Vision, Modeling, and Visualization (2023)
T. Grosch and M. Guthe (Eds.)

On the Beat
Supplemental Material

Malte Menzel [1], Jan-Philipp Tauscher[1], and Marcus Magnor[1]

[1] Institut für Computergraphik, TU Braunschweig, Germany
malte.menzel@tu-bs.de, {tauscher, magnor}@cg.cs.tu-bs.de

This supplemental material provides additional context for different aspects of our paper. It first explains additional concepts to refine the understanding of our work, it then shows additional technical details that were not presented in our paper and finally gives the full results of our tests which were summarised in our paper.

1. Additional Concepts

This section further explains different concepts needed for our work.

1.1. Kernel Smoothing

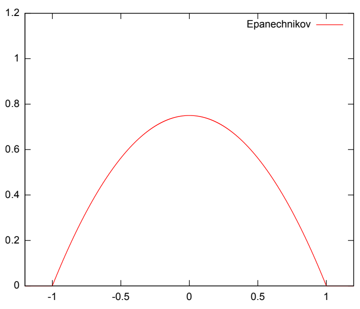


Figure 1: Plot of Epanechnikovs kernel (by Brian Amberg under Creative Commons Attribution-Share Alike 3.0 Unported).

Epanechnikovs kernel defines the weight K of each neighbour as

$$K(u) = \frac{3}{4}(1-u^2) \tag{1}$$

with $|u| <= 1$ being the normalised position of the neighbour in the sliding window. A visual representation of the weighting produced by Epanechnikovs kernel can be seen in Fig. 1. As shown in this figure, the actual frame itself receives the highest weighting, and the weighting decreases for each further frame. It further actually reaches a weighting of 0 at its edges unlike similar kernels such as Gaussian implementations, which is useful, since we are only interested in closely neighboring values.

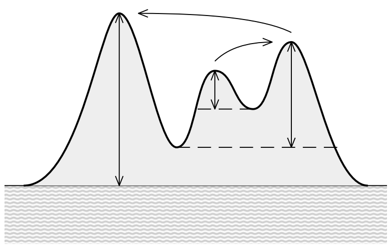


Figure 2: Example of the prominence of three peaks (by Cmglee under Creative Commons Attribution-Share Alike 3.0 Unported).

1.2. Prominence

Fig. 2 shows a visual example of prominence. The prominence of the middle peak, for example, is defined as the distance downwards to get to the valley to its right, as this valley leads to a peak of greater value. If no greater peak exists, as with the left of the three extremes, the prominence will be the distance from the peak to the lowest global value.

2. Additional Technical Details

This section includes additional technical details of our work.

2.1. Removing False Positives

Although the OpenPose development team describes the *BODY_25B* model as a version with reduced false positives, we still regularly encountered instances of OpenPose detecting persons where there actually weren't any persons. Examples of this include lamps, chairs or sometimes even in empty space. However, OpenPose has an optional *number_max_people_count* argument that can significantly help remove false positives. This argument should describe the maximum amount of people in the scene at any point in the video. In our approach, the user can enter this value manually if they know how many dancers are in a video they are analysing. If they do not enter a number, we detect it automatically. First, we consider OpenPose's confidence statistic. With each detected keypoint's x- and y-position, OpenPose also returns an internal confidence stat, describing how likely the returned

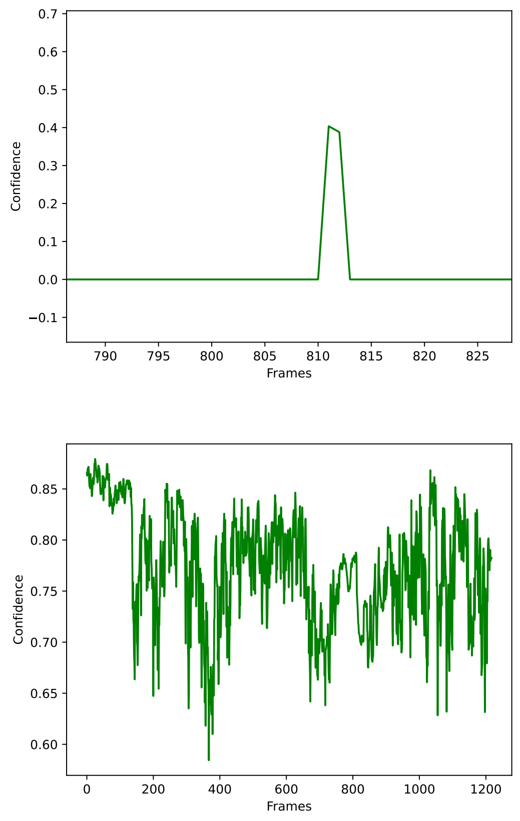


Figure 3: Plots of the average confidence of two detected persons.

keypoint belongs to a real person. This value ranges between 0.0 and 1.0, with higher values indicating higher confidence. To filter out persons, we average the confidence of all keypoints over the entire duration that the person was active. Through manual testing with about 20 sample videos, we found that most actual persons average a confidence score of about 0.7. Most false positives are much lower, never reaching above 0.5 in our testing. Two typical confidence value plots can be seen in Fig. 3. The first image shows the confidence values of a manually confirmed false positive. In contrast, the second image shows the confidence of a correctly detected person. Our work uses a minimum average confidence value of 0.6 to filter out false positives.

Second, we consider the time a person is active for. This is also a useful criterion, as most false positives are only detected for a few frames. When looking at true positives, though, OpenPose manages to detect them consistently for the entire duration they are in the scene. Because of this, we require a minimum active time of at least one second, as everything existing for less than this time span is not sensible to analyse. Fig. 3 shows the difference between the confidence values of an actual dancer and a false positive.

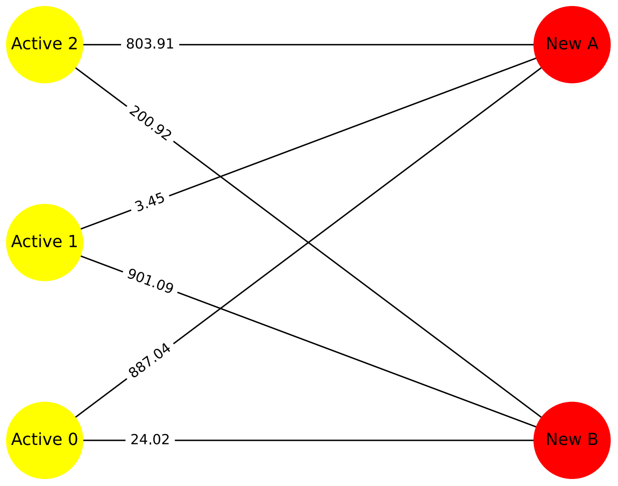


Figure 4: A graph display of matching with different numbers of persons in each frame.

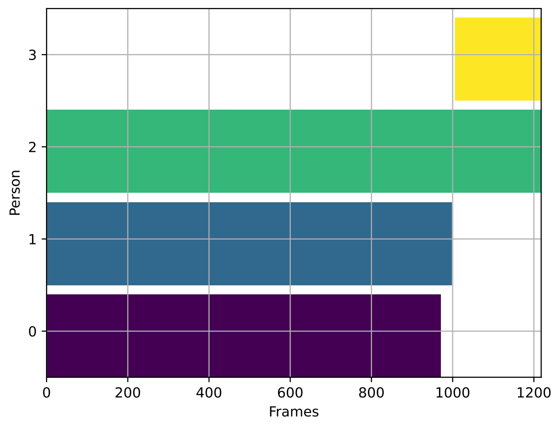


Figure 5: Example timing windows for each person in a typical test video.

2.2. Tracking Active Persons / Uneven Matching

Regarding tracking, we also must remember that sometimes the number of detected persons differs in two consecutive video frames, for example when a new person enters the recorded area or is occluded by an object. In this case, we still perform our matching algorithm but now from the side with fewer participants. Therefore, if a new person enters the frame, we create an additional active person object for the new person that did not receive a partner. Conversely, we do the opposite when a person leaves the recorded area. All active persons who received a partner continue to exist for the next frame, whilst those who did not will be closed. Fig. 4 shows an uneven matching.

Fig. 5 shows the timing windows during which each person is active for a typical test video from our database. The video starts with three persons being visible. Around frame 1000, two dancers exit the video and are no longer detected. Lastly, a couple of frames later, another dancer enters the scene, and now two people are once again active at the same time.

2.3. Average Person Height and Width

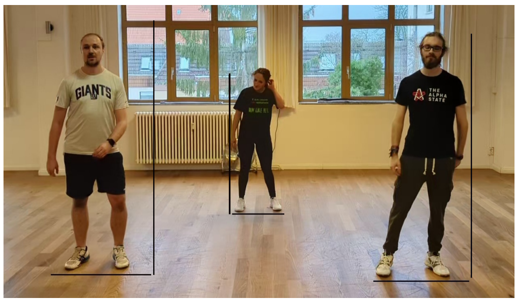


Figure 6: Average height and width of each person visualised.

When calculating the prominences needed to filter actual movement changes from noise, the size of a person matters. If the person is smaller, their movement changes will include smaller extremes since they do not move as much as larger persons. Second, the correct prominence value depends on the distance at which a person is standing away from the camera. Suppose two persons of the same size perform the same dance, one being further away from the camera. In that case, this person will have smaller prominence values for their extremes, as their entire movement will take up fewer pixels than their counterpart's. Fig. 6 shows a render of the average size and height of each dancer.

2.4. Audio and Video Correlation

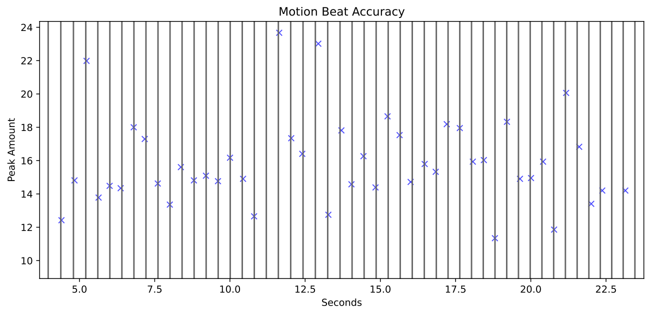


Figure 7: A typical video's motion beats and audio beats aligned temporally.

An example of both motion beats and audio beats in correlation can be seen in Fig. 7. Here, the audio beats are displayed as vertical lines. The motion beats are displayed as 'x' symbols, with their y-position indicating how many keypoints were peaking during each motion beat.

2.5. Display to the User

For each dancer, we calculate an accuracy score for every detected motion beat. We then present a diagram that shows the scores of each motion beat over time, the average scores and the standard deviation of these scores for each dancer. This allows the user to infer information on how synchronous each dancer was and which parts of the performance they might need to work on. Such a diagram can be seen in Fig. 8. The dancer here is performing well, with many motion beats being perfectly on the beat or close to it, and the rest is within acceptable margins.

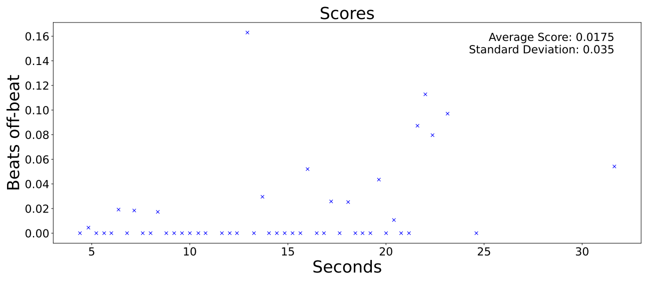


Figure 8: Display of typical scores of a well-performing dancer.

3. Detailed Results

This section gives more details on the results already presented in the main paper.

3.1. Component Test

This section provides the more detailed results of our component test that was briefly presented in the main paper.

Removal of False Positives We begin by testing the removal of false positives in our test videos, by manually checking the rendered OpenPose output. Here, we find no false positives in any of the 22 videos.

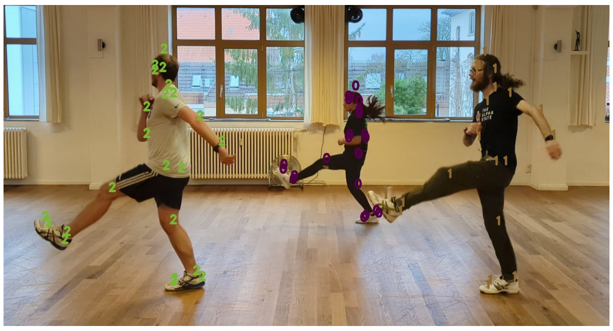
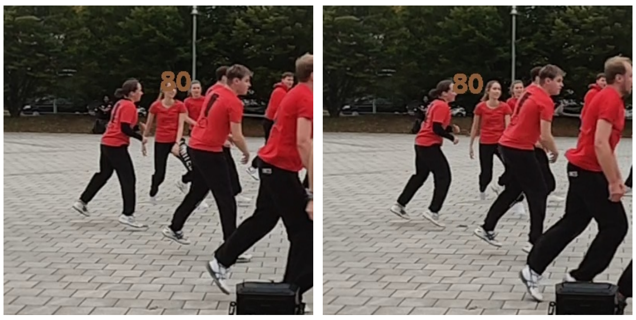


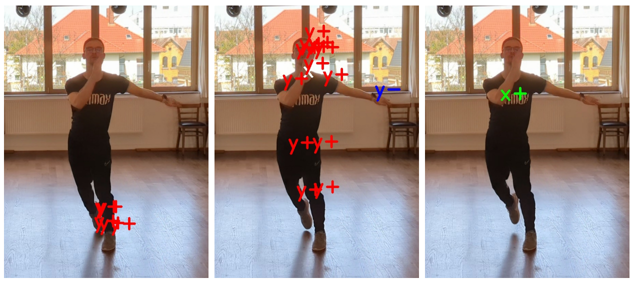
Figure 9: A video frame with the IDs of every dancer rendered as an overlay.

Tracking To test our tracking algorithm, we render a new video for each input video, with the IDs of every person object overlayed on every detected keypoint of that person. Our algorithm keeps a record of each detected person that we call a *person object*. An example frame from one of these videos can be seen in Fig. 9. We then manually check our tracking algorithm for correct behaviour on each video. Of the 22 videos, 20 show no problems in tracking. Two videos, however, share an uncommon problem, as OpenPose struggled to detect every dancer in every frame consistently. If OpenPose stops detecting one person that was detected during previous frames and starts detecting an entirely new person in the same frame, the number of detected persons is the same for both frames. As such, our algorithm does not detect that it needs to close a person object and start a new one. Instead, it performs the standard matching algorithm. With the previous frame containing a person that is no longer there and the new frame containing a new person at a possibly entirely different location, this can lead to the wrong persons being matched with each other. This results in one person object suddenly representing two actual dancers, as it is not closed correctly and instead reassigned to another dancer entirely. An example of this happening can be seen in Fig. 10. As this however is only a problem when OpenPose malfunctions, we do not attribute this as an error of our tracking solution.



(a) Frame 1889 of the video (b) Frame 1890 of the video

Figure 10: Two frames of the large-scale test video showcase how a person object can switch dancers.



(a) Frame 525 (b) Frame 526 (c) Frame 527

Figure 11: Three video frames showing the moment a dancer lands.

Motion Beat Detection To test our motion beat detection, we generate a new video for each test video, where we mark every keypoint currently peaking each frame. We also specify if a keypoint is peaking in the positive or negative direction and in which axis. This example can be seen in Fig. 11. For each of the 22 videos, we manually check if the peaks are at the correct times. For example, y-positive peaks should be displayed when a keypoint reaches its lowest point. Also, any small movements that are less than $\frac{1}{30}$th of the dancer's size should be filtered out by our prominence filtering. This works correctly for all videos, except in two instances where the tracking fails, as mentioned above. When our tracking algorithm moves a person's object from one dancer to another, it interprets it as a large movement. If this movement is in the correct direction, our algorithm can mark it as a motion change. These are the only times we noticed a failure of our motion beat detection system. As these are a direct result of the tracking system's behaviour, we do not consider them as a failure of our motion beat detection but rather as a consequence of our tracking system.

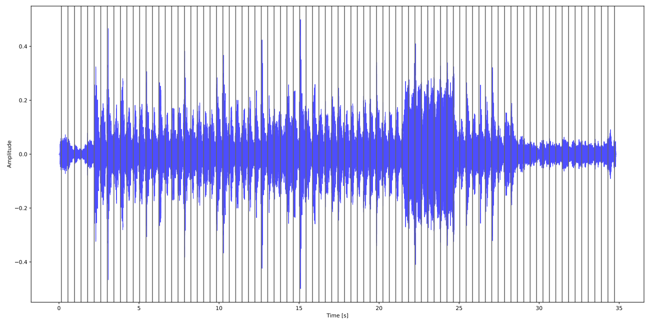


Figure 12: Render of the waveform of a song and the resulting audio beats detected by AUFTAKT V4.

Audio Processing To test our audio processing, we render the waveform of the sound of each video, overlayed with the timestamps of each audio beat detected by AUFTAKT V4. Such a visualisation can be seen in Fig. 12. Each vertical line represents an audio beat detected by AUFTAKT V4. As Jumpstyle music features prominent kick drums on most beats, it is possible to correlate the location of audio beats with the location of high amplitudes in the waveform. We also create a program that, when given the .wav file as input, replaces each location of a detected audio beat with 20 milliseconds of a sine wave. This program allows us to check AUFTAKT V4s accuracy by listening to the audible clicks that should appear on each beat. Out of the 22 videos, AUFTAKT V4 correctly detects every single audio beat in 18 of them. Four videos return wrong audio beats when processed by AUFTAKT V4.

- The first video is 18 seconds long and features a song with a tempo of 150 BPM. Here, AUFTAKT V4 finds the correct rhythm and audio beat locations for most of the song. However, the first four seconds of the song are interpreted incorrectly at a lower tempo. Then, the kick drum sets in and AUFTAKT V4 immediately finds the correct tempo for the rest of the video.
- The second video is 22 seconds long and includes a song at 160 BPM. Once again, AUFTAKT V4 starts at the wrong tempo. Interestingly, the buildup of the song ending at ∼8.5 seconds and the kick drum starting again do not fix the problem. AUFTAKT V4 takes another 3.5 seconds and, at second 12, finally finds the correct tempo, which lasts until the end of the video.
- The third video lasts 22 seconds, and the song playing in the background is at 150 BPM. Unlike with the first two videos, AUFTAKT V4 detects the right tempo and beat locations at the start of the video, which includes the buildup part of the song. It holds this for the first six seconds of the following drop part

until it then suddenly drifts off-beat. Oddly, there is no change in music here. The rest of the song repeats the current phrase of the drop and sounds almost identical to the start of the drop that was identified correctly.

- The last video is 22 seconds long and features the same song playing in the third video. However, this time, a few seconds less of buildup are included. Instead, more of the drop is playing. AUFTAKT V4 once again starts off correctly during the buildup, but this time immediately begins to drift off-beat as the drop comes in at ∼5 seconds. It continues to drift until it suddenly latches onto the beat grid at 19 seconds, as the melody repeats a second time.

Overall, AUFTAKT V4 works perfectly for almost all of our test videos, with one of the incorrectly analysed ones even becoming usable after a few seconds.

3.2. Details on Off-Beat Video Results

Our data set features five off-beat videos. The results for every off-beat case are detailed here.

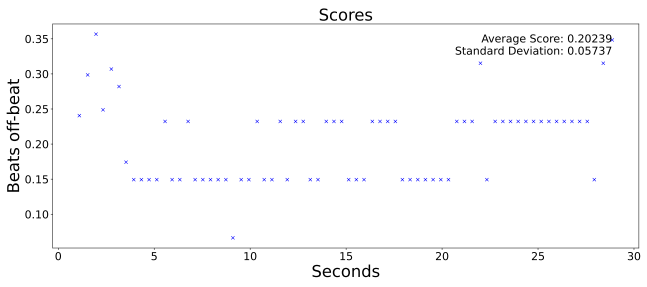


Figure 13: Scores of a dancer performing off-beat.

Delay We begin with the simplest off-beat case: a performer being constantly off-beat. In other words, while the performer is able to hold the correct tempo, they dance with a permanent delay. Every motion beat happens a short while too late, so the entire performance is off-beat. Our data set contains one case that was labelled as constantly off-beat. The results of our method for this video can be seen in Fig. 13.

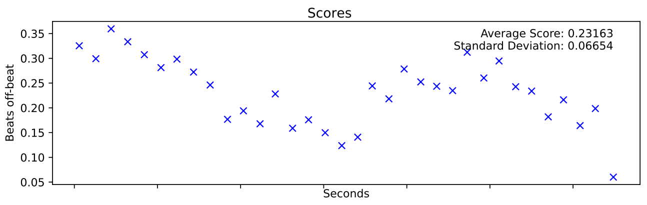


Figure 14: Scores of a simulation of permanent delay.

To further test this interpretation of our results, we created a second video that artificially matches this case. To do so, we took the recording of dancer 7 from the on-beat videos and edited the video stream to start with a delay of about 0.2 seconds after the audio stream. This introduces a constant delay, and each motion beat should now be scored worse than the previous good ones from the original performance. The results of analysing this video with our method can be seen in Fig. 14.

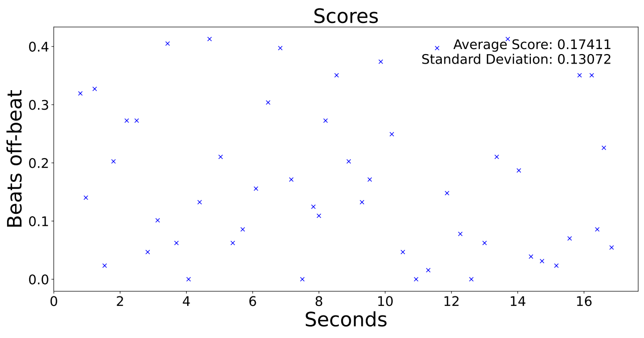


Figure 15: Scores of a dancer performing at the wrong tempo.

Wrong Tempo Next in our data set is a video where one dancer performs at the wrong tempo. The dancer is able to hold the incorrect tempo throughout the entire performance. This leads to an obviously asynchronous performance that is even recognised by amateurs. Fig. 15 shows the results of our method for analysing this video.

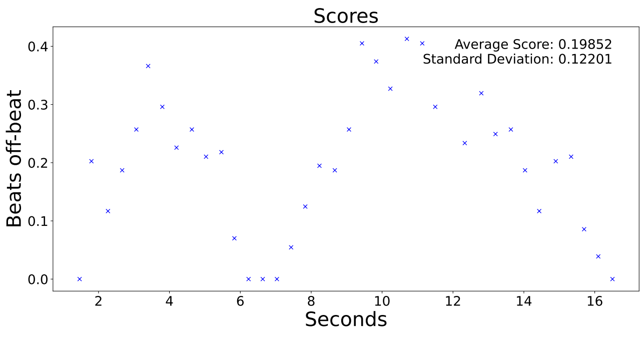


Figure 16: Scores of a dancer regularly losing the beat of the music.

Short Term Missteps Another interesting video contained in our data set is the following one, where the dancer performs both on- and off-beat as they start out dancing synchronously to the music. However, later, they lose the beat and slowly drift towards asynchrony. They then notice their mistake and manage to synchronise to the music again. This process then repeats a second time, ending with the dancer finishing the performance on-beat. Once again, the results of our analysis can be seen in Fig. 16.

No Dancing Further, our data set includes a video of a person not dancing at all. Whilst music is playing in the background, the person only moves around idly and makes no effort to dance. Nonetheless, this video is interesting to analyse, as random movements might still fulfil our criteria for motion beats. The scores of this person can be seen in Fig. 17.

References

[AD14] ALEXIADIS D. S., DARAS P.: Quaternionic signal processing techniques for automatic evaluation of dance performances from mocap data. *IEEE Trans. Multimedia* (2014), 1391–1406.

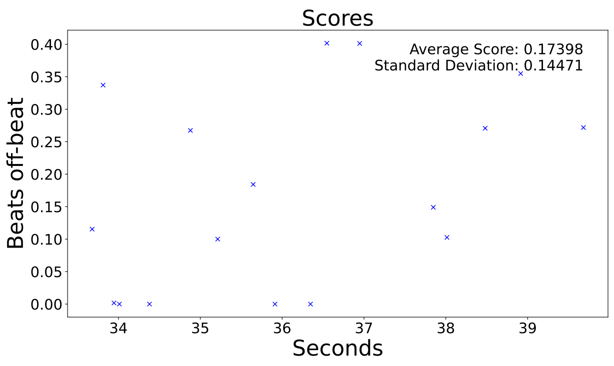


Figure 17: Scores of a person not dancing at all.

[AI16] Argüello C., Iregui M.: Exploring rhythmic patterns in dance movements by video analysis. In *DHM* (2016), pp. 123–131.

[BCMC*12] Bläsing B., Calvo-Merino B., Cross E. S., Jola C., Honisch J., Stevens C. J.: Neurocognitive control in dance perception and performance. *Acta psychol.* (2012), 300–308.

[BGR*20] Bazarevsky V., Grishchenko I., Raveendran K., Zhu T., Zhang F., Grundmann M.: Blazepose: On-device real-time body pose tracking. *arXiv preprint arXiv:2006.10204* (2020).

[BKCO18] Bellini R., Kleiman Y., Cohen-Or D.: Dance to the beat: Synchronizing motion to audio. *Computational Visual Media* (2018), 197–208.

[CHS*19] Cao Z., Hidalgo Martinez G., Simon T., Wei S., Sheikh Y. A.: Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Trans. Pattern Anal. Mach. Intell.* (2019).

[CRO01] Cross I.: Music, cognition, culture, and evolution. *Ann. N. Y. Acad. Sci.* (2001), 28–42.

[CT11] Chu W.-T., Tsai S.-Y.: Rhythm of motion extraction and rhythm-based cross-media alignment for dance videos. *IEEE Trans. Multimedia* (2011), 129–141.

[DA18] Davis A., Agrawala M.: Visual rhythm and beat. *ACM Trans. Graph.* (2018), 1–11.

[Dix01] Dixon S.: Automatic extraction of tempo and beat from expressive performances. *J. New Music Res.* (2001), 39–58.

[DP07] Davies M. E., Plumbley M. D.: Context-dependent beat tracking of musical audio. *IEEE/ACM Trans. Audio Speech Lang. Process.* (2007), 1009–1020.

[DWB09] Drobny D., Weiss M., Borchers J.: Saltate! a sensor-based system to support dance beginners. In *CHI EA* (2009), ACM, pp. 3943–3948.

[EAT*12] Essid S., Alexiadis D., Tournemenne R., Gowing M., Kelly P., Monaghan D., Daras P., Drémeau A., O'Connor N. E.: An advanced virtual dance performance evaluator. In *IEEE Int. Conf. Acoust. Speech Signal Process.* (2012), pp. 2269–2272.

[Ege12] Ege A.: Schnelle hüpfer im jumpstyle, 2012. max.de, accessed 22/08/2023.

[Ell07] Ellis D. P. W.: Beat tracking by dynamic programming. *J. New Music Res.* (2007), 51–60.

[Epa69] Epanechnikov V. A.: Non-parametric estimation of a multivariate probability density. *Theory of Probability & Its Applications* (1969), 153–158.

[FLT*22] Fang H.-S., Li J., Tang H., Xu C., Zhu H., Xiu Y., Li Y.-L., Lu C.: Alphapose: Whole-body regional multi-person pose estimation and tracking in real-time. *IEEE Trans. Pattern Anal. Mach. Intell.* (2022).

[GM94] Goto M., Muraoka Y.: A beat tracking system for acoustic signals of music. In *ACM MM* (1994), pp. 365–372.

[Gue06] Guedes C.: Extracting musically-relevant rhythmic information from dance movement by applying pitch tracking techniques to a video signal. In *Sound Music Computing* (2006).

[HM81] Herndon M., McLeod N.: *Music as culture*. Norwood, 1981.

[Hon12] Honing H.: Without it no music: beat induction as a fundamental musical trait. *Ann. N. Y. Acad. Sci.* (2012), 85–91.

[HTLC13] Ho C., Tsai W.-T., Lin K.-S., Chen H. H.: Extraction and alignment evaluation of motion beats for street dance. In *IEEE Int. Conf. Acoust. Speech Signal Process.* (2013), pp. 2429–2433.

[HTW12] Hadley L., Tidhar D., Woolhouse M.: Effects of observed music-gesture synchronicity on gaze and memory. In *ICMPC* (2012), pp. 384–388.

[Ive08] Ivers B.: What Is It? Jumpstyle, 2008. https://xlr8r.com/features/what-is-it-jumpstyle, accessed 22/08/2023.

[Jum] Jump It: *Jumpstyle Dance Formation, Braunschweig, Germany*. http://www.youtube.com/@formationjumpit2863.

[jum08] jumpisthestyle.com: Over jumpen (about jump), 2008. http://jumpisthestyle.com/jumpstyle/overjump, accessed 22/08/2023.

[Kar80] Karp R. M.: An algorithm to solve the m × n assignment problem in expected time o(mn log n). *Networks* (1980), 143–152.

[KK18] Kim Y., Kim D.: Real-time dance evaluation by markerless human pose estimation. *Multimed. Tools Appl.* (2018), 31199–31220.

[LT16] Laraba S., Tilmanne J.: Dance performance evaluation using hidden markov models. *Comput. Animat. Virtual Worlds* (2016), 321–329.

[QD20] Q-Dance: What is hardstyle music?, 2020. https://www.q-dance.com/en/static/hardstyle, accessed 22/08/2023.

[Rep06] Repp B. H.: Musical synchronization. *Music, motor control, and the brain* (2006), 55–76.

[RJK*16] Reason M., Jola C., Kay R., Reynolds D., Kauppi J.-P., Grobras M.-H., Tohka J., Pollick F. E.: Spectators' aesthetic experience of sound and movement in dance performance: A transdisciplinary investigation. *Psychol. Aesthet. Creat. Arts* (2016), 42.

[TSS18] Tongpaeng Y., Sribunthankul P., Sureephong P.: Evaluating real-time thai dance using thai dance training tool. In *ECTI DAMT* (2018), pp. 185–189.

[WJ94] Wand M. P., Jones M. C.: *Kernel smoothing*. CRC press, 1994.

[WYB*14] Wei Y., Yan H., Bie R., Wang S. t. m. d., Sun L.: Performance monitoring and evaluation in dance teaching with mobile sensing technology. *Pers. Ubiquitous Comput.* (2014), 1929–1939.

[zpl23] zplane.development: *AUFTAKT V4*. zplane.development GmbH & Co KG Grunewaldstr. 83 10823 Berlin, Germany, 2023. https://licensing.zplane.de/technology#auftakt.

[ZXY21] Zhou Z., Xu A., Yatani K.: Syncup: Vision-based practice support for synchronized dancing. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* (2021).