J. Vis. Commun. Image R. 71 (2020) 102812

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

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

Generating video animation from single still image in social media based on intelligent computing ☆

Tao Hu [a,b], Chao Liang [a], Geyong Min [c], Keqin Li [d], Chunxia Xiao [a,*]

[a] School of Computer Science, Wuhan University, Wuhan 430072, China
[b] School of Information Engineering, Hubei Minzu University, Enshi, Hubei 445000, China
[c] College of Engineering, Mathematics and Physical Science, University of Exeter, Exeter EX4 4QF, United Kingdom
[d] Department of Computer Science, State University of New York, New Paltz, NY 12561, USA

ARTICLE INFO

Article history:
Received 14 October 2019
Revised 16 February 2020
Accepted 4 April 2020
Available online 23 July 2020

Keywords:
Animation
Convolutional neural network
Image motion analysis
Shape context
Stochastic motion texture

ABSTRACT

Bringing a single still image into reality is a challenging topic in computer animation because the driven and structural information in single still image is inadequate. In this paper, we present an image animating method for enhancing single still image in social media with virtual realistic and animated motions without prior information. We imitate the interaction between the active objects in an image and their neighboring passive objects. The existing actions in the image and the virtual specified force are employed to animate the active objects. Observing that the change between two subsequent motions of the active objects derives a motion tendency, we can calculate a virtual driving force based on the motion tendency. By virtue of the virtual driving force, the stochastic motion texture is used to animate the passive objects. Finally, the convolutional neural network is employed to optimize the virtual motion animations. In this way, the proposed method produces visually natural results while guaranteeing motion harmony between active objects and passive objects. To demonstrate the applicability and rationality of virtual animation driving force, our method generates several animations from still images in Social Media.

1. Introduction

When appreciating a photograph in social media that includes a number of objects (*e.g.* animals and plants), we can subconsciously comprehend much more information than the static image. We may imagine that the objects in the image will exhibit some certain movement phenomena with the laws of nature. For example, given an image with some seagulls standing on the branches, we may imagine that the seagulls are swinging their head and the branches are slightly swaying. The proposed image animation framework has many potential applications in social media, such as image/video enhancement editing and synthetic, personal photo album editing, movie and game production, and augmented reality. Moreover, the other image animating models, for example, the stochastic motion texture driven by the wind, can also be integrated into our system, which will make our system more versatile.

Without adequate prior information, driving video, such as the 3D shape of the objects, wind strength, animating a static image is very challenging because it is difficult to find out enough key animating information from one picture. Many research efforts have been made on animating a still image [1–6]. For example, ELOR et al. [1] used driving video to generate a portrait video sequence from a still target portrait image based on 2D Warp and continuously transferring fine-scale, which is one of prior information methods. Chuang et al. [2] animated the still picture using stochastic motion textures. This method works well for animating the passive elements such as plants, trees, water and clouds. Xu *et.al.* [3] animated a still image of a moving animal group by morphing among the ordered snapshots extracted from the image. The core idea in this method is the shape matching of the animal's motions. This method generates a motion cycle of the animal object based on the result of shape matching. Our method aims to process complex image that contains more types of objects and to animate a still image by simulating the interaction between the active objects and their neighboring passive objects. Compared to [3], we focus on different research points, such as analyzing the principle of how to generate a virtual force based on object's motion in still image.

☆ This paper has been recommended for acceptance by Zicheng Liu.

* Corresponding author.
E-mail addresses: cliang@whu.edu.cn (C. Liang), g.min@exeter.ac.uk (G. Min), lik@newpaltz.edu (K. Li), cxxiao@whu.edu.cn (C. Xiao).

https://doi.org/10.1016/j.jvcir.2020.102812
1047-3203/

Guided by the available 3D models, Kholgade et al. [4] performed 3D manipulations on the objects in a photograph. Jhou et al. [5] presented a cloud appearance model which combines two computable properties: cloudiness and cloud structure, to synthesize the motion of cloud flow based on a content-aware wind field. Although these methods can produce visually pleasing results, there is still much room left to be improved. For example, most algorithms cannot animate the still image with complex natural scene because it is very difficult to find out the prior knowledge in still image, such as the correlation between many objects in the image and the animation driven methods for different types of objects.

Different from the existing literature [2–5], this paper aims at animating a still image by computing the interaction between the moving objects (active objects) and their neighboring objects (passive objects). For example, when some birds standing on small branches are swinging their heads, the branches should shake accordingly. Thus, to produce natural and realistic animation results, we need to compute not only the motion of the swinging birds, but also the motion of the branches. In addition, we need to guarantee the consistency between two types of movements.

Deep Learning is one of the most major researching topic in Computer Science. Some deep learning models are employed to generate fabricated videos from single input image. For example, Olszewski et al. [7] use GANs to generate realistic dynamic facial to infer realistic per-frame texture deformations from a Single Image. Those methods require a lots of data to train the deep learning model, which is a very complicated and time-consuming process. Whether need a lot of data driving, that is the significant difference between deep learning methods and our method. And based on the benefits deep learning in image interpolation [8–11], our method uses the convolutional neural network (CNN) to smooth the generated video animation.

We animate the motion of active objects such as swinging birds using the existing actions [3] in the original image. For some active objects, such as the athlete on the diving board, and the dropping raindrop, we simulate their motion according to automatic falling model or Newton's force model. The key problem to compute the interaction between the active objects and the passive objects is to derive the driving force, which will be used in stochastic motion synthesis model [2,12]. We observe that the change between two subsequent motions of the active objects will determine a motion tendency. With mechanical analysis for the motion tendency, we derive the motion force that drives the surrounding passive objects to move.

The major contributions of this paper are summarized as follows:

(1) A new animation method is proposed to generate a video animation from still image based on intelligent computing. Our method classifies active and passive objects in the image by interaction. The active objects can generate a virtual driving force to drive the passive objects.
(2) An innovative analytical method is designed to generate a virtual driving force for active objects. Based on the shape context of each active object, we calculate the motion tendency of their similarity. Then, we employ the Newton's laws of motion [13] to estimate the magnitude and direction of virtual driving force.
(3) We specify two types of animation driving force: virtual driving force and virtual wind force. Joining the animation driving force and stochastic motion texture, we calculate a time-varying motion map for each object, which is used to generate a new motion in new animation frame based on last motion. And CNN is used to smooth the generated animation frames. So our method can simultaneously drive the active objects and passive objects in the image to generate complex video animation.

The rest of this paper is organized as follows: We first present the related work in Section 2, and then describe the framework of our system in Section 3. In Section 4, we present the technical details of the proposed method. Experimental results and discussion are presented in Section 5. Finally, we conclude our work and present some future research directions in Section 6.

2. Related work

Over the past few years, many studies have been conducted on animating a static image, leading to numerous classic animation generation methods, such as stochastic motion texture model, shape contexts matching model, 3D motion capture data driven model, and image interpolation.

The prior knowledge is very useful to animate motion from still image because it can help to achieve physically realistic results. Simulating the natural motion phenomenon based on stochastic models is a classic problem. Stochastic models have been widely used in animating the fractional Brownian motion of terrains [14]. For example, Chen and Johan [15] proposed a 2D method for real-time animation of vegetation in 3D scenes based on 2D harmonic motion. Okabe et al. [16] used the fluid video database to synthesize fluid animation from a single image. Based on two-phase bond between water particles, which are located on a non-absorbent hydrophilic surface, Chen et al. [17] converted an input still image into a water-art-style artwork. Motion driven by wind is a common method in animating motion form still image. Sun et al. [18] presented an inverse harmonic oscillation method to extract parameters of wind and regular water waves, then, they used harmonic oscillation model to synthetic object motion based on the extracted parameters.

However, the stochastic motion texture model can not generate complex motion from the still image. Using 3D motion capture data to animate photos of 2D characters [19] is another effective animating method, which transfers the motion of a 3D skeletion onto a 2D shape in image space and generates realistic movement. Jain et al. [20] proposed an augmenting method to create a hand-drawn animation of human characters with 3D physical virtual effects, which the driving points in two dimensions are reconstructed into three dimensions. They further employed 3D proxies to connect hand-drawn animation and 3D computer animation to generate more complex and smooth 2D animation [21].

In some common cases, image interpolation [22,23,26] is a common technique to generate image animation results. It is used to reduce blurring artifacts between non-successive motions to improve the quality of animation video. Romano et al. [24] combined non-local self-similarity and sparse representation modeling to image interpolation. Xu *et.al.* [3] animated a still image of a moving animal group by morphing among the ordered snapshots extracted from the image. The core idea in this method is to shape matching of the animal's motions. This method generates a motion cycle of the animal based on the results of shape matching. In our paper, the convolutional neural network [8–10,25] is used to generate more smooth virtual motion actions by motion interpolation.

In our approach, we are interested in generating an animation video from still image based on intelligent computing. In this paper, we employ the stochastic motion texture to calculate the motion displacement of objects driven by virtual motions, which are generated by our method, aiming at processing complex image that contains more types of objects. We target to animate a still image by simulating the interaction between the active objects

and their neighboring passive objects. Compared with [3], we focus on different research questions, for example, analyzing the principle of how to generate a virtual force based on object's motion in still image and smoothing the video based on DeepMotion CNN model [8]. What's more, our method does not employ 3D model library to generate a virtual motion.

3. The proposed system and framework

In this section, we describe the main idea on how to generate a virtual motion frame, and then describe the framework of our system.

3.1. System overview

Given a single image, we aim to animate the specified objects and their interaction with the surroundings. We specify the objects that generate the driving force as *ForceObjects*. Image usually includes three kinds of objects: *ForceObjects*, *ForceObjects*' surrounding objects, and other objects (Fig. 1). Different models are employed to drive motion for different objects. The animation of *ForceObjects* is created according to different motion styles. The animation of *ForceObjects*' surrounding objects and other objects are inferred from physical analysis of mechanics and natural phenomena. So we must firstly specify the motion type for different animation objects in the still image through user interactions. If there is a single animating object in the still image, we can also specify whether the animating driven force is the wind force or another virtual force. Based on the object's type and the driven force, we can then generate a list of virtual motions using the same algorithm.

A time-varying motion map M is defined to create a motion in one frame. This motion map $M(K(p), p, t)$ is a function of motion class $K(p)$, pixel p and time t, where $K(p)$ returns the motion type of the pixel p. Applying the map directly to a potential animating object $O(p, t)$ results in a forward motion $O(p, t+1)$ such that

$$O(p, t+1) = O(p, t) + M(K(p), p, t) \quad (1)$$

where $M(K(p), p, t)$ is a relative displacement of motion class $K(p)$ in pixel p between time $t+1$ and t. And the forward motion $O(p, t+1)$ indicates the whole displacement of motion class $K(p)$ at time $t+1$. In other words, $O(p, t+1)$ shows a new motion of $K(p)$.

Based on different motion class K, the motion map M is defined as follows, which are specified by the users of our system:

- *ForceObjects* motion: the motion map is calculated from the motion cycle.
- *ForceObjects*' surrounding object motion: The motion map is calculated from the force generated by *ForceObjects*.
- Other objects: the motion map is calculated by wind force.

We focus on three types of virtual motion, which can generate an animation video from a still image. These virtual motions can be described as motion cycle movement, natural gravity movement and swimming. Each virtual motion employs the basic method of stochastic motion texture to calculate the motion map. But the principles to trigger virtual motion are not the same, and Table 1 lists the differences between each type of virtual motion.

3.2. The framework of our system

Based on the motion map, the displacement of each object can be calculated. Using the map, we construct new motion for each object. Then, we synthesize new motion frame according to the new motions. Finally, we render the animation video based on motion frames. The framework of our system (Fig. 2) consists of the following steps: motion objects extracting, driving force analysis and motion editing, and finally animation rendering.

(1) Motion objects extracting. The first step is to segment the input image I into several motion classes. The same class is animated with the same motion model. For example, for the image in Fig. 2(a), we have the following classes: seagull and tree stem. We assume that stem shaking is caused by the swinging of seagulls' head.

Each extracted object O_i consists of color information I_i, alpha matte α_i and motion type K_i. Considering that there may be different motions with the same kind of objects in the single image, a global combination order OL is generated by comparing the similarity between the motion shape of *ForceObjects*. The OL determines the rendering order of motion in the final frame of motion animation. In order to ensure the continuity of the virtual animating motions, we employ the motion cycle to indicate the motion order for each kind of objects based on OL.

Driving force analysis and Motion editing. Editing the motion of each driven object is another work in this process. Currently, we provide three motion types: *ForceObjects* motion, object motion driven by *ForceObjects* (shaking, swaying), and object motion driven by the wind force. For each motion type, we use novel method to generate the animation results that confirm with the law of nature.

Through the global combination order OL, a virtual motion cycle is generated from the motions extracted from a still image. Through simulating the stochastic motion, motion textures are time-varying 2D displacement maps for each static object. Motion textures are employed to drive those objects (such as leaves, trunk and water) to take some actions. There is a close connection between motion cycle and motion textures. And virtual motions

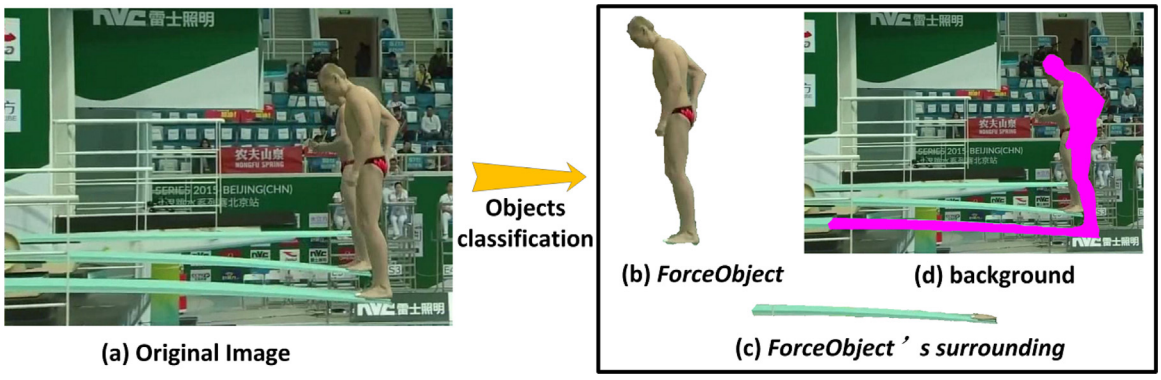


Fig. 1. An example of object classification in an image.

Table 1
The principle of each virtual motion animation.

Virtual motion type	The principle of implementation
Wind Force driven	Stochastic motion texture
Motion cycle driven	Interaction force generated by continuous motion
Gravity driven	Virtual gravity by estimating mass of *ForceObject*

will drive some objects to move. For example, if seagulls in Fig. 2(a) swing their heads side by side, a virtual force is generated to drive the trunk below the seagulls to shake up and down in 2-D plane. We describe the motion textures and motion cycle in detail in Section 4.

Animation rendering The virtual motion can be generated by using Eq. (1), and CNN model is used to interpolate more virtual frames between virtual motions. During the animating process, all virtual motions will be synthesized together with inpainted background image each time. Considering the shadowing and viewing effects, we use the global combination order *OL* to synthesize each virtual motion, and the background. Then the background will be updated as synthesizing results.

The virtual motion can be generated by using Eq. 1. After motion definition and editing process, we can construct two kinds of virtual motion: static object motion, and motion based on shape. Motion rendering combines virtual motion cycle, motion textures, combination order and time. For each time t, using *OL* to determine the rendering order of each virtual motion, using virtual motion cycle to determine which action to adopt, using motion textures to calculate the displacement of static objects. The rendering result is an animated video which contains some objects with a variety of actions and some objects with natural movement.

3.3. Stochastic motion texture model with virtual wind

Many natural motion phenomena, *e.g.,* a swinging leaf by the wind, can be described as harmonic oscillations [18], which are employed to simulate natural motions in computer graphics. In this section, we describe the core idea of stochastic motion texture based on harmonic oscillations and virtual motion driven by wind force.

A stochastic motion texture of an object is defined by a time-varying map of displacement, which is one kind of motion map. Through simulating harmonic oscillations with appropriate parameters, we can get the displacement of one object at time t. Then, we can construct the motion posture of the object using motion texture at time t.

To the best of our knowledge, several motion models are designed to simulate different motion phenomena with harmonic oscillations. In this paper, we design two models to simulate two kinds of motion phenomena; one is driven by wind force and another is driven by virtual object motion. Fig. 3 is the process flow for generating stochastic motion texture. We first introduce the stochastic motion texture and wind force driven animation, and then describe several typical static object motions in detail using stochastic motion texture with interaction driven forces in Section 4.

3.3.1. Stochastic motion texture based on harmonic oscillations

Considering animating motion of objects with different materials, we describe the harmonic oscillation firstly. Then, we present motion displacement propagation of the non-rigid objects.

The displacement $d(t)$ at the time of a damped harmonic oscillator is:

$$\ddot{d}(t) + \gamma\dot{d}(t) + 4\pi^2(f_o)^2 d(t) = \omega(t)/m \tag{2}$$

where m is the mass of oscillator, $\omega(t)$ is the driving force, $d(t)$ is the displacement of the oscillator at time t, γ is the damping coefficient, and f_o is the natural frequency. In other words, $d(t)$ means the displacement of the force point in the object at time t.

We use Newton's Equation to solve Eq. (2) and express the driving force in Fourier series expansion. Taking the Fourier transform of Eq. (2), the result becomes as Eq. (3).

$$-4\pi^2 f^2 D(f) + i2\pi\gamma f D(f) + 4\pi^2 f_o^2 D(f) = W(f)/m \tag{3}$$

where $D(f)$ and $W(f)$ are the Fourier transforms of $d(t)$ and $\omega(t)$, and $i = \sqrt{-1}$. So the solver of $D(f)$ can be described as follows:

$$D(f) = \frac{W(f)e^{i2\pi\theta}}{\sqrt{(2\pi(f^2 - f_o^2) + \gamma^2 f^2}} \tag{4}$$

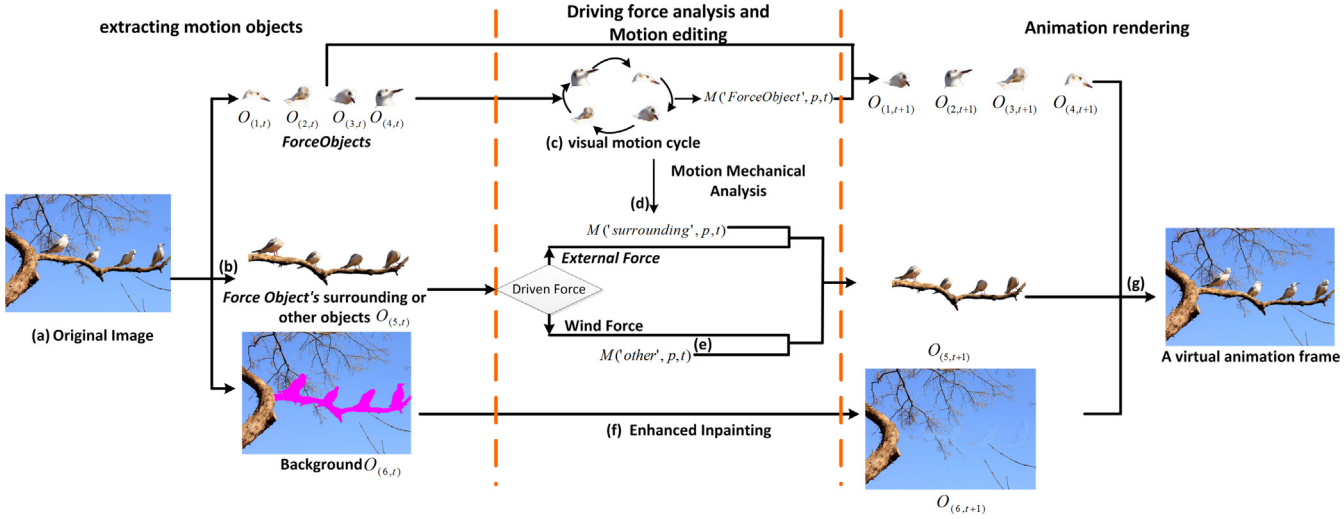


Fig. 2. System Framework. The input still image (a) is segmented into several motion classes, (b) each foreground animating class is then animated with time-varying motion map $M(k, p, t)$. Firstly, generating a virtual motion cycle, (c) through shape context, and then determining the type of driving force for each other object. For the objects surrounding with *ForceObject*, we calculate the driven force for these objects to get their motion map (d) through mechanics analysis of the *ForceObject* motion. For *ForceObject*'s surrounding objects (other objects), the time-varying motion map (e) that is driven by wind force can be simulated in stochastic modeling. We utilize the enhanced inpainting algorithm (f) to repair the background. Finally, different motions are composed together to produce a new frame. All frames are rendered to generate the final animation (g).

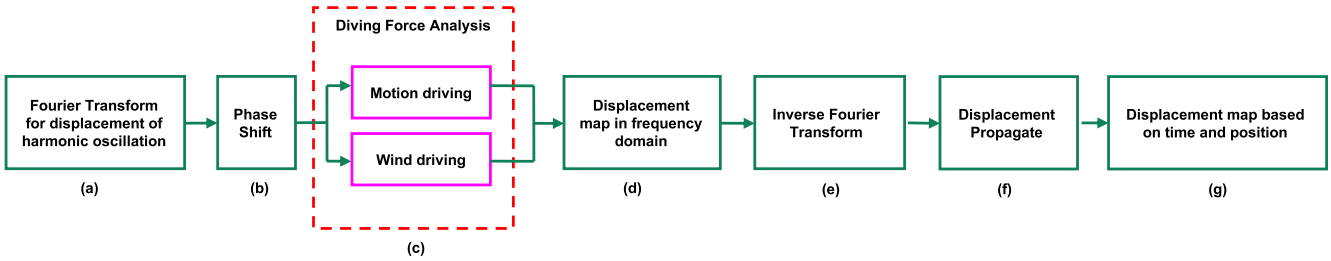


Fig. 3. The process flow of stochastic motion texture. There are seven steps in the flow. (a) Transform harmonic oscillation model into frequency domain. (b) Analyze the phase Shift of motion frequency. (c) Analyze two kinds of driving force: wind and motion force. (d) Generate the displacement of the force point in the object based on different driving force in frequency domain. (e) The displacement map transform into time domain for displacement using inverse Fourier transform. (f) Propagate the displacement from force point to other parts of the object. (g) Displacement map is generated after the above six steps. Using the map, we construct a new motion for one object at time t.

where f is the function of frequency, and the phase shift θ is defined as Eq. (5).

$$\theta = \arctan\left(\frac{\gamma f}{2\pi(f^2 - f_0^2)}\right) \tag{5}$$

The problem of solving Eq. (4) is how to get the approximate value of driving force. For the wind force which can drive the objects in the still image to perform a slight shaking or swinging, we can give some initial parameters of wind, such as velocity and direction. The object motion force, whose initial parameters are generated from the virtual motions, can drive the objects surrounding with *ForceObjects* to perform some motions consistent with the virtual motions.

3.3.2. Stochastic motion texture driven by wind force

In the wind energy community some fitting methods of velocity power spectra are often used. Based on Kaimal's formula and Kolmogrov theory, the fluctuating wind velocity is defined as follows:

$$E(v,f) = \frac{v_*}{(1 + \kappa f / v_*)^{5/3}} \tag{6}$$

where f is the frequency, $E(v,f)$ is the wind power spectrum, v_* is the shear wind speed, and κ is generally a constant of altitude. Based on $E(v,f)$, the velocity spectrum of wind force with a random Gaussian noise field $G(f)$ in the frequency domain is formulated as follows:

$$W(f) \approx G(f)\sqrt{E(v,f)} \tag{7}$$

Based on Eqs. (6) and (7), the wind force can be calculated. Integrating the wind force into Eq. (4) and performing inverse Fourier transform, we can obtain the displacement of the stress point driven by wind force.

4. Motion driven by stochastic motion texture and virtual force

In this section, we focus on simulating the interaction between the active objects and their neighboring passive objects. When animating the passive objects using stochastic motion texture, the key problem is to derive the driving force, which will be used in stochastic motion synthesis model. And we give the basic principles of driving the stochastic motion texture using two kinds of virtual motion.

4.1. Stochastic motion texture driven by virtual motion

Based on the change between two virtual motions defined by virtual motion cycle, a motion tendency is found. With mechanical analysis for the motion tendency, we can obtain a motion force which can act on the objects surrounding the *ForceObject*.

Mass is an important factor to mechanical analysis. For example, we can confirm that the gravity of diving athlete causes the wobbling motion of diving board, and the water on the leaf tip drops down based on the gravity of water. So we must estimate the mass of those *Forceobjects* in those scenes. Moreover, Eq. (2) is used to calculate the displacement $d(t)$ for each animating object, the mass of animating object should be known.

4.1.1. Mass estimating method based on particle number

Because it is not possible to get the mass of animating objects from a single image directly, we propose a mass estimating method based on particle number. We take each pixel of animating object as a particle. Different particles have different density. For simplicity, we employ the same mechanical analysis method for each animation object in a single image, which includes *ForceObjects* and their surroundings. We infer that the mass of animating object is proportional to the number of particles. The mass estimating method is described as:

$$mass_i = \log\left(\frac{Particle_i}{\sum_{i=1}^{n} Particle_i}\right) \cdot density_i \tag{8}$$

where $mass_i$ is the mass of animating object i, $Particle_i$ is the particle number of animating object i, $\sum_{i=1}^{n} Particle_i$ is the particle number of all animating objects, and $density_i$ is the particle density of animating object i.

4.1.2. Motion driven by motion cycle

For highly regular and repetitive motions [3] in a still image, we can calculate similarity between different motion actions based on their shapes. The shape context [27] is employed to quantify the shape feature of action, and describe the distribution of the relative positions of all action shape contour points in a spatial histogram. We first use intelligent scissors to extract all motion actions and their matte [28] from a still image. At the same time, the contour points are saved during extracting motion action. Then, we construct an action shape context by using a shape similarity metric, and measure the similarity between the two corresponding shape contexts. Finally, we infer the motion cycle based on the most similar actions among all motion actions.

Feature points, which express the appearance of the entire shape, are sampled from the contour points of each animal action. Shape context uses a histogram for each point on the shape to describe the distribution over the relative positions of other points in log-polar coordinate system. The motion matching measures the similarity of the feature points in A_i and their corresponding points in A_j. The distance $D(A_i, A_j)$ between shapes can determine the dis-

parity between two action shapes. The motion matching formulation is given by

$$D(A_i,A_j)=\frac{1}{C_i}\sum_{i\in A_i}\arg\min\left(\frac{1}{2}\sum_{k=1}^{K}\frac{[(g(k)-\tau(h(k)))]^2}{g(k)+\tau(h(k))}\right)+\frac{1}{C_j}\sum_{i\in A_j}\arg\min\left(\frac{1}{2}\sum_{k=1}^{K}\frac{[(h(k)-\tau(g(k)))]^2}{g(k)+\tau(h(k))}\right) \quad (9)$$

where C_i (C_j) is the total number of contour points of A_i (A_j); $g(k)$ ($h(k)$) is the shape context feature vector for the contour points of A_i (A_j). A shape transformation τ is employed to ensure the minimum cost of shape matching before calculating the difference between $g(k)$ and $h(k)$. A smaller $D(A_i,A_j)$ indicates more similar shapes.

A matching matrix is generated after matching all extracted motion actions in pair. Each row is the matching result between one action and others. We firstly find a minimum value of one row in the matrix, which means the largest similarity between the current action A_i and another action A_j. Then, for A_j, we determine its most similar action A_k using the same method. By iteratively searching the matrix, a motion cycle of object actions has been constructed.

Since the virtual motions are generated from different objects, some smooth transitions are employed to guarantee the virtual motion actions for each object before the rendering process. Those transitions keep the consistency in motion pose, morphology and appearance. For a contour point q_i on the motion action and its corresponding point $q\prime_i$ on the most similar action, an affine transformation matrix is used to ensure the pose consistency between q_i and $q\prime_i$. For each action A_i and its corresponding action $A\prime_i$, we use convolutional neural network [8] to ensure a smooth transition between A_i and $A\prime_i$.

Fig. 4 is one motion cycle example based on shape context. We initialize a motion order number for each running wildebeest (Fig. 4(b)). We calculate motion similarity by matching shape context of each wildebeest, as illustrated in Fig. 4(d). We construct a wildebeest running motion cycle based on the motion similarity matching matrix, as in Fig. 4(e). The motions in the red rectangular box (Fig. 4(b)) are the virtual smooth frames between two motions from the motion cycle. An example of motion shape matching of two motions (Fig. 4(b)) based on the shape context as shown in Fig. 5.

4.2. Motion driven by virtual force

There are two kinds of virtual motion: One is the driving force derived from the object motion cycle based on shape context, the other one is the driving force derived from the object gravity. And we explore the displacement propagation model of non-rigid objects.

4.2.1. Motion driven by virtual motion force

Assuming the swinging of seagull head is one kind of uniform motion, so the driving force is generated during the first change of motions, and is going to act on other motions. Fig. 6 shows the mechanical analysis process for seagull head shaking. According to the physics laws of motion, the driving force is generated at the beginning of motion. The rest of motion stages are in uniform motion state. We can construct the velocity field of head shaking list (Fig. 6(a)), which is illustrated in Fig. 6(b). The direction of yellow arrows means the motion direction from one shaking state to another. The length of yellow arrows means the motion distance correspondingly. Assuming the driving force is generated during the stage as demonstrated in Fig. 6(c). The kinetic energy produced belongs to the motion, which makes the head of seagull have a velocity. Based on Newton's laws of motion, we can calculate the velocity of the head shaking as follows Eq. (10).

$$force_k=\frac{m_k\Delta s}{\Delta t^2} \quad (10)$$

where m_k is the mass of seagull's head, Δs is the distance of seagull shaking, and Δt is the time between two motions. We employ Eq. (8) to estimate m_k. Based on Eq. (11), the total driving force, which drives the tree trunk and all seagulls to make harmonic motion, can be generated as:

$$Force=\sum_{k=1}^{N}force_k\cos\theta_k \quad (11)$$

where θ_k is the angle between driving force and vertical direction, and N is the number of seagulls. Based on the force $E(v,f)$ in Eqs. (6) and (7), the velocity spectrum of driving force for seagull head shaking with a random Gaussian noise field in frequency domain is estimated as follows:

$$W(f)\approx G(f)\sqrt{Force} \quad (12)$$

Using $W(f)$ in Eq. (12), we can calculate the displacement of tree trunk and all seagulls in frequency domain. Fig. 6(d) is the tip displacement of tree trunk within a time period, and Fig. 6(e) is the displacement of tree trunk within a time period.

4.2.2. Motion driven by gravity

Gravity is a common driving force to make object motion. For example, assuming there is one athlete standing on the diving board, and the athlete is going to dive into the water. The athlete will pause for a few seconds to ensure the diving board to become balance. As the gravity of the athlete can make the diving board wobble, we simulate the wobble animation driven by the gravity of the athlete.

Given a still diving image as shown in Fig. 7(a), we analyze the force of diving board (Fig. 7(b)). There is a gravity *Force_g* caused by the athlete in the point P_0 of diving board, and a holding force *Force_s* exists in the point P_1. When the athlete walks to P_0, the diving board will wobble strongly due to the impact of *Force_g*. Then, the wobble will stop gradually by the holding force *Force_s*. So the force *F_d* of diving board can be described as Eq. (13).

$$F\_d=Force\_g-Force\_s \quad (13)$$

where $Force\_g=mass_a\times g$, $mass_a$ is the mass of athlete, which can be estimated by Eq. (8), and g is the acceleration of gravity.

We can describe the wobble as that *F_d* generates a downward displacement to diving board at the beginning of wobble. We assume that the wobble motion of both athlete and diving board is just the results driven by the gravity of athlete. As we do not know any information of *Force_s*, the supporting function of diving board is ignored in the moment when the athlete stands on the diving board. The *F_d* drives the diving board to do upward moment in another time until the board and the athlete achieve a state of equilibrium. We get the $W(f)$ of *F_d* based on Fourier transform.

4.2.3. Displacement propagation model

In this section, we describe how to propagate the displacement from stress point to the whole object based on the features of stochastic motion model. We use the displacement propagation model to calculate the whole object's displacement from the stress point, such as diving board wobbling, leaf and trunk jittering. Due to the deformation of different objects driven by different external forces, we use a trunk as an example to describe the displacement propagation for non-rigid object.

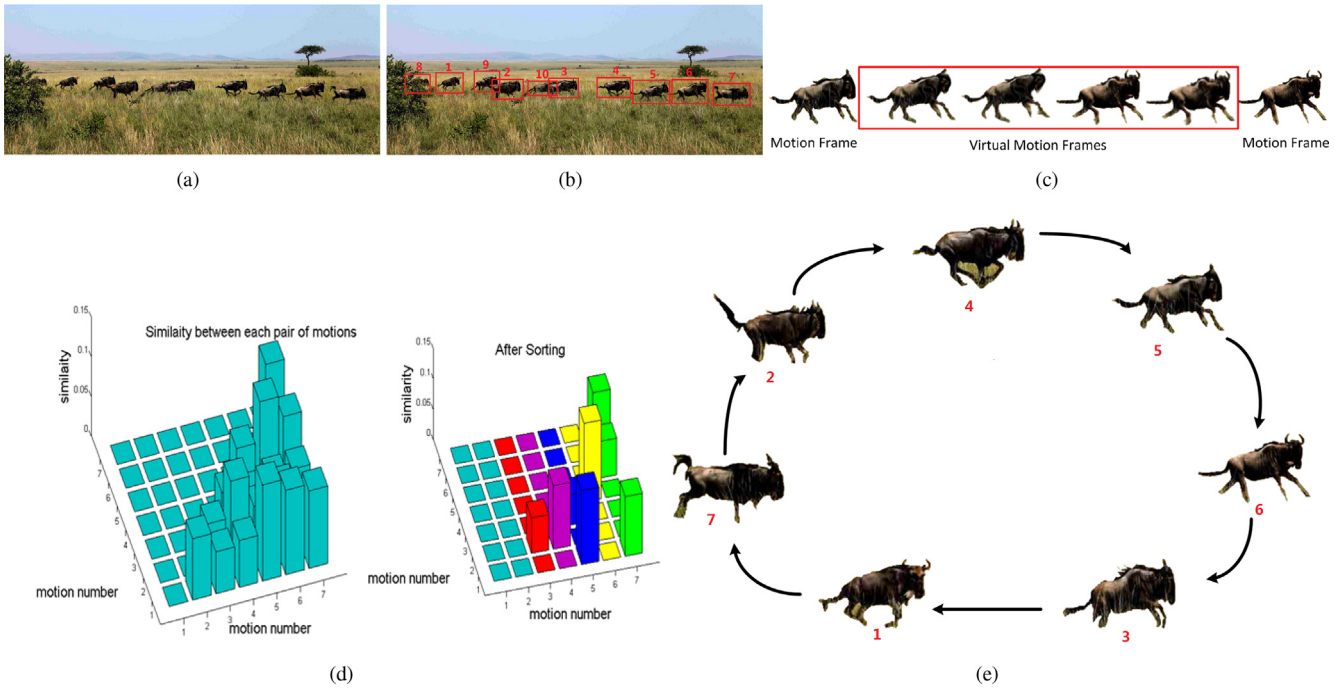


Fig. 4. The example of virtual motion cycle. (a) The original still image; (b) the initialization motion order of each running wildebeest; (c) the smooth frames between two motion; (d) the histogram of motion similarity by matching the motion shape context, which are extracted from (b); The left histogram in (d) is the total similarity of each motion pair, and the right histogram is the largest similarity between each motion pair; (e) the motion cycle resulting from (d), while the matching animation motion order is: 5→6→3→1→7→2→4.

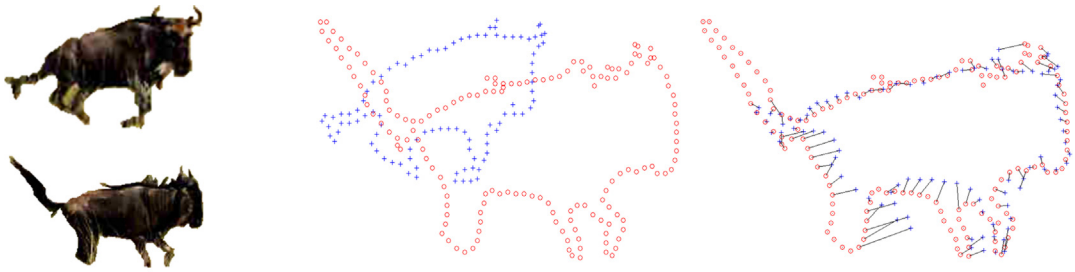


Fig. 5. Motion matching example. (a) The extracted motions of wildebeest from Fig. 4(a), (b) the original shape of two motions, (c) the matching result based on shape context, and the original shapes have been transformed before shape matching.

For example, we can calculate the shaking displacement of one point on the trunk based on Eqs. (4) and (7) in frequency domain. After the inverse Fourier transform, the displacement has been changed in time domain. Then, we employ our displacement propagation model to estimate the displacements of other points on the trunk.

When wind blows, the jitter displacement of branches and trunks of the tree are different. For example, the jitter displacement of tip is much larger than the displacement in root. So we estimate the motion of trunk by a point to plane model. In this model, the leaf, branch and trunk are segmented into several parts with 2D-line segment method. We assume that the tip is stress point. We use a line propagation method to distribute the displacement from tip to root of the object based on this segmentation. The propagation model of the non-rigid object can be simplified to the form:

$$d(s,t) = d_{tip}(t)\left[\frac{1}{3}s^4 - \frac{4}{3}s^3 + 2s^2\right] \tag{14}$$

where $d_{tip}(t)$ is the displacement in tip at time t, which can be calculated by our model in Eqs. (2)and (7), s is the parametric representation of the line segment which ranges from 0 to 1, and $d(s,t)$ is the displacement in location s of the object at time t.

When the objects are moving from their original position to new position, the left blank background should be inpainted or completed, so as to avoid that a visible blank background appears during rendering. We use an image melding algorithm [29] to fill the blank regions in background image after extracting the objects, which can make the blank background regions have the consistent texture and content with their surroundings.

5. Experiments and results

We applied our animation model to several still images in social media. The accompanying animating videos for the input images contain three types of motion classes: the motion of *ForceObjects*, object's motion driven by virtual motion, and object's motion driven by wind force.

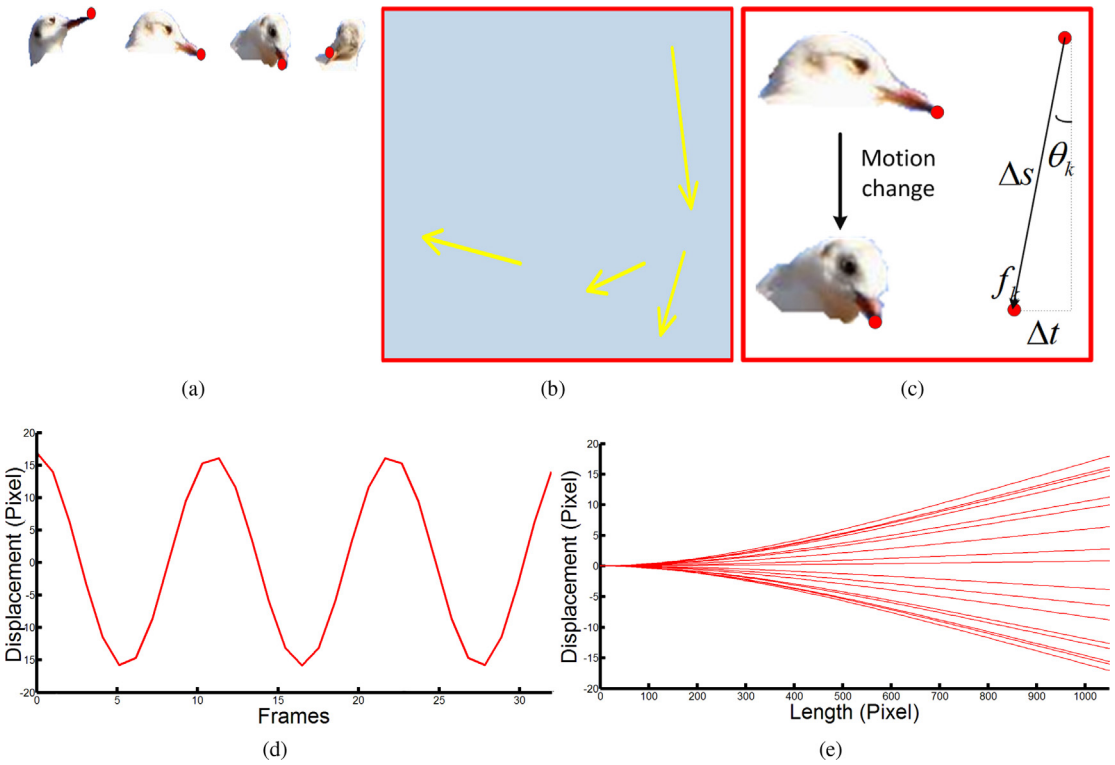


Fig. 6. Mechanical analysis process for seagull head shaking. (a) an example for seagull head shaking list, where the red points are the reference points of movement of the heads; (b) the velocity field of head motion of (a); (c) an example of mechanical analysis process for the motion changing; (d) the tip displacement of tree trunk within a time period; (f) the displacement of tree trunk within a time period.

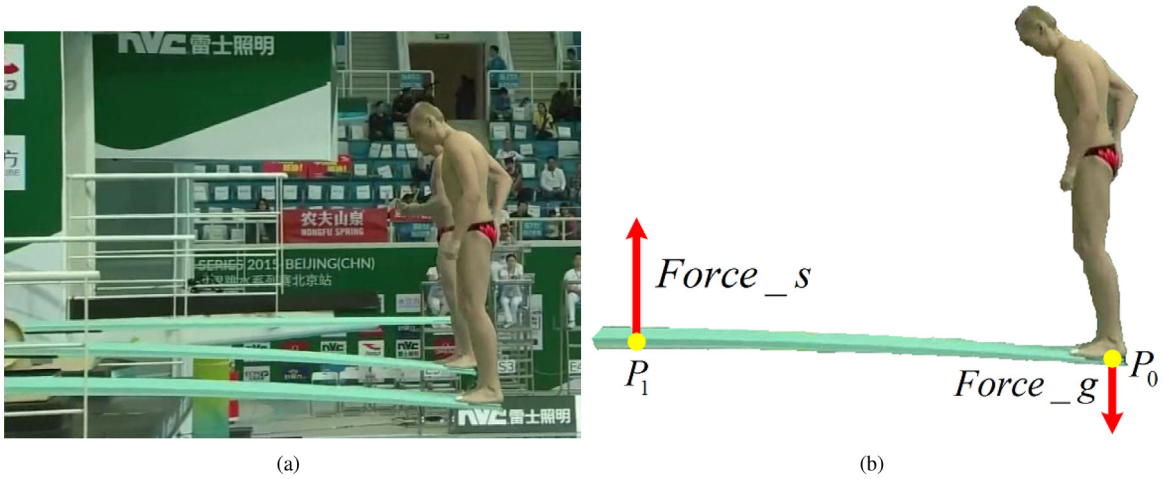


Fig. 7. Mechanical analysis for diving board. (a) The image with athlete standing on the diving board, (b) mechanical analysis for the diving board.

We test a list of still images to verify the usability of our animation model. We generate a wobbling animation to test the stochastic motion texture driven by virtual motion firstly. On the basis of primary stochastic motion experiment, we add the wind force to generate another wobbling scene. Then, an animation of seagulls head shaking is generated, which is also based on stochastic motion texture. The seagulls head shaking is going to cause a jittering phenomenon of trunk. To distinguish the original shape context model, we introduce a wind force driving animation in wildebeests running scene. Finally, we test the virtual swimming wake animation in the wild geese flying and swimming image.

We interactively set the initial animation parameters. Table 2 lists the important initialization parameters. For example, we specify the wobbling pivot and tip point of diving board in the original image. To get approximate mass value of the driven objects, we set the density value for them. If the wind is the driven force,

Table 2
The initialization parameters.

Notation	Meaning in our algorithm
density_i	The particle density of animation object *i* in Eq. (8)
V_t	The virtual animation driving type: *Gravity_type*, *VirtualMotion_type*, *Wind_type*, *Water_wave* in Eq. (1)
v_*	The speed of virtual wind in Eq. (6)
$\vec{v}$	The direction of virtual wind in Eq. (6)
Point_j	The reference point of Virtual Motion *j*

we initialize wind speed and wind direction in the scene. The parameters determine the amplitude of virtual motions. When the time-varying motion map for a still image is generated, we construct new animation frames by image synthesis method. Finally, we combine those frames into an animation video.

5.1. Frames interpolation based on CNN model

We optimize our model by a convolutional neural network (CNN) [8]. The DeepMotion CNN model [8] is used to eliminate incoherence between virtual motion frames by generating intermediate frames. The work [8] described the detail of how to train datasets and predict the intermediate frames. Giving two video frames, which are generated by the proposed method, we can generate the intermediate frame between them by DeepMotion CNN model [8]. And based on DeepMotion CNN model [8], we can do the quadratic interpolation between original and intermediate frames. The final frame interpolation results are shown as Fig. 8.

5.2. Wobbling animation driven by gravity

In our experiments, we set different *density* for the diving athlete to validate the effects of gravity on board wobble. We extract an initial frame (Fig. 7(a)) from a board diving video in social media. We set the *density* in the interval $[0.0 : 0.1 : 5.0]$. The density of diving board is a constant value at the same frame. Fig. 9 is the wobble displacement value of P_0 (shown in Fig. 7(b)) in diving board with different athlete densities. We can find out that the effect of the athlete gravity on diving wobble becomes smaller when the athlete density is in the interval $[0.0 : 1.0]$. The board displacement tends to be a constant with $density > 1.5$.

We select three density values $\{0.8, 1.4, 4.0\}$ for the athlete to valid our method. We also compare our animations with the original video (Fig. 10). The validity of the proposed model is discussed in Fig. 11. From the animations of wobble, we find that the displacement in animation is slightly smaller than ground-truth. Due to lack of the prior knowledge to simulate the wobble of board and athlete, we just use the gravity to drive the motions. However, the running force of athlete may be a key factor to cause the board wobble in actual situation. Due to $F\_s \gg F\_g$ (Fig. 7(b)), the wobble amplitude can not be greater than 0 which is one drawback of our method. Compared with the ground-truth, we perform error analysis on the wobbling amplitude values under different density, which is shown as Table 3. From Table 3, we get the minimum MSE (Mean Squared Error) and RMSE (Root Mean Squared Error) with $density = 0.8$, and get the minimum MAE (Mean Absolut Error) with $density = 4.0$. As the MAE can better reflect the actual situation of the prediction error, we infer that our method can simulate the wobbling most similar to the real situation with $density = 4.0$. The final offset displacements of board are almost the same as ground-truth, which shows that our method can effectively simulate the wobbling driven by the gravity of the athlete.

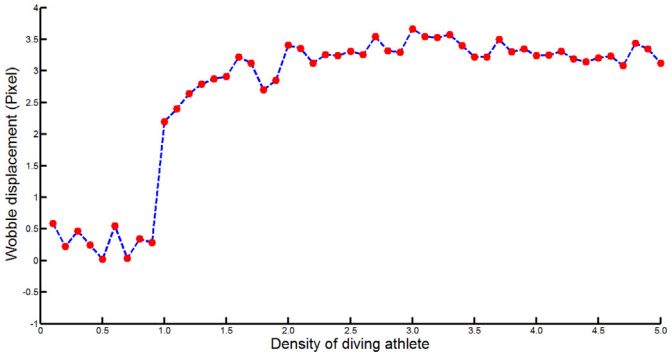


Fig. 9. Wobble displacement of P_0 (Fig. 7(b)) on diving board with different athlete densities.

We also use the proposed method to simulate the raindrop falling effects from a leaf (Fig. 12). We extract the animating objects in the original image firstly (Fig. 12(b)). Due to the power of gravity, the raindrop will fall down from the leaf, which can cause a slight jitter of the leaf at the same time. The animation result is shown as Figs. 12(c)-(i). From the animation video, we can observe a significantly downward acceleration of raindrop. Furthermore, we add a virtual wind on the raindrop to create a more intense jitter effect for the leaf. The wind force will also change the trajectory of the raindrop falling. The direction of virtual wind is set from right to left. The animation result driven by both gravity and wind force is shown in Fig. 13. From Fig. 13, we can confirm that the gravity and wind force can drive the an drifting trajectory with the wind of raindrop.

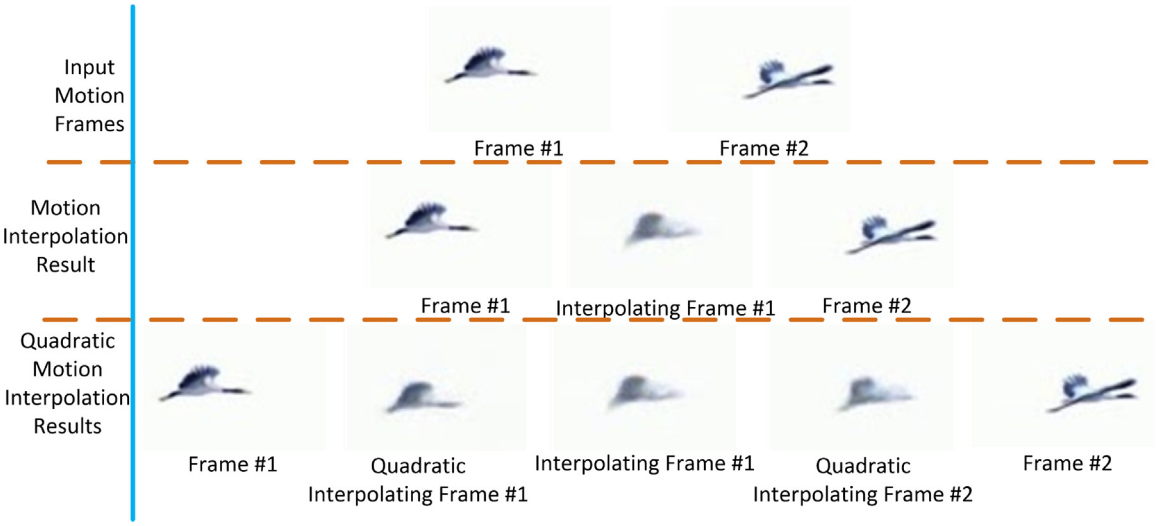
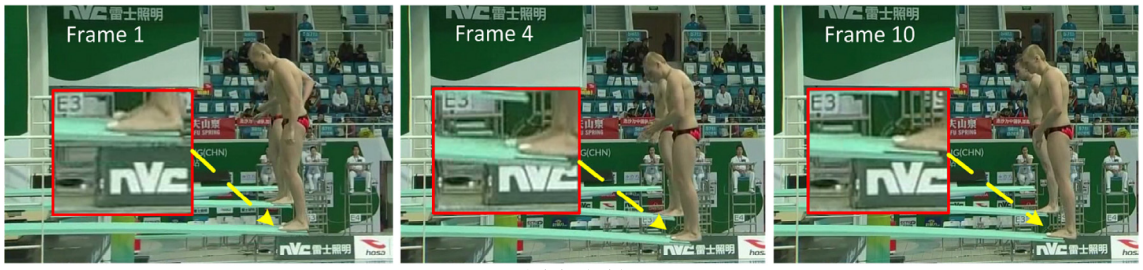
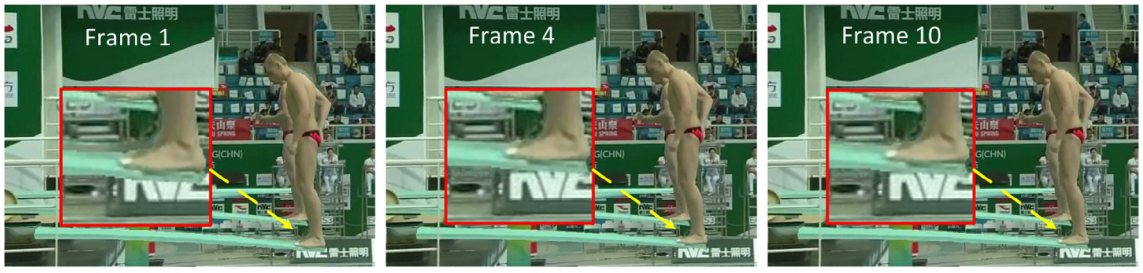


Fig. 8. The motion interpolation results based on DeepMotion CNN model [8]. The first line is the input video frames, the second line is the motion interpolation result based on DeepMotion CNN model [8], and the third line is the quadratic interpolation results based on DeepMotion CNN model [8].

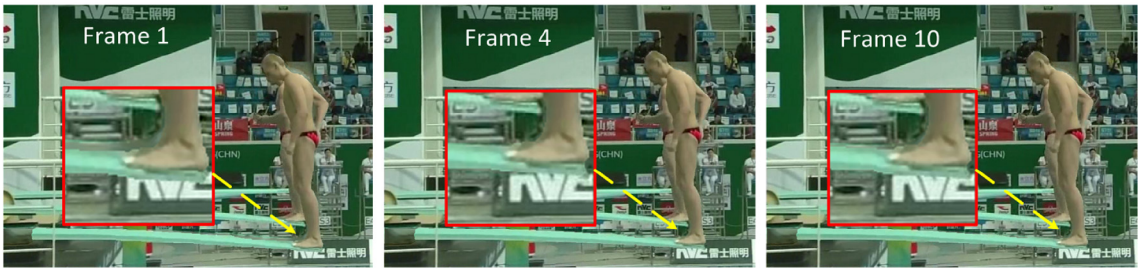


(a) Original video



(b) Wobbling animation with density 0.8

(c) Wobbling animation with density=1.4



(d) Wobbling animation with density=4.0

Fig. 10. Wobbling animation example. (a) is the original video, (b), (c) and (d) are the corresponding virtual motion videos generated by our model with different athlete values. Please click on the sub figures to watch the animation video in detail. (Please use Adobe reader to read our paper.)

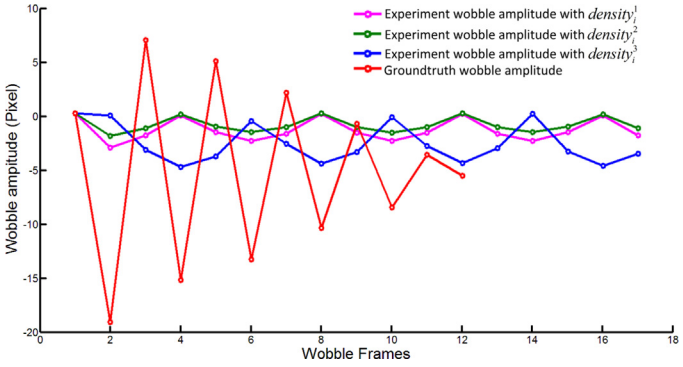


Fig. 11. Wobble amplitudes in wobble animation compared to Ground-truth.

Table 3
Error analysis of wobbling amplitude in wobbling animation with different density.

	MSE	RMSE	MAE
Wobbling amplitude with density = 0.8	84.28	9.18	4.47
Wobbling amplitude with density = 1.4	89.00	9.43	4.61
Wobbling amplitude with density = 4.0	87.04	9.33	4.24

5.3. Seagulls head shaking animation

For the seagulls standing on the trunk, we animate the trunk jittering based on seagull head shaking. We extract the head part of the seagulls from Fig. 2(a), and generate the head motion cycle after morphological preprocessing. Based on the motion cycle, we fuse the head and body of seagull together in accordance with

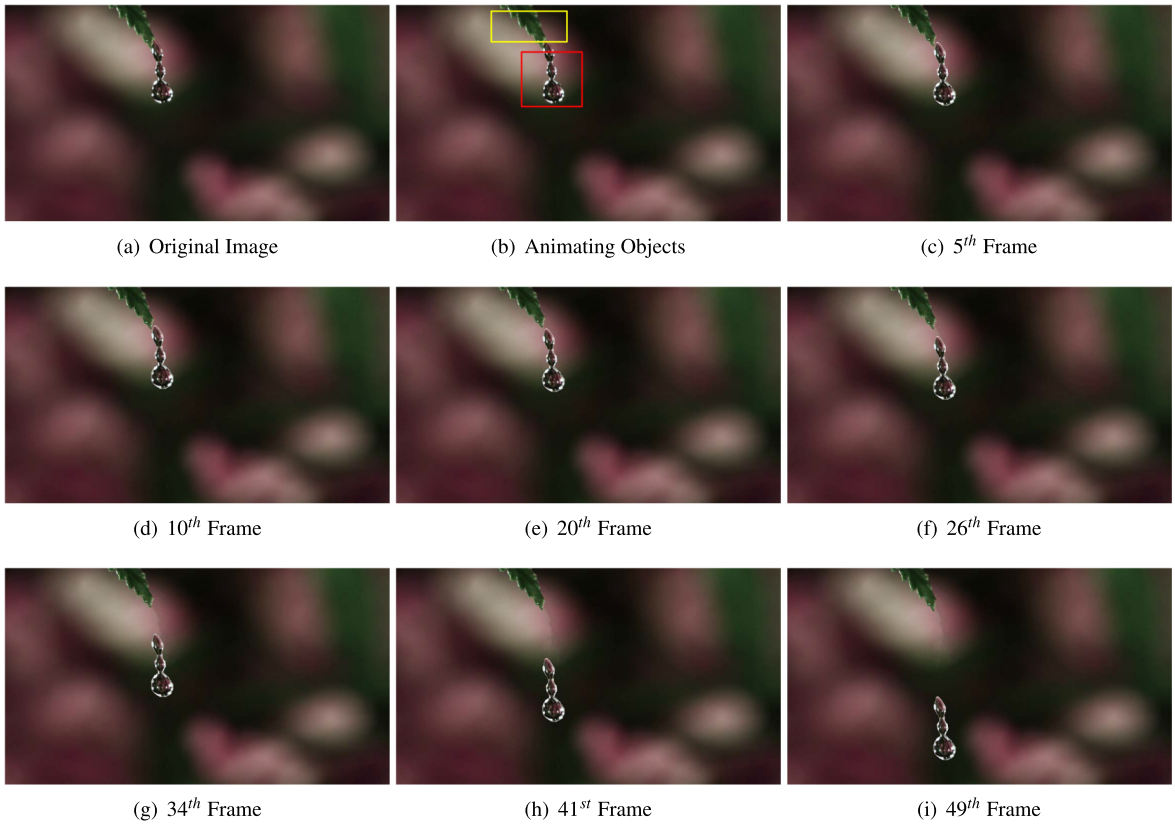


Fig. 12. Simulating a raindrop falling from a leaf.

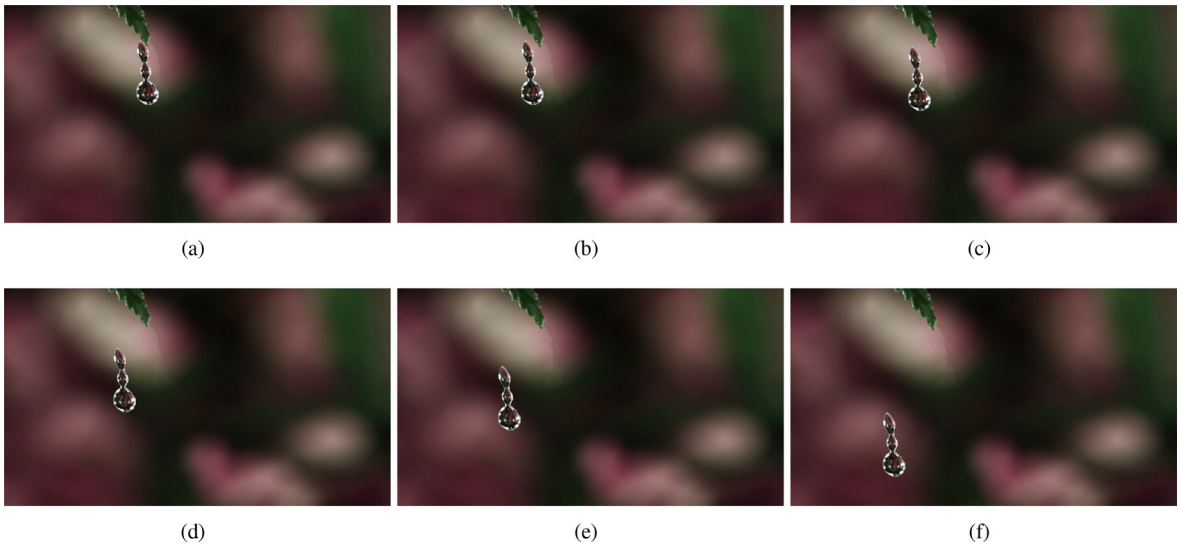


Fig. 13. Virtual animation frames of raindrop falling, which are driven by both gravity and wind force.

time. At each time, the virtual seagulls and trunk are assumed as a whole object which is driven by continuous head shaking motions in the image. There are two types of motions: seagull head shaking and trunk jittering driven by the motion of all virtual seagulls. Fig. 14 is an example for motion cycle of seagulls head shaking. From Fig. 14, there are regular and common-sense virtual motion in the video.

5.4. Wildebeests running

Fig. 15 is an example for animating wildebeests running in prairie from a single image. We first extract all wildebeests from the image, then the background of this image is repaired. We use method [3] to calculate motion similarity by matching shape context of each wildebeest to generate wildebeest

Fig. 14. Seagulls head shaking animation. (a) The original still image; (b)-(e) The seagulls are shaking their heads and the underlying trunk are jittering accordingly.

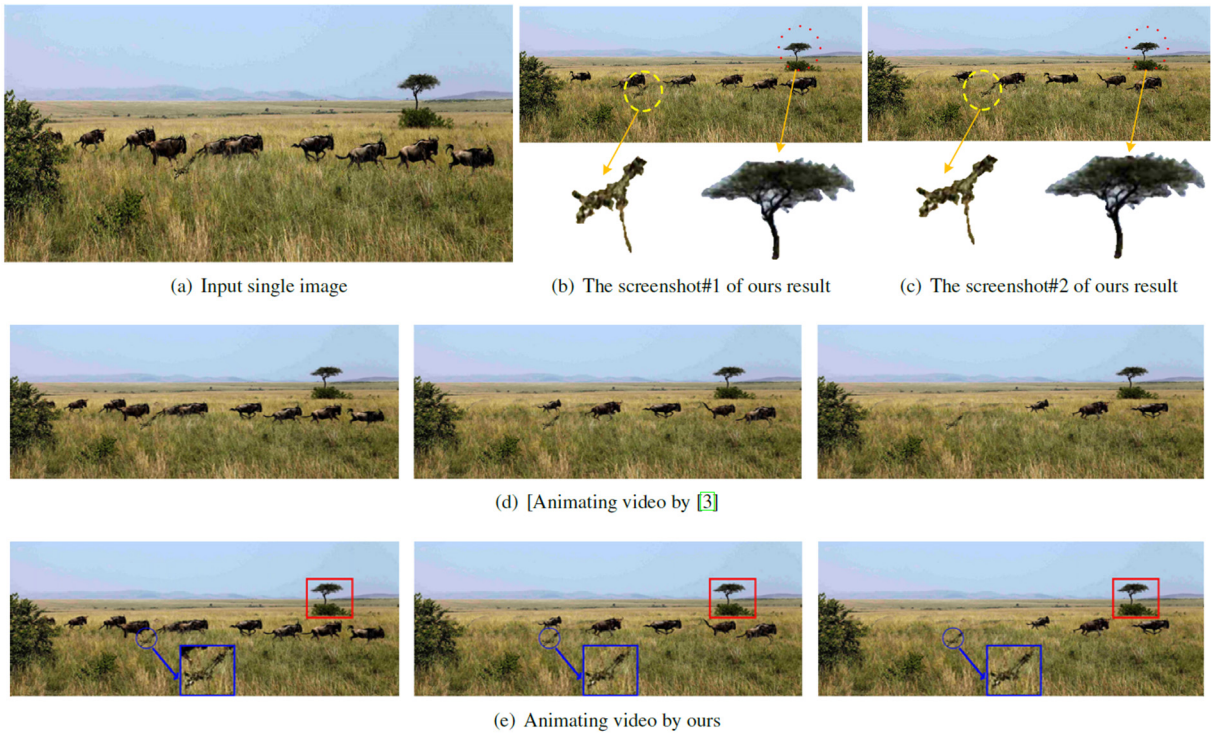


Fig. 15. Animating wildebeests running. (a) A wildebeests image from Internet; (b)-(c) are the wildebeests running animation frames from ours result; (d) is the animation video by [3], where the plants are not driven to motion; and (e) is the wildebeests running animation video with swaying plants, which is driven by wind force. Please click on the sub figures (d) and (e) to watch the animation video in detail. (Please use Adobe reader to read our paper.)

running order (Fig. 4). We initialize the running speed of wildebeest, which can construct a wildebeests running animation, as shown in Fig. 15(d).

Considering in the scene where a natural wind may blow, we initialize the velocity and direction of wind force in image. The velocity of virtual wind force is set from $[1m/s, \ldots, 5m/s]$, and the direction is set from left to right in the image. We extract one tree in distance scene and one sapling in near scene. Then we generate the time-varying 2D displacement maps for the tree and sapling using stochastic motion texture based on the initialization wind parameter. Fig. 15(b)-(c) are the motion frames which contain running wildebeests, the swinging tree and swaying sapling, and Fig. 15(e) is the animation video correspondingly. Compared with method [3], the proposed method can drive more objects by the wind force in the still image.

The validity of stochastic motion texture is discussed in Fig. 16. We employ the different wind velocities to calculate the displacement of the tree tip in Fig. 15(a). The result shows that a proportional relationship exists in displacement and wind speed. The effects of the wind gradually decline with time growth.

5.5. Wild geese flying and swimming animation

Finally, to show the practicability of our model, we employ the framework of our model to animate the water wave when the geese are swimming on calm surface. We also synthesize the

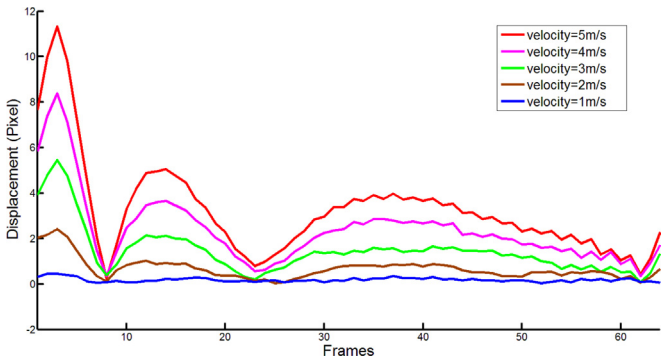
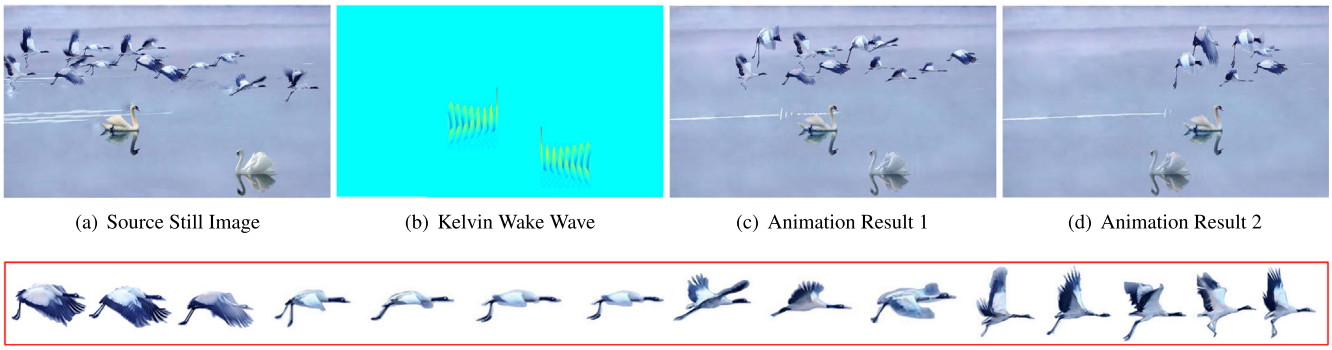


Fig. 16. The displacements of one leaf tip with different wind velocity.



(a) Source Still Image (b) Kelvin Wake Wave (c) Animation Result 1 (d) Animation Result 2

(e) Flying Motion list

Fig. 17. The animation of wild geese flying and swimming generated by our techniques. (a) A still nature image, (e) a flying order list of wild geese generated by our method, (b) the real time swimming wake waves of wild geese generated by our method, (c) and (d) the animating frame based on (e) and (b).

motion animation, which contains the flying and swimming wild geese, as shown in Fig. 17. We generate the wake waves of wild geese based on Kelvin wake pattern [30] firstly, which are shown as Fig. 17(b). We then generate the geese flying motion order list based on shape context firstly (Fig. 17(e)). Finally, we synthesize flying and swimming motion for each wild goose correspondingly (Figs. 17(c) and (d)), which are smoothed by DeepMotion CNN model [8].

5.6. Discussion

Table 4 presents total frames of each experiment with the our basic model and the optimization model. From the Table 4, we can infer that our model can generate enough virtual motion frames with quadratic motion interpolation.

We run all our experiments on a single PC, with 64-bit Window 10 system, Intel Core 3.4 GHz CPU and 4 GB RAM. Table 5 presents time cost of the main steps of each experiment. As illustrated in Table 5, interactive object extraction and virtual motion generation consume relatively more time. And then, we discuss the time complexity of those two steps as follows.

(1) Objects extracting: this step employs manual interaction to extract the animating objects from a still image. The time complexity of objects extracting is determined by the number of animating objects and qualification of interaction.
(2) Motion generation: there are two procedures in this step: the displacement propagation for each pixel of an animating object and motion cycle generation. Based on Eq. (14), the time complexity of displacement propagation is $O(s^3)$. The motion cycle generation can be divided into two parts: motion matching and matching matrix sorting. Based on Eq. (9), the time complexity of feature points matching is $O(C \cdot K)$. And the time complexity for one pair of motion matching is $O(N_f) \cdot O(C \cdot K)$, where N_f is the number of feature points of motion. The larger the size of matching motion shape, the more time consuming of motion matching based on shape context. As our model generates a matching matrix based on motion matching, the number of motion matching and the size of matching shape are the proportional factors of time cost. For example, there are 13 geese in Experiment 6, we must generate a 13×13 matching matrix, which need conduct 96 times of motion matching. The user interaction step for object extracting also takes some time.

Table 4
The Total Frames of Generation Animations.

Animation	Total Frames	Total Frames based on DeepMotion [8] Interpolation	Total Frames based on Quadratic DeepMotion [8] Interpolation
Wobbling	16	32	68
Raindrop falling based on gravity	20	50	100
Raindrop falling based on gravity and wind force	20	50	100
Seagulls head shaking	84	168	340
Wildebeests running	63	129	255
Wild geese flying and swimming	74	146	292

6. Conclusions

We have presented an approach for generating object animation from still images based on virtual motion driving, and effectively simulated the interaction between the active objects and their neighboring passive objects. We divide the image into *ForceObjects* and some static objects. The animation of *ForceObject* is

Table 5
The time consume of Generation Animations.

Animation	Image Size	Total Driven Objects	Objects Extracting	Force Analysis	Virtual Motion Generation	Animation Rendering
Wobbling	1280 × 720	2	4 min	0.2 s	88 s	36 s
raindrop falling from a leaf driven by gravity	768 × 432	2	4 min	0.3 s	214 s	19 s
raindrop falling from a leaf driven by gravity and wind force	768 × 432	2	4 min	0.4 s	217 s	19 s
Seagulls head shaking	1300 × 870	5	10 min	1.2 s	328 s	24 s
Wildebeests running	2200 × 877	8	18 min	0.2 s	8740 s	22 s
Wild geese flying and swimming	1000 × 644	17	35 min	0.8 s	2282 s	35 s

driven by virtual force or similar motion posture, and the animation of other objects is driven by the corresponding *ForceObject*. We employ time-varying displacement maps, which are generated by harmonic oscillation model, to determine the shift position and posture of the animating objects in each frame. The shape context is applied to create the posture shift map (motion cycle), which determines the next posture of other static objects. In order to smooth the generated video, we use the DeepMotion CNN model [8] to interpolate more intermediate frames between each pairs of generated video frames. Finally, the virtual motions are synthesized according to the layer order. We have animated different kinds of scenes to validate the effectiveness of the proposed method.

In the future, we will intend to retrieve the motion actions from the image set to construct more delicate models to animate more complex scenes. To transfer the motion of the 3D moving object (or the moving object in the video) to the elements in the image is also an interesting work, which will improve the quality of the motion.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors would like to thank the anonymous reviewers for their constructive comments. This work was supported by The Key Technological Innovation Projects of Hubei Province under Grants 2018AAA062, and Wuhan Science and Technology Plan Project under Grant 2017010201010109, and The National Key Research and Development Program of China under Grant 2017YFB1002600, and National Natural Science Foundation of China under Grants 61672390, 61562025, 61972298, 61962019.

References

[1] H. Averbuch-Elor, D. Cohen-Or, J. Kopf, M.F. Cohen, Bringing portraits to life, ACM Trans. Graph. (Proc. SIGGRAPH Asia 2017) 36 (2017) 196.
[2] Y.-Y. Chuang, D.B. Goldman, K.C. Zheng, B. Curless, D.H. Salesin, R. Szeliski, Animating pictures with stochastic motion textures, ACM Trans. Graph. 24 (2005) 853–860.
[3] X. Xu, L. Wan, X. Liu, T.-T. Wong, L. Wang, C.-S. Leung, Animating animal motion from still, ACM Trans. Graph. 27 (2008) 32–39.
[4] N. Kholgade, T. Simon, A. Efros, Y. Sheikh, 3d object manipulation in a single photograph using stock 3d models, ACM Trans. Graph. 33 (2014) 1–12.
[5] W.C. Jhou, W.H. Cheng, Animating still landscape photographs through cloud motion creation, IEEE Trans. Multimedia 18 (2016) 4–13.
[6] Y. Nie, H. Sun, P. Li, C. Xiao, K. Ma, Object movements synopsis via part assembling and stitching, IEEE Trans. Visual Comput. Graphics 20 (2014) 1303–1315.
[7] K. Olszewski, H. Li, Z. Li, C. Yang, Y. Zhou, R. Yu, Z. Huang, S. Xiang, S. Saito, P. Kohli, Realistic dynamic facial textures from a single image using gans, in: IEEE International Conference on Computer Vision, 2017, pp. 5439–5448.
[8] D.W. Neil Joshi, Deep motion: A convolutional neural network for frame interpolation, https://github.com/neil454/deep-motion/, 2017.
[9] Z. Liu, R.A. Yeh, X. Tang, Y. Liu, A. Agarwala, Video frame synthesis using deep voxel flow, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 4463–4471.
[10] S. Niklaus, M. Long, F. Liu, Video frame interpolation via adaptive convolution, in: Computer Vision & Pattern Recognition, 2017.
[11] Z. Xu, Q. Zhang, Z. Cao, C. Xiao, Video background completion using motion-guided pixel assignment optimization, IEEE Trans. Circuits Syst. Video Technol. 26 (2016) 1393–1406.
[12] M. Aittala, T. Aila, J. Lehtinen, Reflectance modeling by neural texture synthesis, ACM Trans. Graph. 35 (2016) 1–13.
[13] B.Y. White, Designing computer games to help physics students understand newton's laws of motion, Cogn. Instruct. 1 (1984) 69–108.
[14] I. Zachevsky, Y.Y. Zeevi, Statistics of natural stochastic textures and their application in image denoising, IEEE Trans. Image Process. 25 (2016) 2130–2145.
[15] K. Chen, H. Johan, Animating 3d vegetation in real-time using a 2d approach, in: Proceedings of the 19th Symposium on Interactive 3D Graphics and Games, 2015, pp. 69–76.
[16] M. Okabe, K. Anjyor, R. Onai, Creating fluid animation from a single image using video database, Comput. Graph. Forum 30 (2011) 1973–1982.
[17] P.-S. Chen, S.-K. Wong, W.-C. Lin, Two-dimensional digital water art creation on a non-absorbent hydrophilic surface, in: Proceedings of IEEE International Conference on Multimedia and Expo, 2015, pp. 1–6.
[18] M. Sun, A.D. Jepson, E. Fiume, Video input driven animation (vida), in: Proceedings of the Ninth IEEE International Conference on Computer Vision, vol. 2, 2003, pp. 96–103.
[19] H. Yasin, U. Iqbal, B. Kruger, A. Weber, J. Gall, A dual-source approach for 3d pose estimation from a single image, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4948–4956.
[20] E. Jain, Y. Sheikh, M. Mahler, J. Hodgins, Augmenting hand animation with three-dimensional secondary motion, in: Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2010, pp. 93–102.
[21] E. Jain, Y. Sheikh, M. Mahler, J. Hodgins, Three-dimensional proxies for hand-drawn characters, ACM Trans. Graph. 31 (2012) 1–16.
[22] X. Liu, D. Zhao, J. Zhou, W. Gao, H. Sun, Image interpolation via graph-based bayesian label propagation, IEEE Trans. Image Process. 23 (2014) 1084–1096.
[23] N.C. Tang, C.T. Hsu, M.F. Weng, T.Y. Lin, Example-based human motion extrapolation and motion repairing using contour manifold, IEEE Trans. Multimedia 16 (2014) 47–59.
[24] Y. Romano, M. Protter, M. Elad, Single image interpolation via adaptive nonlocal sparsity-based modeling, IEEE Trans. Image Process. 23 (2014) 3085–3098.
[25] B. Ding, C. Long, L. Zhang, C. Xiao, Argan: attentive recurrent generative adversarial network for shadow detection and removal, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 10213–10222.
[26] L. Zhang, Q. Zhang, C. Xiao, Shadow remover: Image shadow removal based on illumination recovering optimization, IEEE Trans. Image Process. 24 (2015) 4623–4636.
[27] G. Mori, S. Belongie, J. Malik, Efficient shape matching using shape contexts, IEEE Trans. Pattern Anal. Mach. Intell. 27 (2005) 1832–1837.
[28] C. Xiao, M. Liu, D. Xiao, Z. Dong, K.-L. Ma, Fast closed-form matting using a hierarchical data structure, IEEE Trans. Circuits Syst. Video Technol. 24 (2014) 49–62.
[29] S. Darabi, E. Shechtman, C. Barnes, D.B. Goldman, P. Sen, Image melding: combining inconsistent images using patch-based synthesis, ACM Trans. Graph. 31 (2012) 13–15.
[30] G. Zilman, A. Zapolski, M. Marom, On detectability of a ship's kelvin wake in simulated sar images of rough sea surface, IEEE Trans. Geosci. Remote Sens. 53 (2015) 609–619.