more stable overall.

The algorithms are of a low enough computational cost that they can be implemented on an embedded platform and used to track subjects in real time. We will show an implementation of both the hand/body SOM and the segment-plane extension for the hand on an OMAP-4430 powered Pandaboard, using time-of-flight (PMD camboard) or structured light (Microsoft Kinect) cameras as an input device for 3D data. Our algorithm is able to track the user at the native framerate of the camera.

2. Related work

The most commonly used methods to gather accurate data for skeleton tracking are marker-based motion capture systems in the case of the whole-body skeleton or by the use of a "data glove" for hand pose estimation [6]. These methods are cumbersome and can be used only in controlled environments. Thus, marker-less pose estimation is a heavily researched area in image processing — recent surveys cite dozens of papers [2] on hand pose estimation and several hundred [7] on human motion capture and analysis.

For example, the authors of [8] use kinematic models and build a hand state model, which consists of a set of lines and points generated by the projection of the hand model to the image plane. Hand pose estimation based on features derived from projections of the hand and its shadow is presented in [9]. The method requires controlled background and lighting and is susceptible to occlusion. In [10] and [11], the authors use a feature extraction approach based on Curvature Scale Space to achieve translation, scale and rotation invariant recognition of hand postures. Again, the method is tested in a controlled environment, as it requires an accurate segmentation of the hand contour.

The authors of [12] introduce a machine learning architecture for matching image features to 3D hand example poses, which requires to solve an optimization problem based on Bayes' rule. Another approach is to estimate the hand pose with a database of synthetic hand images. For instance, in [13] an indexed image database is used to retrieve the closest hand match, with an adapted chamfer distance and line matching algorithm. In [14], the authors implement a cascade of increasingly complex classifiers to determine the hand pose from synthetic training data. In order to better handle occlusions, particle filters can be used. In [15], the authors apply a meta-descent algorithm to minimize the distance between a predicted position and the observed position, while particle filters predict new sample positions and help the optimization algorithm to recover from local minima. As shown in [16], the combined usage of intensity images and range information provides a good framework for body tracking.

Regarding performance, most algorithms surveyed by Erol et al. [2] stay below 30 frames per second (which we regard as being real-time), with only one exception [17]. Other solutions leverage the computing power of the GPU in order to achieve high frame rates [18–20]. Most existing approaches are aimed at high-performance desktop machines.

3. The SOM tracking algorithm

The node-based SOM tracking algorithm (which we will refer from now on as the Standard SOM Algorithm) starts with the initialization of its network weights, followed by the iteration of two steps: the competition and the update of the weights. At every iteration, a sample point from the dataset is randomly chosen. First, during the competition phase, a winner node (i.e. the weight with the minimum Euclidean distance to the sample point) is computed.

Given a network with n neurons and a sample point $\mathbf{x} \in \mathbb{R}^3$, we determine the winner node $\hat{i}$ as follows:

$$\hat{i} = \arg\left\{ \min_i \|\mathbf{x} - \mathbf{w}_i\|_2 \right\},\ i = 1, \ldots, n \tag{1}$$

with $\mathbf{w}_i \in \mathbb{R}^3$ being the weight of node i. Next, the update phase aims at decreasing the distance between the winner-node weight and the sample point, by an amount given by the learning rate $\epsilon(t)$. First, let us define the learning rate function as

$$\epsilon(t) = \epsilon_i \left(\frac{\epsilon_f}{\epsilon_i} \right)^{t/t_{max}}, \tag{2}$$

where ϵ_i is the initial learning rate, ϵ_f is the final learning rate, t is the current iteration, and t_{max} is the maximum number of iterations performed on the network. Then, the weight $\mathbf{w}_{\hat{i}}$ is updated at step t according to

$$\mathbf{w}_{\hat{i}}(t+1) = \mathbf{w}_{\hat{i}}(t) + \epsilon(t)(\mathbf{x} - \mathbf{w}_{\hat{i}}(t)). \tag{3}$$

The standard SOM algorithm then also applies a neighborhood update, in the sense that not only the winner-node weight is updated, but also the weights of the neighbor-nodes (in general with a smaller learning rate). In our case we updated only the direct topological neighbors of the winner node according to (3), but with a learning rate of $\tilde{\epsilon}(t) = \epsilon(t)/2$.

These steps are repeated for hundreds or thousands of iterations. This makes the skeleton graph fit to the point cloud and stay within its confines.

4. Topology expansion

First, we expand the 44-node upper body topology presented in [4] (Fig. 1a) to two topologies, one representing the whole body (Fig. 1b), and the other representing the human hand (Fig. 1c). The models were chosen so they mimic the anatomical landmarks of their real-world counterparts – limbs and joints for the body and phalanges and interphalangeal joints for the hand. The rigid bodies (torso and palm) are modeled as a mesh. Both produce good qualitative results in our implementation. The end results achieved with the standard SOM for the hand and body are shown in Figs. 2 and 3, showcasing the method's robustness.

It can be seen that the hand tracker is able to cope with missing data (Fig. 2b,c as white areas on the palm), the skeleton's topology remaining stable, the fingers being retracted in the palm. This is considered to be correct behavior, as the fingers will be reported as "bent" to a subsequent gesture recognition algorithm.

For the full-body tracker, the topology is robust enough to perform a good fitting over the subject's body, even when there is occlusion occurring. This is shown in Fig. 3b–d: it can be seen that when the user crosses his arms in front of him the skeleton retains its geometry afterwards, even if one of the arms occludes the other. This is true also for the rest of the topology nodes, such as the torso. Fig. 3a shows how the skeleton tracks the body shape in 3D, following the user's leg even though it is not in the same plane as the body.

Such configurations would be very hard to track using just a 2D image, particularly because of the juxtaposition of the hands and torso. This issue is resolved by using a 3D camera, which can differentiate between surfaces of various depths.