Creating Deep Learning-based Acrobatic Videos Using Imitation Videos

Jong In Choi[1] and Sang Hun Nam[2]*
[1] Department of Digital Media Design and Application, Seoul Women's University
Seoul, South Korea
[2] Department of Culture Technology, Changwon National University
Changwon-si, Gyeongsangnam-do, South Korea
[e-mail: funtech@swu.ac,kr, sanghunnam@changwon.ac.kr]
*Corresponding author: Sang Hun Nam

Received September 13, 2020; revised September 30, 2020; accepted October 30, 2020; published February 28, 2021

Abstract

This paper proposes an augmented reality technique to generate acrobatic scenes from hitting motion videos. After a user shoots a motion that mimics hitting an object with hands or feet, their pose is analyzed using motion tracking with deep learning to track hand or foot movement while hitting the object. Hitting position and time are then extracted to generate the object's moving trajectory using physics optimization and synchronized with the video. The proposed method can create videos for hitting objects with feet, e.g. soccer ball lifting; fists, e.g. tap ball, etc. and is suitable for augmented reality applications to include virtual objects.

Keywords: Augmented Reality, Artificial Intelligence, Pose Tracking, Motion Analysis, Physics Optimization

A preliminary version of this paper was presented at APIC-IST 2020, and was selected as an outstanding paper. This work was supported by a research grant from Seoul Women's University(2020-0181) and Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT)(No.2020-0-00238, Virtual Reality Force Feedback Controller Based User Biosignal Interface and Contents Control Technology).

http://doi.org/10.3837/tiis.2021.02.018 **ISSN : 1976-7277**

1. Introduction

Artificial intelligence has become an active topic for the fourth industrial revolution and has been applied to various fields, with tremendous development speed. Machine learning is a major thrust for artificial intelligence, being one of the most important emerging artificial intelligence technologies. Traditional computer graphics is gradually adopting these deep learning techniques, in particular for character animation. Techniques have been developed to automatically generate repetitive movements [1], with many video techniques being brought over to learning human movements [2]. Motion capture is an essential technology for character animation, supplying the required motion details, but remains computationally expensive and time consuming. Therefore, considerable effort has been focused to solve these shortcomings, and advanced motion capture technologies are starting to emerge [3, 4]. However, computational capabilities lag the performance from top-end motion capture equipment. On the other hand, some of the software developments have potential outside strictly motion capture applications, e.g. to compare expert motion to that for new or average users, or to quickly check practical motion against prototype motion data.

Consider two key base skills: lifting a soccer ball and tap ball. Lifting a soccer ball uses both feet to control the ball in the air without allowing it to drop to the ground. This is an essential skills for any soccer player but requires considerable time and effort to learn. Tap ball attaches a light rubber ball with a rubber band on the player's cap, and the player practices continuous punches with both fists. This improves a boxer's ability to see moving objects and develops their ability to hit objects accurately. However, this is also difficult skill for beginners to learn. This paper uses pose tracking and augmented reality to identify hitting motions, such as punches and kicks, from videos. Consequently, we propose a method to create acrobatic scenes, such as lifting a soccer ball or tap ball, from a beginner's video.

We used motion analysis from deep learning to build the proposed framework. User motion was analyzed in real time tracking the person's 2-dimensional (2D) pose from a video [5]. This is technique can extract skeleton structures for several people in a single video. Although it does not provide 3-dimensional (3D) location information, it is possible to precisely extract human skeleton structure from a 2D image. Thus, the proposed framework does not require 3D information because it creates an acrobatic scene by synchronizing a virtual object to a person in the video, without requiring precise 2D position or information. The hit point for the desired body part is extracted and object motion is subsequently generated and synchronized. Thus, we can quickly create acrobatic scenes for various objects. The proposed method is particularly useful for augmented reality, generating an acrobatic video by overlaying virtual objects on the original video.

2. Related Work

Estimating human pose from an image focuses on finding individual's body parts [6-9]. In particular, estimating each person's pose in a scene where several unrelated people appear simultaneously is a difficult problem.

① Unknown people in each image appear at random locations and sizes.

② It is difficult to identify each person's pose accurately due to complex spatial interference due to interactions between humans in contact with each other or occluding parts of other people's bodies.

③ Complexity tends to increase rapidly with increasing number of people, making it difficult to extract all poses in real time.

Current approaches [10-13] use detection algorithms to extract people, and perform pose estimation for each detected person. This top-down approach directly utilizes first person pose estimation techniques [14-17], but detection tends to fail rapid pose detection is required, particularly when people are close together. Execution time is proportional to the number of people since individual pose estimation is required for every identified person, and hence increased number of people increases computational cost.

Bottom-up approaches are attractive since the provide stability for fast detection and have the potential to separate real-time complexity from the number of people in the image. However, these approach do not use other body parts or other people's global context signals, and hence previous bottom-up techniques [18, 19] were less efficient because final parsing required time consuming global inference. Insafutdinov et al.[19] significantly improved real-time performance based on Pishchulin et al.'s approach [18], using a robust body part detection algorithm based on ResNet [20] and image dependent pairwise scores. However, the method limited the number of suggestions per site and required several minutes per image. Cao et al. [5] proposed a technique to quickly and accurately find skeleton structures for multiple people in a single image in real time. This paper quickly analyzes user poses in video frames using the Cao et al. technique, and subsequently generates acrobatic motion.

Various techniques have been applied previously to control rigid bodies [21, 22], smoke and water [23, 24] and deformable bodies [25, 26]. The proposed framework incorporates concepts from interactive interfaces to control rigid body movement [22], employing a few physical parameters to control passive rigid body simulation. Manually controlling body movement paths and character poses [27] can help to create characters that handle objects. Choi et al. [28, 29] proposed a method to generating object motions depending on character motion in 3D space. This paper adopts a similar concept, but uses motion capture data to provide accurate locations. Generally, locations extracted from images are an approximation and contain considerable noise. However, we create object motion using skeleton position rather than speed taken from the moving person's video.

Several previous studies have considered objects using physically based characters [30, 31]. Chemin et al. [30] created a juggling motion from a 2D physical character's upper body using reinforcement learning to generate various juggling motions. However, but since the motion was created reinforcement learning, it was somewhat different from actual human motion. Hong et al. [31] generated relatively realistic soccer ball dribbling motions using model predictive control (MPC). The physical based character was able to fine-tune ankle movement to send the ball in the correct direction. Merel et al. [32] applied vision-based reinforcement learning to physics-based characters to create a character's motion that can catch an arbitrary object and throw it at a target point. Eom et al. [33] created a physics-based character that can effectively catch a flying object by using the model predictive control framework that synthesizes the movements of the eyes and head. Starke et al. [34] proposed a effective method to generate motion in ball games handling an object by training local bones individually. Especially, it can generate basketball dribbling naturally and various connected motions. In contrast, the method proposed here automatically controls ankle movement using path control for the rigid body simulation to create an appropriate movement path depending on the position of the part handling the object in the video.

3. Proposed Method

3.1 Pose Tracking

User motion video quality is very important, since more similar user to the desired acrobatic motion will allow better synchronization with object movement and hence more natural appearing resulting video. Generated acrobatic scene quality can also be improved by practicing while watching acrobatic motion videos prior to shooting the user video. Therefore, we created lifting scenes for ball shaped objects using feet or thighs, and had users practice feet and thighs movements several times while watching the created lifting videos. Human body part locations were extracted from user video using Cao et al.'s technique [5] to track posture in real time for various people in a video using deep learning. We improved posture tracking performance by limiting the number of people to one and reducing video quality.

Fig. 1 provides an overview of the proposed method. **Fig. 1(a)** is the input user video (lifting a soccer ball while walking forward), **Fig. 1(c)** is the generated soccer ball lifting video, and **Fig. 1(b)** shows the analysis to calculate hitting information, and subsequently generate the object's moving trajectory. Analyzing user motion and synchronizing the object's movement trajectory to the user's video creates a scene lifting the soccer ball.

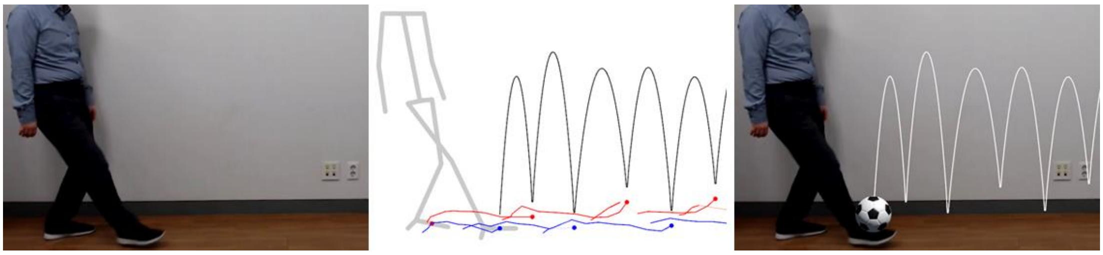

(a) (b) (c)

Fig. 1. Proposed method overview

3.2 Hit Spot Search

It is essential to generate hit spot information to subsequently generate an object's moving trajectory. There are two relevant information types regarding hit spots: location and time information. It is very important to control the applied force to successfully lift a soccer ball with your feet. We analyzed player videos (particularly lifting the soccer ball) while reducing foot speed to the minimum when hitting the ball. Players tended to pull their feet in the opposite direction as they were going to hit the soccer ball. Thus, this was used as a criterion for identifying object hitting points.

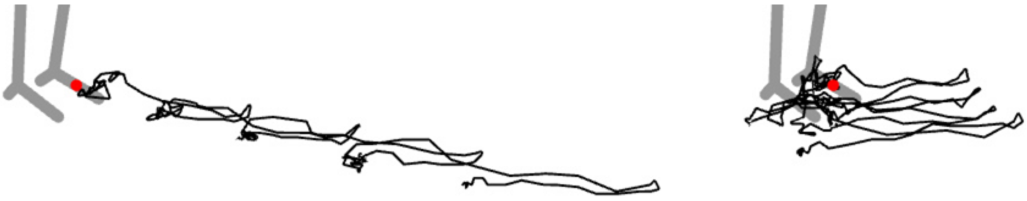

(a) (b)

Fig. 2. Original and relative movement path for the left foot

This study created not only the motion to lift an object while stationary, but also a lifting scene while moving. **Fig. 2** shows that after finding the movement trajectory for the body part hitting the object, the relative movement path was generated based on the skeleton joint that provided the reference for the hitting part. **Fig. 2(a)** is the original hitting part movement path, and **Fig. 2(b)** is the relative movement path calculated for a specific joint. For example, relative paths for left and right feet were generated based on the left and right hip joints, respectively. The red dot indicates the exact hitting part position, i.e., the instep, midway between toes and ankle. The gray skeleton is the user pose tracked from the video. In principle relative movement trajectory should around the origin, but we moved the origin to the initial position to compare with the original movement path.

Hit lines were extracted to find hit spot information, and the target hit direction was calculated and used to find hit lines for the hitting part. The optimal slope for the straight line passing through the points was calculated using all positions for the relative movement path as point cloud data to provide the reference hit direction,

$$\mathbf{y = ax + b}$$

$$\mathbf{a} = \frac{\sum_{i=1}^{n}(\boldsymbol{x_i} - \bar{\boldsymbol{x}})(\boldsymbol{y_i} - \bar{\boldsymbol{y}})}{\sum_{i=1}^{n}(\boldsymbol{x_i} - \bar{\boldsymbol{x}})^2}, \qquad \mathbf{b} = \bar{\boldsymbol{y}} - \mathbf{a}\bar{\boldsymbol{x}} \tag{1}$$

$$\bar{\boldsymbol{x}} = \frac{\sum_{i=1}^{n}\boldsymbol{x_i}}{\boldsymbol{n}}, \qquad \bar{\boldsymbol{y}} = \frac{\sum_{i=1}^{n}\boldsymbol{y_i}}{\boldsymbol{n}}$$

where **a** is the, **b** is the y-intercept, $\bar{\boldsymbol{x}}$ and $\bar{\boldsymbol{y}}$ are average ***x*** and ***y*** values, and ***n*** is the number of points. The reference hit direction is from closer to distant points relative to the hitting part's initial position.

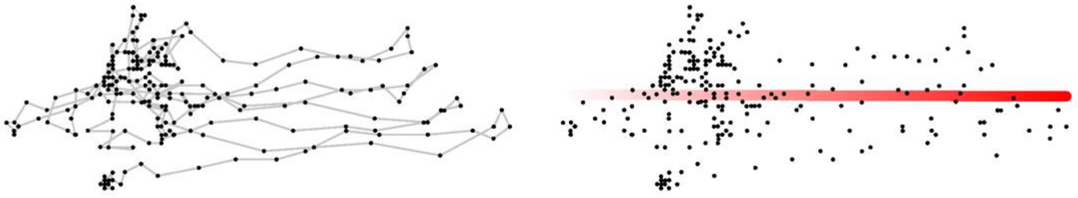

(a) (b)

Fig. 3. Calculating reference hit direction

Fig. 3 Shows the general approach to calculating the reference hit direction from relative movement paths (**Fig. 3(a)**) converted to point cloud data (**Fig. 3(b)**). Gray lines are relative movement paths, black dots hitting part location in each frame. The reference hit direction was calculated from the point cloud data and the thick red line segment is the standard hit direction, where the hitting part moves from the lighter to darker end.

The hit line connects points in the point cloud facing the same direction as the reference hit direction. From the initial position, the direction to each subsequent position is derived using the two previous points from the current position, which becomes the movement direction for the current position. This minimizes error effects due fundamental pose tracking issues. Although the movement direction could be taken as the direction from the current to next position, this will exacerbate noise effects. Thus, more accurate results can be obtained by using the previous and next positions.

The criteria for a hit line are that the angle difference between moving and reference hit direction for the current position should be within 60°, and there must be at least 3 consecutive points. These conditions filter most noise and provide a relatively accurate position. **Fig. 4(a)** shows extracted hit lines. Hitting position was the point where movement direction changed rapidly. However, it was not possible to completely remove noise (**Fig. 4(a)**), and hence additional methods are required. Remaining noise was characterized by shorter length than normal hit lines. **Fig. 4(b)** shows that removing lines shorter than 30% maximum length significantly reduces noise. Tighter controls, i.e., increasing the cutoff length, would further reduce noise, but cause problems with specifying different data. Therefore, 30% threshold was selected as a suitable trade-off with remaining noise removed using the time interval constraint for hitting time.

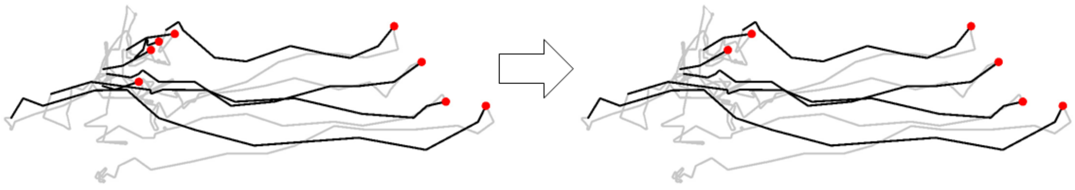

(a) (b)

Fig. 4. Extracting hit lines and removing noise

Table 1. Hitting information for the left foot

Frame	39	48	59	112	164	214
Length	12.29	12.01	54.44	75.22	94.33	113.45

Table 2. Hitting information for the right foot

Frame	29	86	138	189
Length	59.24	74.39	83.05	93.44

Table 1 and **Table 2** summarize extracted hit spot information for the left and right feet, respectively. The first and second left foot lengths (**Table 1**, red text) are relatively short, i.e., these are noise and can be cleanly removed with the time interval filter; whereas all right foot noise was removed by the threshold (**Table 2**). Increased left foot noise is due to the left foot being regularly occluded by the right foot since the video was taken from the user's right. The reverse would apply if the video were shot from the user's left side.

Fig. 5 shows hit lines calculated from left and right foot relative movement paths (red and blue lines, respectively), with corresponding blue and red dots showing hit spots. Gray curves are the relative movement paths for the hitting part.

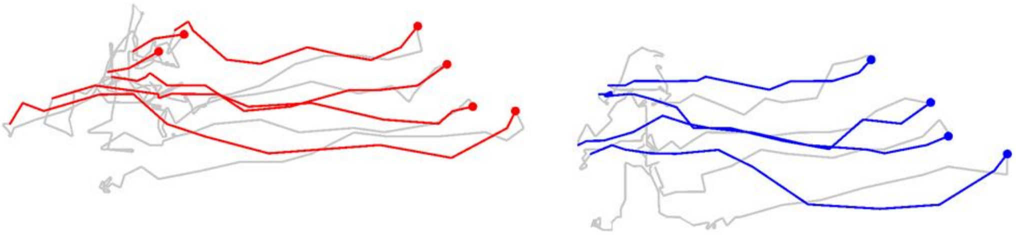

(a) (b)

Fig. 5. Hit lines for (a) left and (b) right feet

3.3 Object Synchroization

Generating accurate hitting information is essential to synchronize object motion with the video. As discussed above, the generated information includes noise and cannot be used immediately. Remaining noise can be removed using a suitable hitting time interval, e.g. it is difficult to lift a soccer ball more than every 0.5 seconds. Therefore, we can filter the data by checking if the interval between neighboring hit spots is less than 0.5 s, i.e., 15 frames, in which case one of the two hit cases should be considered as noise and the shorter hit line removed. However, if both feet are used then the interval between hitting points for one foot will be relatively large. Therefore, we need to combine hitting information generated by all hitting parts and then sort them in chronological order to more clearly identify noise.

Table 3 shows collects typical hitting information for both feet, sorted chronologically. Consider the first hit point pair in the table. The second hit (frame 39, left foot, red text) is only 10 frames from the first (frame 29, right foot, black text), which fails the criterion that hitting interval should be 15 frames or more, and hence one of the pair should be eliminated. Comparing hit line lengths, the first hit lines is longer than the second, and hence the second hit is regarded as noise and removed. Similarly, the second pair (frame 48, left foot, red text; and frame 59, left foot, black text) differ by 11 frames, and the first hit line is considerably shorter than the second. Thus, the first hit of the pair (frame 48) should be treated as noise and removed. All remaining hit pairs have larger hit intervals, and long hit lines, hence only valid hitting information remains.

Table 3. Hitting information for both feet

Foot	Right	Left	Left	Left	Right	Left	Right	Left	Right	Left
Frame	29	39	48	59	86	112	138	164	189	214
Length	59.24	12.29	12.01	54.44	74.39	75.22	83.05	94.33	93.44	113.45

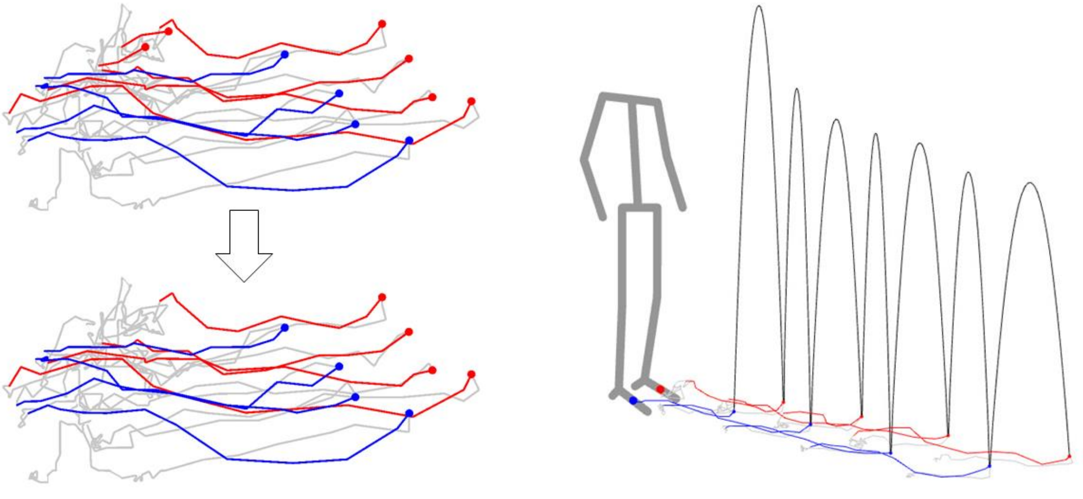

(a) (b)

Fig. 6. (a) Hit lines for both feet and (b) object movement trajectory

Fig. 6 shows the hit lines and object trajectories for the data in **Table 3**. **Fig. 6(a)** upper shows the original hit lines for both feet, and the lower image shows the same data after noise removal (i.e., removing short hit lines). Red and blue lines and dots represent left and right foot hit lines and spots, respectively. Gray lines represent overall movement path for both feet.

Since the exact hitting information is generated, the object trajectory can be created using Popovic et al.'s [22] method, an effective mechanism to create rigid movement trajectory through optimization. We regarded the object as an elastic body with an appropriate modulus to generate the object moving trajectory. **Fig. 6(b)** shows the resulting object trajectory corresponding to the noise filtered hit data. Lower curves are hit lines, shown as absolute rather than relative movement, where red and blue lines represent the left and right feet, respectively, with colored dots indicating hitting positions. The black curve represents the derived object trajectory. Since the real object is not a point or particle, we need to slightly adjust the hitting position with an appropriate offset, i.e., moving the object trajectory upwards by object radius, to provide a more natural synchronization with the video.

4. Experimental Results

To increase the generated acrobatic video realism, we adjusted physical parameters for the generated by the proposed method, based on video of an expert actually lifting the object, as shown in **Fig. 7**. The magenta circle is the ball position as calculated by the proposed method, which closely matches the actual position. **Fig. 8** shows a series of images captured every 0.1 seconds for an expert hitting a tap ball with both fists. Similarly, the (smaller) magenta circle is tap ball position as calculated by the proposed method. The calculated position is similar to the actual ball position, but differs slightly since the actual ball is quite light and hence affected by irregular factors, such as wind and air resistance. In contrast, the soccer ball (**Fig. 7**) was relatively heavy, hence less affected by wind or air resistance, providing almost identical actual and calculated positions.

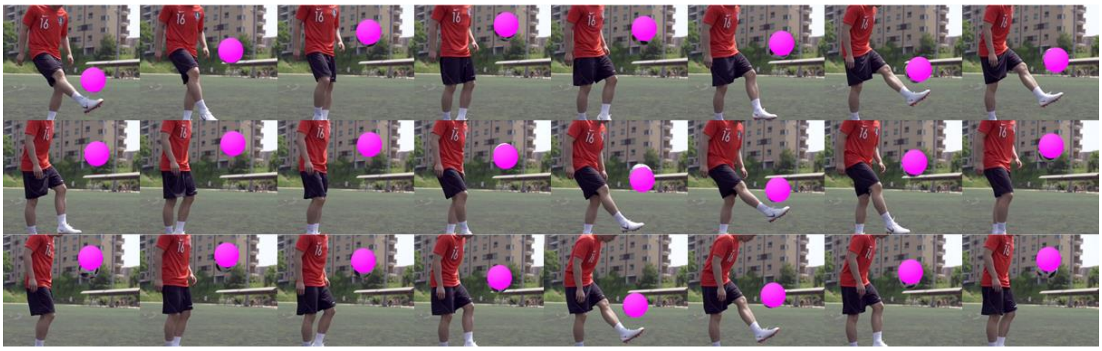


Fig. 7. Typical model results for soccer ball lifting

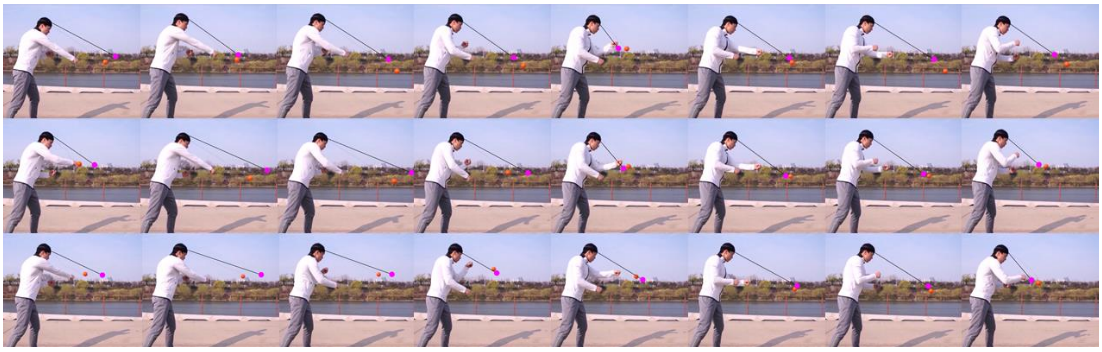


Fig. 8. Typical model results for tap ball

The proposed framework was implemented using a computer with Intel Core i7-7700K CPU, 48GB RAM, and NVIDIA GeForce GTX 1080Ti GPU to generate experimental results. Videos used in the experiment were all 1280×720 pixel resolution, and were shot at 30 frames per second. And we used game engine, Unity for generating our results.

Table 4 shows typical acrobatic motion information generated from user videos. Video length is the captured user video time, and number of frames refers to how many frames were used to analyze user pose from the input video. Calculation time is the time required analyze user pose and generate object motion to synchronize with the video. We selected six representative moving videos for this paper, excluding simple repetitive motions, with details as shown in **Table 4**.

① Hitting the soccer ball with right and left feet alternately while moving forward.

② Hitting the soccer ball with right and left thighs alternatively while moving forward.

③ Hitting the soccer ball with left foot and right knee alternately while moving forward.

④ Hitting the tap ball with right and left fists alternately while walking forward.

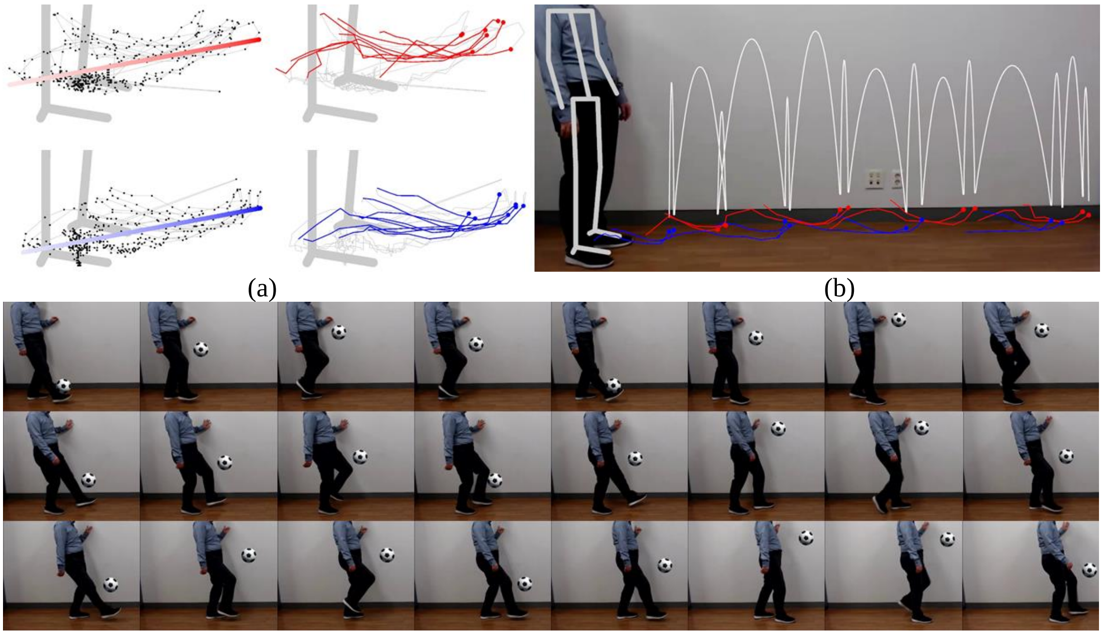
⑤ Hitting the tap ball with right and left fists alternately while walking backward.

⑥ Hitting the tap ball left and right fists alternately while simultaneously lifting the soccer ball with left and right feet alternately, while moving forward.

Table 4. Synthesized video details

No.	Description	Video length (s)	Number of frames	Calculation time (s)
1	Soccer Lifting with Both Feet Moving Forward	15.2	457	9.6
2	Soccer Lifting with Both Thighs Moving Forward	12.8	385	8.0
3	Soccer Lifting with Foot and Thigh Moving Forward	14.8	444	9.2
4	Tabball Hitting with Both Fists Moving Forward	10.0	301	6.5
5	Tabball Hitting with Both Fists Moving Backward	13.3	401	8.3
6	Soccer Lifting and Tabball Hitting Moving Forward	8.4	253	5.4

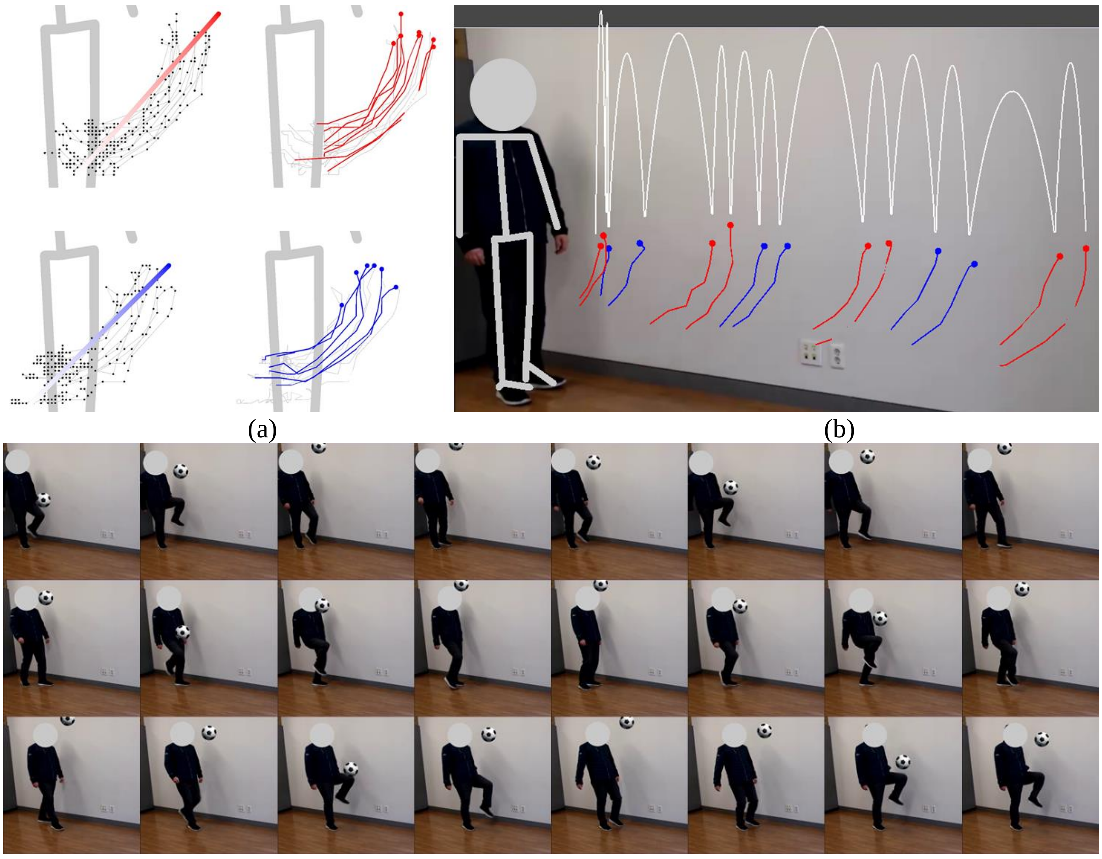
Fig. 9 to **14** show experiment outcomes using videos 1 to 6 from **Table 4**. Subfigures (a) show user pose analysis outcomes in relative coordinates. The left pair of images indicate where the hitting part moved with the point cloud (black dots) and average hit direction (the thick line). Red and blue lines indicates left and right areas, respectively. The right pair of images show hit lines and positions (red and blue curves and dots). Subfigures (b) show the generated object trajectories in the original coordinates from the input video. White curves represent the object trajectory, and red and blue curves and dots represent hit lines and positions for left and right areas, respectively. Subfigures (c) shows a series of images from the resulting video. **Fig. 9** to **11** were captured every 0.2 s and **Fig. 12** to **14** every 0.1 s. Black lines indicate the rubber band connected to the tap ball object (where appropriate). **Fig. 9** and **Fig. 14** were captured almost perpendicular to the camera, whereas **Fig. 10** to **13** were captured at a slightly oblique angle to ensure accurate results by avoiding significant occluded portions.



(a) (b)

(c)

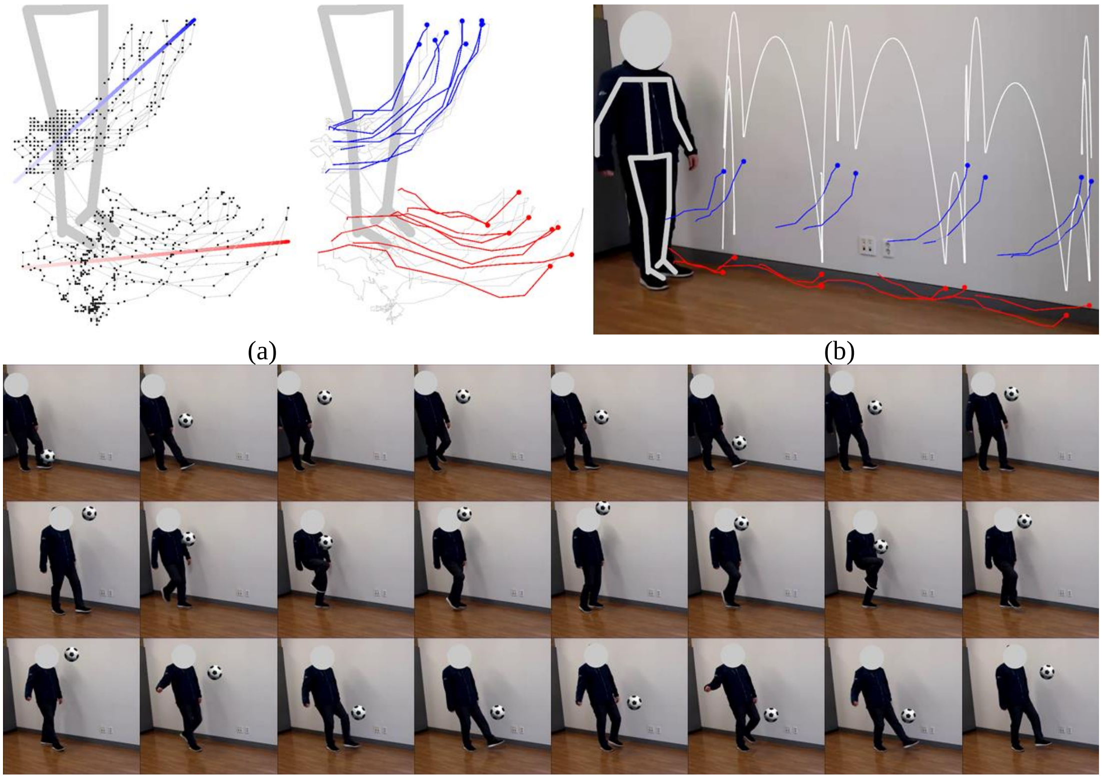
Fig. 9. Soccer lifting while moving forward



(a) (b)

(c)

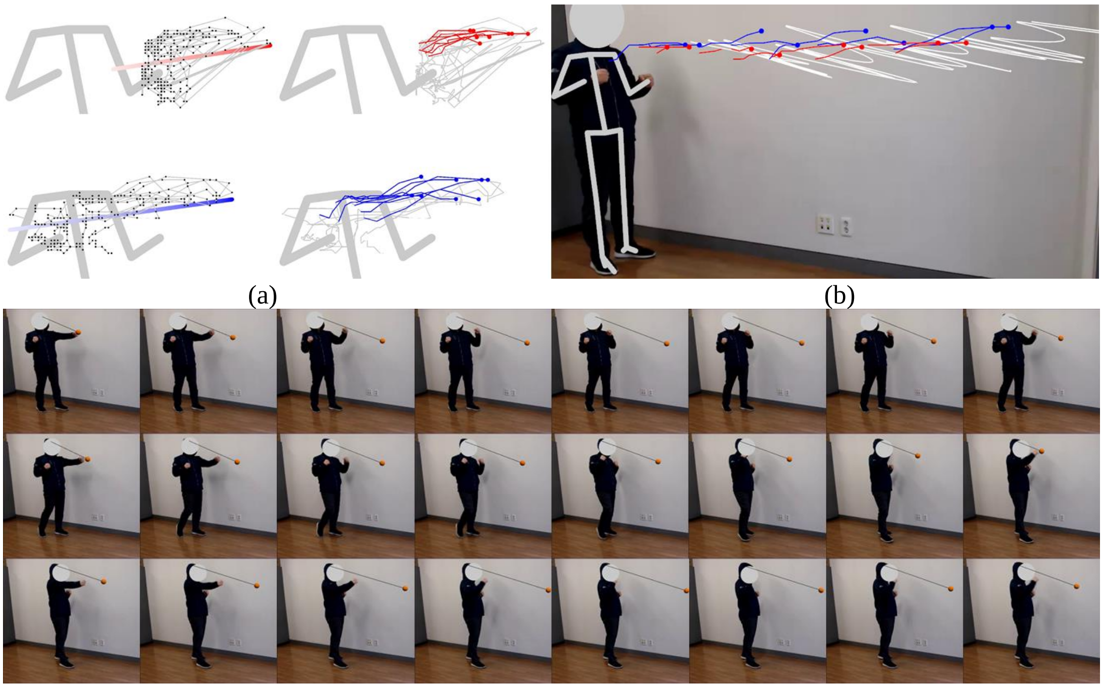
Fig. 10. Soccer lifting from thighs while moving forward



(a) (b)

(c)

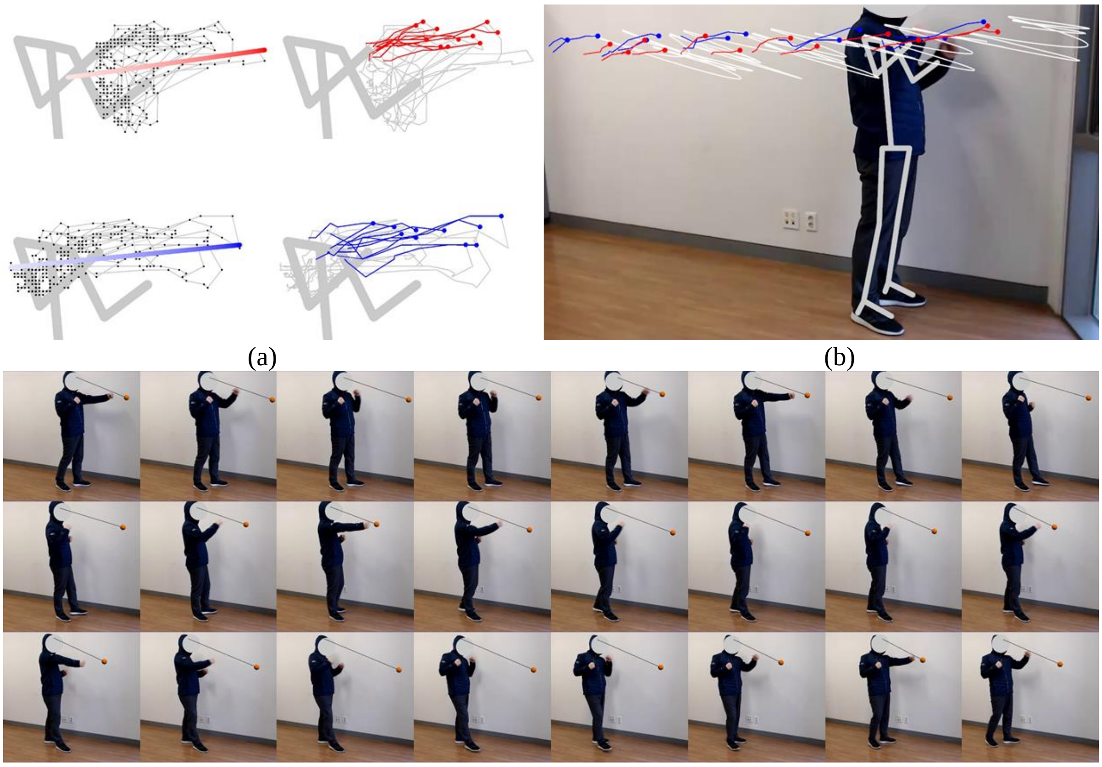
Fig. 11. Soccer lifting from feet and thighs while moving forward



(a) (b)

(c)

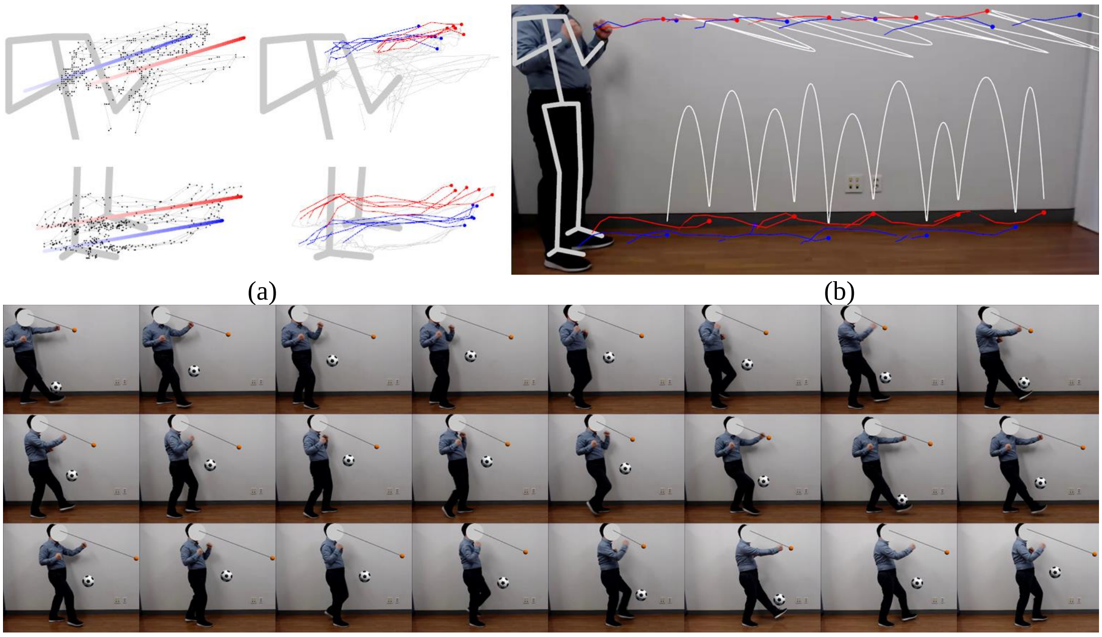
Fig. 12. Tap ball with both fists while moving forward



(a) (b)

(c)

Fig. 13. Tap ball with both fists while moving backward



(a) (b)

(c)

Fig. 14. Soccer lifting with both feet and tap ball with both fists while moving forward

5. Discussion

The proposed method was intended to allow adding virtual objects into existing video, and several somewhat awkward aspects remain: color differences depending on light direction, object shadow (or absence), and the object is always in front of the video. It also looks somewhat unnatural how the foot touches the ball due to errors tracking user poses in 2D coordinates. As discussed above, awkward hitting was alleviated by adding an offset to the hitting point.

The proposed method provides a useful mechanism to model an object while hitting it, but cannot maintain contact with or hold the object. Thus, maintaining the object on the instep or thigh cannot be modelled. In addition, scenes alternating feet and thigh with the same leg cannot be automatically generated, although this can be accomplished by manually distinguishing what foot or thigh to use. It is also impossible to process occluded areas, due to motion tracking limitations from a video. Hence, the video shot direction must be carefully selected to ensure the complete pose well exposed. The most effective direction was approximately 60° for stationery shots.

We set the criterion to remove noise as the hit line having length less than 30% of maximum hit line length. However, this sometimes removed valid hit lines. Therefore, noise removal was performed while simultaneously checking against the video. The adjacent hit time interval criterion (less than 15 frames) worked well for soccer ball lifting, but not as well for tap ball, since the user could quite easily swing their fists somewhat faster than this. Therefore, the criterion was reduced to 10 frames for tap ball videos.

6. Conclusion

This paper proposed a framework to generating an acrobatic video adding a virtual object with various motions by analyzing the input video and generating pose tracking and object motion. Creating such a combined video manually would require considerable time and effort, whereas the proposed method produced acceptable results easily and within seconds. We considered various irregular motions for the experiments, but these also needed to motions any user could easily learn, and hence we were limited to somewhat repetitive and regular motions. However, the proposed framework is capable of representing more complex and irregular motions. For example, feet and thighs could be used alternately at different times, and it is also possible to lift the soccer ball with the heels. As users became more familiar with the exercise, the could also handle the balls irregularly with both feet and fists.

The proposed method will help humans solve interesting problems with object movements; allowing simple inclusion of regular interaction between humans and objects, as commonly seen in real surroundings, to create various application scenes. Thus, it is a very important to interact between humans and the environment. Currently, character motion is created against a fixed environment, but the proposed framework allows object motion to respond to human motion. This provides an alternative to human-object interaction, and could be applied to various entertainment industries. It requires considerable time to learn complex techniques, such as lifting a soccer ball. However, the proposed framework would allow anyone can to perform such difficult movements in an augmented reality environment within seconds. Putting this augmented reality application onto mobile phones would provide new user experiences not previously considered possible.

Future studies will solve the various problems noted above and expand the proposed method in various ways. We will allow users to generate acrobatic videos in real time by receiving user motion from webcams, and expand the framework to allow multiple users to create collaborative acrobatic motions. Various obstacle objects can be introduced to provide means for users to manipulate virtual objects more intuitively and complexly. We also plan to expand the object handling areas to other body parts, such as hands and heads, to create more dynamic acrobatic scenes.

References

[1] D. Holden, T. Komura, and J. Saito, "Phase-functioned neural networks for character control," *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 1-13, 2017. Article (CrossRef Link)

[2] X. Peng, G. Berseth, K. Yin, and M. Panne, "Deeploco: dynamic locomotion skills using hierarchical deep reinforcement learning," *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 1-13, 2017. Article (CrossRef Link)

[3] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H. P. Seidel, W. Xu, D. Casas, and C. Theobalt, "Vnect: Real-time 3d human pose estimation with a single RGB camera," *ACM Transactions on Graphics*, vol. 36, 2017. Article (CrossRef Link)

[4] D. Mehta, O. Sotnychenko, F. Mueller, W. Xu, M. Elgharib, P. Fua, H. Seidel, H. Rhodin, G. Pons-Moll, and C. Theobalt, "Xnect: Real-time multiperson 3D human pose estimation with a single RGB camera," *ACM Transactions on Graphics*, vol. 39, no. 4, 2020. Article (CrossRef Link)

[5] Z. Cao, G. Hidalgo, T. Simon, S. Wei, and Y. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43 no. 1, pp. 172-186, 2018. Article (CrossRef Link)

[6] M. Andriluka, S. Roth, and B. Schiele, "Pictorial structures revisited: People detection and articulated pose estimation," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1014-1021, 2009. Article (CrossRef Link)

[7] S. Johnson and M. Everingham, "Clustered pose and nonlinear appearance models for human pose estimation," in *Proc. of the British Machine Vision Conference*, pp. 1-11, 2010. Article (CrossRef Link)

[8] A. Bulat and G. Tzimiropoulos, "Human pose estimation via convolutional part heatmap regression," in *Proc. of European Conference on Computer Vision*, vol. 9911, 2016. Article (CrossRef Link)

[9] V. Ramakrishna, D. Munoz, M. Hebert, J. Bagnell, and Y. Sheikh, "Pose machines: Articulated pose estimation via inference machines," in *Proc. of European Conference on Computer Vision*, pp. 33-47, 2014. Article (CrossRef Link)

[10] M. Sun and S. Savarese, "Articulated part-based model for joint object detection and pose estimation," in *Proc. of International Conference on Computer Vision*, pp. 723-730, 2011. Article (CrossRef Link)

[11] L. Pishchulin, A. Jain, M. Andriluka, T. Thormahlen, and B. Schiele, "Articulated people detection and pose estimation: Reshaping the future," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3178-3185, 2012. Article (CrossRef Link)

[12] U. Iqbal and J. Gall, "Multi-person pose estimation with local joint-toperson associations," in *Proc. of European Conference on Computer Vision*, vol. 9914, pp. 627-642, 2016. Article (CrossRef Link)

[13] G. Papandreou, T. Zhu, N. Kanazawa, A. Toshev, J. Tompson, C. Bregler, and K. Murphy, "Towards accurate multi-person pose estimation in the wild," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3711-3719, 2017. Article (CrossRef Link)

[14] A. Toshev and C. Szegedy, "Deeppose: Human pose estimation via deep neural networks," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1653-1660, 2014. Article (CrossRef Link)

[15] W. Ouyang, X. Chu, and X. Wang, "Multi-source deep learning for human pose estimation," in *Proc. of 2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2337-2344, 2014. Article (CrossRef Link)

[16] J. Tompson, A. Jain, Y. Lecun, and C. Bregler, "Joint training of a convolutional network and a graphical model for human pose estimation," *arxiv:1406.2984*, 2014. Article (CrossRef Link)

[17] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Proc. of European Conference on Computer Vision*, vol. 9912, pp. 483-499, 2016. Article (CrossRef Link)

[18] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. Gehler, and B. Schiele, "Deepcut: Joint subset partition and labeling for multi person pose estimation," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4929-4937, 2016. Article (CrossRef Link)

[19] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, and B. Schiele, "Deepercut: A deeper, stronger, and faster multi-person pose estimation model," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 9910, pp. 34-50, 2016. Article (CrossRef Link)

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 7, pp. 770-778, 2016. Article (CrossRef Link)

[21] C. Twigg and D. L. James, "Backward steps in rigid body simulation," *ACM Transactions on Graphics*, vol. 27, no. 3, 2008. Article (CrossRef Link)

[22] J. Popovic, S. Seitz, M. Erdmann, Z. Popovic, and A. Witkin, "Interactive manipulation of rigid body simulations," in *Proc. of ACM SIGGRAPH Conference on Computer Graphics*, 2001. Article (CrossRef Link)

[23] R. Fattal and D. Lischinski, "Target-driven smoke animation," *ACM Transaction on Graphics*, vol. 23, no. 3, 2004. Article (CrossRef Link)

[24] A. Treuille, A. McNamara, Z. Popovic, and J. Stam, "Keyframe control of smoke simulations," *ACM Transaction on Graphics*, vol. 22, no. 3, pp. 716-723, 2003. Article (CrossRef Link)

[25] C. Wojtan, P. Mucha, and G. Turk, "Keyframe control of complex particle systems using the adjoint method," in *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 15-23, 2006. Article (CrossRef Link)

[26] J. Barbˇı and J. Popov´ı, "Real-time control of physically based simulations using gentle forces," *ACM Transactions on Graphics*, vol. 27, no. 5, 2008. Article (CrossRef Link)

[27] S. Jain and C. Liu, "Interactive synthesis of human-object interaction," in *Proc. of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 47-53, 2009. Article (CrossRef Link)

[28] J. Choi, S. Kang, C. Kim, and J. Lee, "Virtual ball player," *The Visual Computer*, vol. 31, pp. 905-914, 2015. Article (CrossRef Link)

[29] J. Choi, S. Kim, C. Kim, and J. Lee, "Let's be a virtual juggler," *Computer Animation and Virtual Worlds*, vol. 27, no. 3-4, pp. 443-450, 2016. Article (CrossRef Link)

[30] J. Chemin and J. Lee. "A physics-based juggling simulation using reinforcement learning," in *Proc. of the 11th Annual International Conference on Motion*, pp. 1-7, 2018. Article (CrossRef Link)

[31] S. Hong, D. Han, K. Cho, J. S. Shin, and J. Noh, "Physics-based full-body soccer motion control for dribbling and shooting," *ACM Transactions on Graphics*, vol. 38, no. 4, pp. 1-12, 2019. Article (CrossRef Link)

[32] J. Merel, S. Tunyasuvunakool, A. Ahuja, Y. Tassa, L. Hasenclever, V. Pham, T. Erez, G. Wayne, and N. Heess, "Catch & Carry: reusable neural controllers for vision-guided whole-body tasks," *ACM Transactions on Graphics*, vol. 39, no. 4, 2020. Article (CrossRef Link)

[33] H. Eom, D. Han, J. S. Shin, and J. Noh, "Model Predictive Control with a Visuomotor System for Physics-based Character Animation," *ACM Transactions on Graphics*, vol. 39, no. 1, 2019. Article (CrossRef Link)

[34] S. Starke, Y. Zhao, T. Komura, and K. Zaman, "Local motion phases for learning multi-contact character movements," *ACM Transactions on Graphics*, vol. 39, no. 4, 2020. Article (CrossRef Link)

Jong In Choi received PhD at Korea University in 2016 from the Department of Computer Science from Korea University. After completion, he joined Nexon Korea as a lead client programmer. He has worked at NCSoft Korea as a lead animation programmer in a new AAA online game. Now he is a professor in department of digital media design and application in Seoul Women’s University.

Sang Hun Nam received Ph. D degree in computer graphics and virtual reality from graduate school of advanced imaging science, multimedia & film in Chung-Ang University, Seoul, South Korea in 2012. He was the Senior researcher at the Center of human-centered interaction for co-existence organized by Korean Government. He was an Assistant Professor with the New Media, Seoul Media Institute of Technology. He has been an Assistant Professor in department of Culture Technology, Changwon National University.