Refining Joint Locations for Human Pose Tracking in Sports Videos

Dan Zecha, Moritz Einfalt, Rainer Lienhart
University of Augsburg
[dan.zecha, moritz.einfalt, rainer.lienhart]@informatik.uni-augsburg.de

Abstract

The estimation of an athlete's pose in video footage enables the automation of athletic performance assessment, the prediction of motion kinematics and dynamics in sports videos and the possibility of technology-assisted, direct training feedback. Despite remarkable progress in the field of deep learning assisted human pose estimation, the performance of such systems decreases while noise and errors increase with the complexity of the scene. In this paper, we focus on aquatic training scenarios, where even novel pose estimators produce several types of orthogonal errors, including joint swaps and prediction outliers. In order to improve the estimation of an athlete's pose in swimming, we propose a graph partitioning problem that connects pose estimates over time and explicitly allows for joints to switch labels if their location better fits each other's trajectory. We optimize the problem using integer linear programming, which partitions the graph into the most probable joint trajectories. We show experimentally that our method of joint rectification improves the joint detection precision of swimmers in a swimming channel by $0.8\% - 4.8\%$ *PCK for anti-symmetrical motion and up to* 1.8% *PCK for symmetrical styles.*

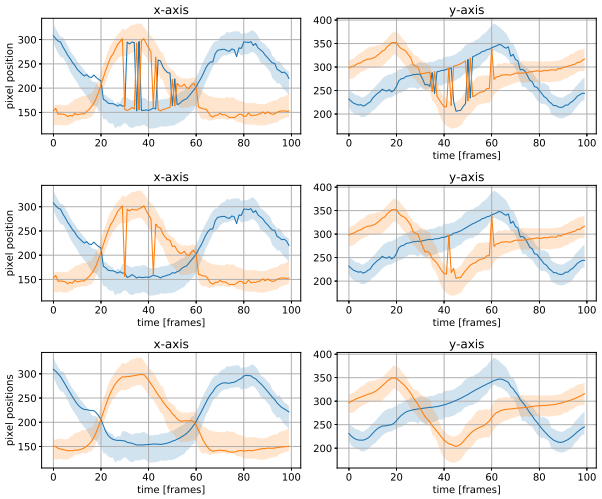

Figure 1. A sequence of joint coordinates of a freestyle swimmer's left (blue) and right (orange) elbow. The first column depicts x coordinates, the second y coordinates. Top: original sequence as predicted by fine-tuned [26]. Middle: after partner graph optimization. Bottom: final result after outlier and noise correction. Colored corridors depict the ground truth time series with $\pm$0.2PCK. Qualitative examples from this sequence are depicted in Figure 2.

1. Introduction

State-of-the-art pose estimation systems like Convolutional Pose Machines (CPM, [26]) or Mask R-CNN [13] enable the continuous estimation of the human pose in sports footage. While state-of-the-art pose estimation algorithms often perform well on benchmark video footage and less complex scenes, the output of such algorithms can be inaccurate and partially incorrect in a visually demanding training scenario. Human pose estimation in aquatic environments is exacerbated by noise from air bubbles, water splashes, constant self-occlusion and refraction, which often impede the prediction of the human pose and consequently the retrieval of key-poses. The unique perspective on a swimming channel entails an additional challenge. The side view exhibits visually ambiguous poses for anti-symmetrical body movements, thereby inducing many false detections from swapped joint pairings.

For human pose estimation to be a useful tool for fully automated, continuous performance analyses and training feedback, high expectations of coaches have to be met with appropriately precise detection results. A straight forward application for pose estimation in sports is visualization, which allows for enhancing images with pose inpainting, therefore illustrating the motion qualitatively. Rectified pose estimates lead to a better pose visualization, thus assisting coaches with a qualitative pose assessment by removing joint landmark wiggling between frames. As a tool for quantitative evaluation, pose estimation can be used to count steps, strokes or kicks, and it can be used to automatically and precisely measure execution times, body angles, jump distances as well as motion kinematics of athletes.

In this paper, we strive to improve human pose estima-

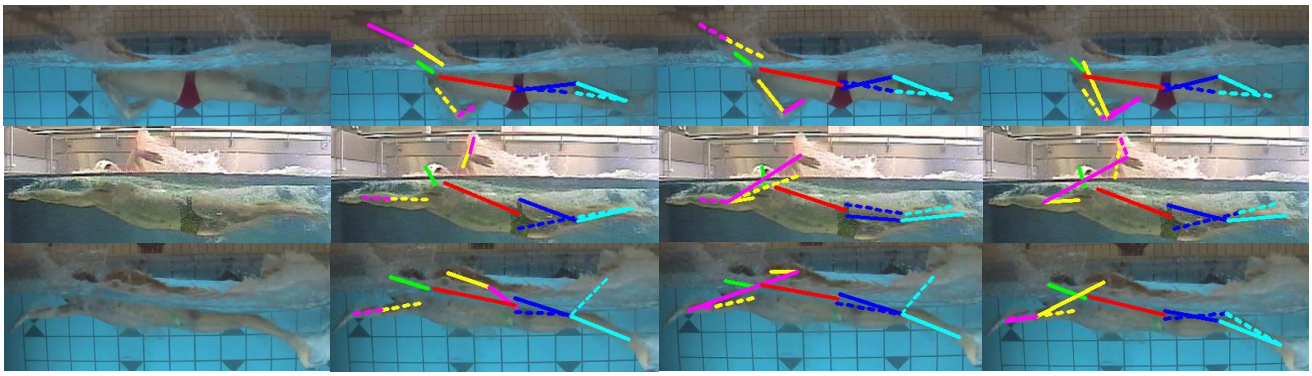

Figure 2. Qualitative examples of different errors. From left to right: original image, ground truth annotations, pose prediction CPM [26], pose prediction Mask R-CNN [13].

tion in aquatic environments. Therefore, we identify a taxonomy of several orthogonal pose estimation errors, which are much more prevalent in aquatic training scenarios. Our goal is to rectify them or, if complete rectification is not possible, at least minimize the overall error magnitude. We achieve this by constructing a weighted graph from pose estimates, where edges connect joint detections over different frames and edge weights encode transition probabilities. To account for swap errors, the proposed graph structure explicitly allows for detections to change their labels, e.g, a detection for the right wrist can be labeled a left wrist after optimization if the right wrist detection better fits the trajectory of the left wrist. Formulating this graph as an integer linear program (ILP) allows for partitioning the graph into subsets of edges, where each subset represents the most probable trajectory for one joint. In a second step, joint trajectories are smoothed and filtered for outliers using a locally robust regression through L1-norm minimization of trajectory residuals.

Contributions. With the proposed method, we make the following contributions: (a) Our approach *explicitly* models the possibility that joints may change their label in order to tackle the problem of joint transposition in human pose estimation sequences. (b) We propose a novel scheme for modeling transitive edges and edge potentials in the optimization graph, which is inspired by motion kinematics. (c) We present an analysis of potential pose estimation errors that are more frequent in sports applications. (d) Our pipeline is merely based on joint coordinate predictions and can be trained with a comparatively small set of annotated training sequences. Obtaining large quantities of high precision annotations to train sophisticated machine learning algorithms remains a time-consuming and tedious challenge which is often not feasible, especially in exotic applications and less popular sports.

The rest of this paper is organized as follows. We first analyze sequences of consecutive pose estimates produced by a refined version of [26] qualitatively and quantitatively to give a better sense of error types, correlation, and frequency. Build on this analysis, a pipeline for joint tracking and joint regression is presented. The experimental section reports improvements for different stages.

2. Related Work

Pose estimation in images has made remarkable progress over the last years. Deep neural networks have been used for learning visual appearance [28, 25] and structure [5, 26, 27, 13, 30, 23] of the human pose. Deep architectures have been proposed for improving pose estimation in videos. For instance, [22] stack multiple contiguous frames as parallel input for a network, while [4] leverage a recurrent network structure for improving pose estimates. [15] incorporate human action into a deformable part model to improve pose estimation in videos and propose a procedure for jointly training both pose and action in one model. [12] feed additional pose priors from previous detections together with an image into a CNN. [8] propose a late fusion network, where the predictions of multiple frames are merged and processed to improve temporal consistency. While all these methods enforce consistency by leveraging some type of smoothing constraints, others [20] favor optical flow as an additional input modality. Recently, multi-person tracking has gained a lot of attention from the scientific community. [16] use LP optimization on a spatiotemporal pose graph for tracking multiple persons. [11] use [13] for keypoint prediction in small clips and link them over an entire video using a lightweight tracking scheme. While both multi-person tracking solutions are structurally similar to our method, none of them allows for explicit label swapping.

Pose rectification can also be seen as a ”data cleaning” task. To this, recurrent frameworks [19], conditional Boltzmann machines [24], Kalman filtering [3], dimensionality reduction [1] or data-dependent random forests [9] have been successfully applied to identify and rectify outliers in

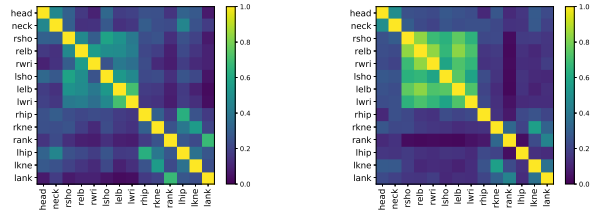

Figure 3. Pearson correlation for residual magnitudes of different joints for freestyle (left) and backstroke (right) swimmers. Larger correlation coefficients emerge for the upper body joints and especially for partner joints, and separately for the lower body joints.

human poses and other modalities. Specifically in sports, [14] combined global and local pose estimation to refine joint predictions of athletes. Human pose estimation in live sports footage has been addressed by [10], who propose a generative learning algorithm for athlete tracking. [29] perform pose rectification for an athlete by means of a convex optimization problem based on motion kinematics of an athlete. Compared to our approach, their solution involves a strictly causal dynamic program, which does not allow for incorporating detections from future frames.

3. Prediction Error Taxonomy

Even novel pose estimation system like [26, 13] have difficulties dealing with the visual ambiguity of the pose from a side-view and to accurately produce reliable outputs. In the following, we discuss different error modes in order to convey a deeper understanding of why it is necessary to address them. Therefore, we performed an analysis of orthogonal joint errors on pose estimates produced by a Convolutional Pose Machine (CPM, [26]), which we fine-tuned on a set of 55 sequences covering 4000 images of freestyle and backstroke swimmers in a swimming channel (see Figure 2 for examples). We focus this assessment on only two of the four major swimming styles because we found that the pose detectors produced most errors, quantitatively and qualitatively, on them. However, the error classes we observe in different swimming styles are at least a subset of what is discussed in the following.

A disadvantage of using a per-frame pose estimator is that no prior temporal consistency between contiguous poses is enforced for improving the temporal stability of continuous pose estimates. In Figure 2, different types of errors are depicted. They usually fall into one of four categories. A **joint transposition** occurs if the estimator swaps joints that are visually similar. For instance, the right wrist is falsely detected as the left wrist and vice versa. In our pose parametrization, there exist six visually similar *partner joints*, which are the wrists, elbows, shoulders, right and left hip, knees, and ankles. Swaps are correlated with visually challenging poses, where the detector is not able to properly distinguish between body sides. This error class is, therefore, most prevalent in antisymmetrical motion, e.g., in freestyle or backstroke swimming. We can find a strong indication of joint swaps being a problem when plotting the correlation between residual magnitudes of different joints for freestyle and backstroke swimmers (Figure 3). It becomes apparent that there is a strong correlation of error magnitudes in the upper body and separately in some joints of the lower body. With correlation coefficients for partner joints reaching values of > 0.8, a larger error in one joint is often answered with an equally large error in the partner joints.

A joint estimate is considered an **outlier** if its residual magnitude is too large. This includes outliers where one joint is placed at the position of the partner. Outliers are classical false positive detections which can often be corrected using robust interpolation. A **false negative** is a ground truth location that was not estimated by the joint detector. All other joint detections that are not predicted precisely at the ground truth location but in close proximity of a joint fall in the last category. This **location jitter** has a joint-specific variance and usually leads to volatile joint trajectories.

Joint transposition and outlier errors are typically considered false positives, while misses are false negative detections. In the context of the Percentage of Correct Keypoints (PCK) measure, the rectification of these error categories leads to a better recall at the largest inlier threshold. Improving joints on a finer scale by removing location noise from the trajectory mainly raises the PCK curve for smaller thresholds.

One important question to ask is how common different error types are. If each error class only rarely occurs, then correcting them would be a trivial task. Figure 1 visualizes the x and y coordinates (blue and orange) of an elbow of a freestyle swimmer. The top row of plots depicts the raw output of the CPM. These graphs are very exemplary for the error frequency we observe in the data: While joint swaps and outliers can appear as standalone events, more often than not they appear in bursts over a longer sequence of frames. In addition to the number of errors, the mixture of different error types proves challenging to tracking algorithms. As becomes apparent in Figure 3, joint transposition errors mix with single and consecutive outliers as well as general detection noise.

4. Joint Refinement

We discuss a pipeline of optimization problems, each of which is designed for identifying the error types discussed in the previous section with the goal to rectify them or, if complete rectification is not possible, at least minimize the overall error magnitude. All optimization problems are merely defined on the raw 2d joint location output of a

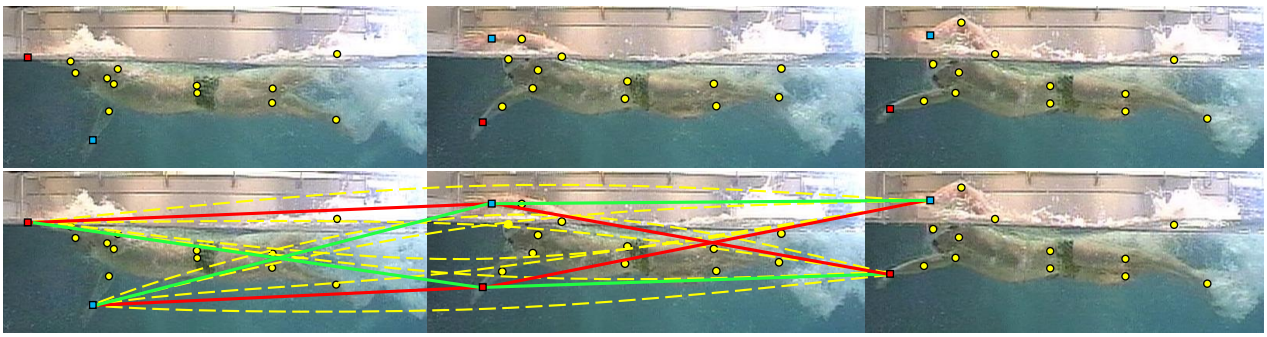

Figure 4. An example of a kinematic partner graph for wrist joints. Top: Three consecutive images with joint predictions. Wrists marker colors indicate joint labels as detected by [26] (swap error in left image). Bottom: ILP edges for wrists. Green edges connect joint predictions from adjoining frames. Red edges explicitly model label swaps between adjoining frames. Transitive edges (yellow) skip one frame, connecting joints from frames t with frames $t+2$.

pose estimator. Therefore, we examine two popular pose estimation frameworks: a CPM [26] and a Mask R-CNN [13] are used to produce joint estimates for each frame in a video depicting an athlete (Figure 4, top). The main focus of this paper is addressing the problem of joint transposition. A weighted graph is constructed from pose estimates, where edges connect joint detections over different frames and edge weights encode transition probabilities. We will use the terms *velocity- and acceleration edges* to differentiate between different edge types. This denomination is not by chance and hints on how the associated edge weights are constituted. Details are presented in Section 4.1.2. To account for swap errors, the proposed graph structure explicitly allows for detections to change their labels, e.g, a detection for the right wrist can be labeled a left wrist after optimization if the right wrist detection better fits the trajectory of the left wrist. Formulating this graph as an integer linear program (ILP) allows for partitioning the graph into subsets of edges, where each subset represents the most probable trajectory for one joint.

4.1. Swap Optimization

The partitioning of the weighted graph can be formulated as an ILP as follows. The goal is to optimize over

$$\begin{aligned} &\text{maximize } \mathbf{w}^T\mathbf{x} \\ &\text{s.t. } C\mathbf{x} \leq \mathbf{b} \\ &\mathbf{x} \in \{0,1\}^n \end{aligned} \tag{1}$$

Here, $\mathbf{x}$ is a binary vector where each entry x_i indicates if an edge is present ($x_i = 1$) or absent ($x_i = 0$). The weight vector $\mathbf{w}$ associates a weight w_i with each edge in $\mathbf{x}$. Matrix C and vector $\mathbf{b}$ encode constraints on the graph, e.g., required co-occurrence and mutual exclusivity for certain edges. Given a set of constraints, optimizing over the problem in (1) yields subsets of edges which maximize the sum of edge weights in the graph. The joint detections connected by a path of edges form the optimal trajectory w.r.t. (1). In the following, we will define edge types which are included in $\mathbf{x}$, how edge weights $\mathbf{w}$ are chosen and what constraints to formulate in order for the problem to be well defined and feasible.

4.1.1 Kinematic Partner Graphs

We define a *kinematic partner graph* as the foundation of the ILP. The nomenclature refers to the fact that edges in this graph model the relationship between partner joints and that the weights for each edge are derived from the motion kinematics of the athlete. A CPM is used to obtain sets of joint detections D_t for each frame t in a video. Each detection $d_j^{(t)} \in D_t$ of joint type j is associated with a joint pixel location $l_j^{(t)} = (x_j, y_j)^T$ in the image. The function $j' = \nu(j)$ returns the partner joint j' of joint j. For instance, if j is the identifier for the left wrist, $\nu(j)$ returns the identifier for the right wrist. Wrists, elbows, shoulders, hips knees and ankles are defined as partner joints, respectively. A *kinematic partner graph* is defined by edges connecting joint detections of the same type and partner joint detections over adjoining frames. Edges can be divided into two types. Let a set of *velocity edges* E_v connect detections over consecutive frames:

$$\begin{aligned} E_v = \{(d_j^{(t)}, d_{j'}^{(t+1)}) : d_j^{(t)} \in D_t \wedge d_{j'}^{(t+1)} \in D_{t+1} \\ \wedge\, j' \in \{j, \nu(j)\}\} \end{aligned} \tag{2}$$

Edges in E_v represent the trajectory of all joint detections as originally predicted by the pose estimator. Additionally, E_v includes connections between a detection $d_j^{(t)}$ of joint j in frame t and its partner joint $d_{\nu(j)}^{(t+1)}$ in frame $t+1$ to account for joint transpositions. These additional edges allow for joints to swap their labels if the associated weights

have a larger benefit for the optimization problem. For each pair of partner joints in two consecutive frames, there are four velocity edges. Adding to E_v, transitive edges or *acceleration edges* E_a are defined as

$$E_a = \{(d_j^{(t)}, d_{j'}^{(t+2)})_{j''} : d_j^{(t)} \in D_t \wedge d_{j'}^{(t+2)} \in D_{t+2} \wedge j', j'' \in \{j, \nu(j)\}\} \tag{3}$$

Acceleration edges connect detections of joint j in frame t with joint detections of the same joint type as well as the partner in frame $t+2$. Note the edge subscript j'' added to the edge definition. It indicates that there are two edges connecting $(d_j^{(t)}, d_{j'}^{(t+2)})$, namely $(d_j^{(t)}, d_{j'}^{(t+2)})_j$ and $(d_j^{(t)}, d_{j'}^{(t+2)})_{\nu(j)}$. Both these edges connect $d_j^{(t)}$ with $d_{j'}^{(t+2)}$, but have different weights. The weights are determined by which of the two joint detections $d_j^{(t+1)}$ and $d_{\nu(j)}^{(t+1)}$ at frame $t+1$ better fit the respective trajectory. This peculiarity leads to overall eight acceleration edges connecting each pair of partner joints over two frames. A fully build partner graph for the wrists in three consecutive images is depicted in Figure 4, where red and green lines are velocity edges and yellow lines depict acceleration edges.

4.1.2 Edge Weighting

The optimization problem in Equation 1 partitions subsets of edges by maximizing over the associated weights. The denomination of velocity and acceleration edges in the previous section is not accidental, as we will use first and second order numerical derivatives of the joint locations to determine edge weights.

Pose preprocessing. All predicted joint locations are resized relative to a reference upper body size s_{ref} to account for athletes of different size in images. In our experimental setup, the size of an athlete in a video is approximately constant. Hence, let s_{up} be the median length of an athlete's upper body size in a video, determined by the distance between right shoulder and left hip. Then, each original joint estimate at location $\hat{\boldsymbol{l}}_j^{(t)}$ for all joints j in all frames t is resized to

$$\boldsymbol{l}_j^{(t)} = \frac{s_{ref}}{s_{up}} \hat{\boldsymbol{l}}_j^{(t)}. \tag{4}$$

The estimate of s_{up} may be more complex if athletes are filmed in different training scenarios. For example, if pan shots are used and the size of the athlete in the footage changes over time, s_{up} needs to be continuously estimated, for instance by a smoothing spline over predicted upper body lengths.

Kernel density edge weights. A weight for each edge can be computed from first and second order numerical derivatives of joint locations. Let the velocity $v_j^{(t)}$ of a joint j in frame t be defined as

$$\boldsymbol{v}_j^{(t)} = \nabla \boldsymbol{l}_j^{(t)} = \boldsymbol{l}_j^{(t)} - \boldsymbol{l}_j^{(t-1)} \tag{5}$$

and the acceleration $a_j^{(t)}$ be the rate of change in velocity, thus

$$\boldsymbol{a}_j^{(t)} = \nabla^2 \boldsymbol{l}_j^{(t)} = \boldsymbol{v}_j^{(t)} - \boldsymbol{v}_j^{(t-1)} = \boldsymbol{l}_j^{(t)} - 2\boldsymbol{l}_j^{(t-1)} + \boldsymbol{l}_j^{(t-2)}. \tag{6}$$

A kernel density estimator (KDE) based on the parabola shaped Epanechnikov kernel $K(v; h) = 1 - v^2/h^2$ and a bandwidth of $h = 1$ is used to determine a probability for a observed velocity magnitude $|\boldsymbol{v}_j|$ (=speed). Therefore, a probability distribution is learned from ground truth speeds in a separate set of training videos. When formulating the ILP, the predicted velocity can be determined from predicted joint locations using Equation 5. The KDE is queried to determine probabilities for predicted speeds, which are then assigned to the respective velocity edge E_v. After all weights are assigned, edge normalization for all velocity edges connecting partner joints from frame t with frame $t+1$ is performed. Therefore, all velocity edges connecting two partner joints, i.e, $\{(d_{j'}^{(t)}, d_{j''}^{(t+1)}) : j', j'' \in \{j, \nu(j)\}\}$, are normalized such that the sum of weights equals one.

The same weighting process is repeated for all acceleration edges E_a. A KDE for each joint type is queried to determine probabilities for the predicted acceleration magnitude $|\boldsymbol{a}_j|$. When using acceleration edges to connect detections in frame $t-2$ with detections in frame t, Equation 6 tells us that detections from frame $t-1$ contribute to acceleration determination. Hence, two transitive edges with distinct weights are generated connecting any joint pairing. An example is shown in Figure 5. Both blue edges f_1 and f_2 connect the same two detections, but have different edge weights, depending on the detections at frame $t+1$ that contribute to the acceleration weight.

4.1.3 Edge Constraints

The weighted partner graph defined in Section 4.1.1 is constructed from velocity and acceleration edges connecting partner joints over all configurations in a sequence. For the IPL to find a feasible partition of this graph, constraints have to be defined. In order to be able to intelligibly explain constraints without requiring an overly intricate notation, we will use the exemplary notation from Figure 5. This figure depicts an abstract sketch of two partner detections (large rectangles) in three consecutive frames (from left to right). Velocity edges e_i (black and red) connect all detections in a trellis graph. Additionally, two transitive acceleration edges f_1 and f_2 are also added in this sketch. For clarity, all other transitive edges have been omitted. We will use the edge

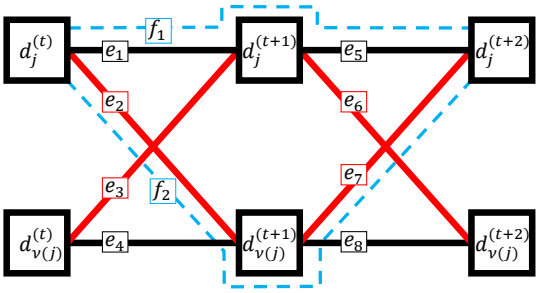

Figure 5. Velocity edges $\{e_1, ..., e_8\}$ between joint detections d of partner joints j and $\nu(j)$ in three consecutive frames $(t, t+1, t+2)$. Colors indicate predicted trajectories (black) and swap-edges (red). Transitive acceleration edges f_1 and f_2 (blue) connect $d_j^{(t)}$ with $d_j^{(t+2)}$. For clarity, only two out of eight transitive edges are shown.

notation from this Figure to exemplary carve out optimization constraints. The full constraint set holds all constraints for all frame triples and all partner joints. Note that e_i and f_i pose as indicator variables. If an edge is present, the variable equals 1 and 0 otherwise.

The first set of constraints model mutual exclusivity between velocity edges: A detection of joint j in frame t can not be connected to both joint j and joint $\nu(j)$ in frame $t+1$ at the same time. Using the nomenclature from Figure 5, this translates to:

$$\begin{aligned} e_1 + e_2 &\leq 1 \\ e3 + e4 &\leq 1 \end{aligned} \tag{7}$$

Both these constraints are violated if mutually exclusive edges appear together in the set. This set of constraints thereby guarantees that the ILP partitions the graph into separate joint trajectories.

The second set of constraints guarantees that the acceleration edges matching a selection of velocity edges are considered. If, for instance, the ILP considers edges e_1 and e_5 for the joint trajectory in one partition, then the transitive edge f_1 has to be added to the path. This constraint is modeled as

$$e_1 + e_5 - f_1 \leq 1 \tag{8}$$

These constraints model the necessary presence of transitive edges, but they do not prevent the arbitrary inclusion of other transitive edges, which automatically happens when maximizing the objective function.

This motivates the third and last set of constraints for mutual exclusivity of transitive edges. This constraint only allows two transitive edges per partner pairing and frame triple to be included in the final solution. It follows that

$$\sum_{i=1}^{8} f_i \leq 2. \tag{9}$$

All constraints allow for the ILP to yield a feasible partition of the graph and thereby a trajectory for each partner joint.

4.1.4 Edge Conflation

The edge set described in the previous paragraphs can be reduced to sets with smaller cardinality by capitalizing the fact that we only argue over partner joints, i.e., pairs of two joint types. Consider both wrist detections in frames t and $t+1$ in Figure 5. If we assume that detection $d_j^{(t)}$ is connected to $d_j^{(t+1)}$ via edge e_1, then it automatically follows that $d_{\nu(j)}^{(t)}$ is connected to $d_{\nu(j)}^{(t+1)}$ via e_4. Edges e_2 and e_3 would be deleted from the graph in this scenario. Then again, if e_2 and e_3 had larger probabilities, then they would connect the detections and e_1 and e_4 would be excluded from the solution. This binary characteristic of the problem allows for merging edges in the graph. For example, edges e_1 and e_4 can be merged into a new edge with its weight being set to the sum of both old edge weights. This edge now represents that no swap occurs from frame t to frame $t+1$. The same can be done by merging edges e_2 and e_3, yielding an edge representing a possible swap. This procedure can be implemented for all velocity and acceleration edges in the graph, reducing the number of weighted edges and associated constraints by half, leading to a more compact optimization problem which can be solved faster.

4.1.5 Optimization

In this work, objective 1 is solved using the Embedded Conic Solver (ECOS, [7]) and modeled with the convex optimization modeling language CVXPY [2]. The test sequences in the experimental section do not exceed 100 frames, for which the optimization of the kinematic graph is still tractable. For evaluating longer sequences, as necessary for video processing, the sequence of poses can be split into subsequences of length 100, overlapping with three frames. The ILP is solved for each subsequence, and the edges for the first frames are fixed to the solution obtained from the preceding subsequence.

4.2. Locally Robust Joint Trajectory Regression

While the focus of this work lies on the optimization of joint partner graphs for finding and rectifying joint transposition, we do not want to completely neglect outlier and joint residual variance. To that, we utilize the algorithm for locally robust regression and data-dependent filtering as proposed in [29]. They propose modifications to the LOWESS [6] algorithm for performing locally weighted regression by minimizing the L1-norm of residuals for strode and weighted subwindows of joint time series. We found this procedure to work decently well for smoothing our joint trajectories and interpolating outliers in the signal.

5. Experiments

	free	back	fly	breast
# sequences	24	32	26	28
# pose annotations	1883	2665	2100	2281

Table 1. Number of pose configurations per style.

Competitive swimming covers four different swimming styles: breaststroke (breast), butterfly (fly), freestyle (free) and backstroke (back). The first two are termed *symmetrical* styles, because both halves of the body perform the same motion at all times. The latter two are denoted *anti-symmetrical* as the left half of the body performs a motion that is mirrored approximately half a cycle later by the right half of the body and vice versa. If viewed from the side, one body half in symmetric styles is mostly indistinguishable due to self-occlusion, hence joint swaps appear to a lesser extent. On the other side, anti-symmetrical styles are affected by joint swaps and outliers. This may change with camera perspective. However, we only consider a side view of athletes in this work.

Dataset. Our dataset is comprised of swimming channel footage. All sequences and videos depict swimmers of different age, stature, gender and body size in two different swimming channels. The athletes are filmed from a side view through a glass wall, hence the whole body - above and below the water surface - is always visible. We distinguish between two overlap-free datasets in our experiments. The first set contains 104 fully annotated *sequences of human poses* with 50 to 100 consecutive images each, giving us a total of 8532 annotated video frames. Table 1 summarizes the dataset size and the number of annotations available for our experiments.

Metrics. For the quantitative evaluation of pose estimates, we apply the Percentage of Correct Keypoints (PCK, [21]) measure. PCK counts a joint as correctly localized if the Euclidean distance to the ground truth annotation does not exceed a fixed fraction α of the upper body size, which is defined by the Euclidean distance between right shoulder and left hip. Commonly, thresholds $\alpha = 0.1$ $(PCK@0.1)$ and $\alpha = 0.2$ $(PCK@0.2)$ are evaluated for comparing the performance of pose estimation systems.

We additionally report the root mean square (RMS) errors for prediction residuals to give a better understanding of how different improvements translate to a decreasing pixel location error. In order to compare different swimmer sizes, RMS values are always given for size normalized athletes, where normalization is performed as described in Section 4.1.2 with a reference upper body size $s_{ref} = 100$ pixels.

Joint Localization. We compare two state-of-the-art pose estimation systems for joint localization: Convolutional Pose Machines [26] with three refinement stages and MaskRCNN [13] build on a ResNet-101 backbone. The CPM model is initially pre-trained on the Leeds-Sports-Pose dataset [17], the Mask R-CNN on the MSCOCO dataset [18]. As we wish to compare the PCK of the rectification pipeline with the baselines but only have a small, fully annotated dataset of motion sequences available, we have to evaluate the pipeline on our training data. Therefore, we refine both the CPM and Mask R-CNN on each swimming style by fine-tuning it on our fully annotated sequences using a *3-fold cross validation*. Hereto, the data is split into three partitions. One partition is kept as the test set while the model is refined using the remaining partitions for training and validation. Thereby, we obtain pose estimates for each frame in our fully annotated pose sequences.

	free	back	fly	breast
Zecha et al. [29]	10.74	11.34	5.78	**6.56**
Mask R-CNN base	31.84	34.77	40.70	26.10
Mask R-CNN swap	28.91	32.02	-	-
Mask R-CNN rect	19.15	20.18	33.31	16.49
CPM base	14.81	17.90	7.07	6.69
CPM swap	12.08	11.76	-	-
CPM rect	**9.13**	**9.55**	**5.77**	6.65

Table 2. RMS values for Euclidean distances between prediction and ground truth for all joints.

Baselines. The PCK on the initial estimates for both networks serves as the baseline and is depicted in Figure 6 (dashed). We find that there is a difference in pose estimation network performance, with CPM coming out on top of Mask R-CNN. For the CPM, symmetrical swimming styles already work very well: the PCK@0.1 is beyond 90%; almost all joints (>99.0%) are predicted within 0.2 of the upper body size. The anti-symmetrical swimming styles are more difficult to predict precisely, with a PCK@0.1 of 78.7% for freestyle and 82% for backstroke. Both hardly exceed 95% PCK@0.2.

Full Rectification. For both tested network structures, the whole optimization pipeline leads to consistent PCK gain for all swim styles (continuous lines in Figure 6). The Mask R-CNN shows surprising gains for freestyle swimmers (+4.8% PCK@0.2), but also for breaststroke and butterfly swimming, especially compared to the marginal CPM gains for these styles. This can be attributed to larger volatility in Mask R-CNN predictions. We found that between consecutive images with only very little change in content, the joint predictions often jump unpredictably around the ground truth. This also becomes evident when comparing RMS values in Table 2, where the largest RMS loss is caused by the second rectification stage. Nevertheless, the localization gain is consistent for all swimming styles and both networks.

Partner Graph Ablation. To get a better understanding of the importance of partner graph optimization, we per-

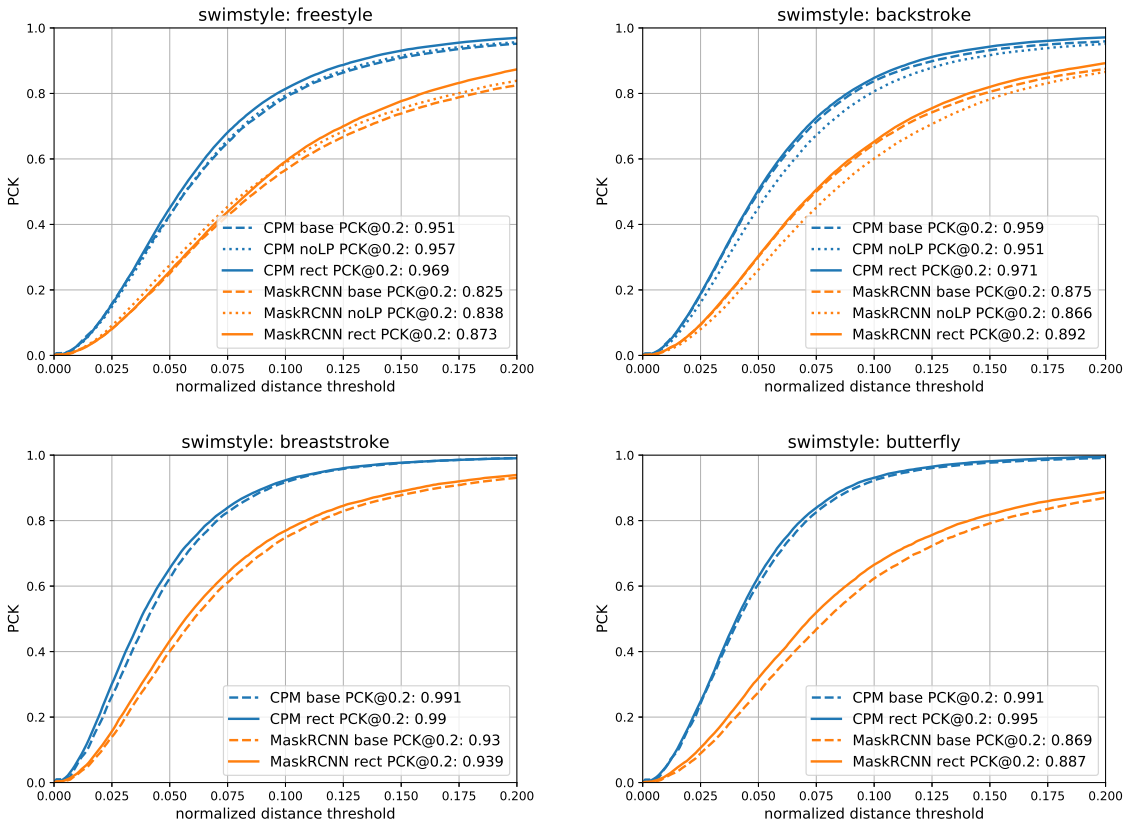

Figure 6. Percentage of Correct Keypoints for all four major swimming styles. Mask R-CNN (orange) is compared to CPM (blue). Dashed lined are raw pose prediction baselines, continuous lines depict the score after rectification. Dotted lines represent robust trajectory approximation without partner graph optimization.

form a simple ablation study by applying robust joint regression from Section 4.2 without optimizing the partner graph from Section 4.1.1. The results are depicted as dotted lines in Figure 6. For freestyle swimmers, the resulting PCK of $95.7\%(+0.5\%)$ for the CPM and $83.8\%(+1.3\%)$ for Mask R-CNN is slightly better than the baselines but worse than the results of the full rectification pipeline. For backstroke swimmers, the PCK drops to $95.1\%(-0.6\%)$ for the CPM and $86.6\%(-0.9\%)$. We conclude that partner graph optimization is not only beneficial to improving pose estimates, it can also be necessary to avoid poor rectification results.

Stage-wise Joint RMS. We evaluate each step in our pose rectification pipeline separately.Again, due to the small database of pose sequences, we apply a leave-one-out training scheme. One sequence is kept for testing, while the pipeline is trained on the remaining sequences. The joint transposition ILP and spatiotemporal regression scheme are applied to all swimming styles, although model specific parameters are optimized for each style individually. These include KDEs and number of clusters for directional offset correction. Table 2 compares root mean square values of squared Euclidean distances between prediction and ground truth. Note that the RMS was measured for size normalized poses as described in Section 4.1.1 with an reference upper body size of $s_{ref} = 100$. The RMS consistently improves for all swimming styles. Compared to [29], RMS values for all styles but breaststroke improved.

6. Conclusion

We addressed the problem of false athlete pose predictions from neural networks in aquatic training scenarios. An ILP optimization problem for joint partner swap identification and rectification was proposed. We can show experimentally that this approach eliminates a considerable amount of errors, leading to consistent performance gains. As we found several very specific mixtures of error sequences that could not be solved with the proposed methodology, future efforts will be directed towards finding more general formulations of the proposed solution strategy that additionally address remaining error mixtures.

References

[1] Ijaz Akhter, Tomas Simon, Sohaib Khan, Iain Matthews, and Yaser Sheikh. Bilinear spatiotemporal basis models. 31(2):17:1–17:12. 2

[2] Steven Diamond Akshay Agrawal, Robin Verschueren and Stephen Boyd. A rewriting system for convex optimization problems. 5(1):42–60. 6

[3] Andreas Aristidou and Joan Lasenby. Real-time marker prediction and CoR estimation in optical motion capture. 29(1):7–26. 2

[4] V. Belagiannis and A. Zisserman. Recurrent human pose estimation. In *2017 12th IEEE International Conference on Automatic Face Gesture Recognition (FG 2017)*, pages 468–475. 2

[5] Yu Chen, Chunhua Shen, Xiu-Shen Wei, Lingqiao Liu, and Jian Yang. Adversarial posenet: A structure-aware convolutional network for human pose estimation. pages 1221–1230. 2

[6] WS Cleveland. Lowess: A program for smoothing scatterplots by robust locally weighted regression. 35:54. 6

[7] A. Domahidi, E. Chu, and S. Boyd. ECOS: An SOCP solver for embedded systems. In *European Control Conference (ECC)*, pages 3071–3076. 6

[8] Moritz Einfalt, Dan Zecha, and Rainer Lienhart. Activity-conditioned continuous human pose estimation for performance analysis of athletes using the example of swimming. In *2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018, Lake Tahoe, NV, USA, March 12-15, 2018*, pages 446–455. 2

[9] S. R. Fanello, C. Keskin, P. Kohli, S. Izadi, J. Shotton, A. Criminisi, U. Pattacini, and T. Paek. Filter forests for learning data-dependent convolutional kernels. In *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 00, pages 1709–1716. 2

[10] Mykyta Fastovets, Jean-Yves Guillemaut, and Adrian Hilton. Athlete pose estimation from monocular tv sports footage. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 3

[11] Rohit Girdhar, Georgia Gkioxari, Lorenzo Torresani, Manohar Paluri, and Du Tran. Detect-and-track: Efficient pose estimation in videos. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 350–359. 2

[12] Georgia Gkioxari, Alexander Toshev, and Navdeep Jaitly. Chained predictions using convolutional neural networks. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *ECCV (4)*, volume 9908 of *Lecture Notes in Computer Science*, pages 728–743. Springer. 2

[13] Kaiming He, Georgia Gkioxari, Piotr Dollr, and Ross Girshick. Mask R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*. 1, 2, 3, 4, 7

[14] Jihye Hwang, Sungheon Park, and Nojun Kwak. Athlete pose estimation by a global-local network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 3

[15] Umar Iqbal, Martin Garbade, and Juergen Gall. Pose for action - action for pose. 2

[16] Umar Iqbal, Anton Milan, and Juergen Gall. Posetrack: Joint multi-person pose estimation and tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2

[17] Sam Johnson and Mark Everingham. Learning effective human pose estimation from inaccurate annotation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 7

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollar, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision – ECCV 2014*, pages 740–755. Springer International Publishing. 7

[19] Utkarsh Mall, G. Roshan Lal, Siddhartha Chaudhuri, and Parag Chaudhuri. A deep recurrent framework for cleaning motion capture data. abs/1712.03380. 2

[20] Tomas Pfister, James Charles, and Andrew Zisserman. Flowing convnets for human pose estimation in videos. In *International Conference on Computer Vision (ICCV)*. 2

[21] Benjamin Sapp and Ben Taskar. MODEC: multimodal decomposable models for human pose estimation. In *CVPR*, pages 3674–3681. IEEE Computer Society. 7

[22] Jie Song, Limin Wang, Luc Van Gool, and Otmar Hilliges. Thin-slicing network: A deep structured model for pose estimation in videos. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2

[23] Wei Tang, Pei Yu, and Ying Wu. Deeply learned compositional models for human pose estimation. In *The European Conference on Computer Vision (ECCV)*. 2

[24] Graham W. Taylor, Geoffrey E Hinton, and Sam T. Roweis. Modeling human motion using binary latent variables. In B. Schlkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1345–1352. MIT Press. 2

[25] Alexander Toshev and Christian Szegedy. DeepPose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1653–1660. 2

[26] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *CVPR*. 1, 2, 3, 4, 7

[27] Wei Yang, Shuang Li, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Learning feature pyramids for human pose estimation. In *arXiv preprint arXiv:1708.01101*. 2

[28] Wei Yang, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. End-to-end learning of deformable mixture of parts and deep convolutional neural networks for human pose estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2

[29] Dan Zecha, Moritz Einfalt, Christian Eggert, and Rainer Lienhart. Kinematic pose rectification for performance analysis and retrieval in sports. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 3, 6, 7, 8

[30] Hong Zhang, Hao Ouyang, Shu Liu, Xiaojuan Qi, Xiaoyong Shen, Ruigang Yang, and Jiaya Jia. Human pose estimation with spatial contextual information. abs/1901.01760. 2