Interactive authoring of bending and twisting motions of short plants using hand gestures

Kan Chen[1,2], Henry Johan[2]

[1]Nanyang Technological University

[2]Fraunhofer IDM@NTU

50 Nanyang Avenue, NS1-1, Level 5, Singapore, 639798

Tel. (+65)65922671 Fax. (+65)67928123

Email. kchen1@ntu.edu.sg, hjohan@fraunhofer.sg

Abstract

In this paper, we propose an approach to interactively author the bending and twisting motions of short plants using hand gestures, especially suitable for grass, flowers and leaves. Our method is based on the observations that hand motions can represent the bending and twisting motions of short plants and using a hand to describe motions is natural and proficient for human. We therefore use a hand as a 'puppet' to author the animation of one single short plant based on transferring the motions of a hand to the motions of a short plant. We first author the global motions of the short plant followed by the motions of its elements such as leaves and flowers. We also propose a framework to utilize the animation results to animate a field of short plants and further adjust the motion effects according to the properties of the short plants, such as rigidity. As a result, users can intuitively and rapidly author and generate their desired motions of short plants under the influence of external forces. Especially, our method is accessible to non-expert users and suitable for fast prototyping and authoring specific motions of short plants such as in cartoons.

Keywords: short plants, animation authoring, hand gestures

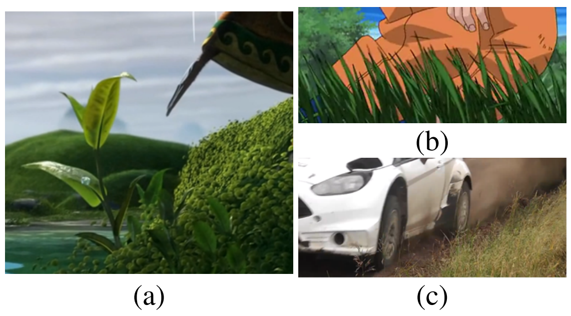


Figure 1: Our method is suitable for scenarios which require designing and simulating specific motions of short plants, such as in cartoons (a), (b) and modeling the motions of short plants under the influence of a passing character (b) or car (c, from www.youtube.com). Cartoon frame (a) is from animation “Kung Fu Panda 2” (©DreamWorks Animation), (b) is from “Naruto” (©Masashi Kishimoto/Hayato Date/Shueisha/Studio Pierrot/TV Tokyo/Viz Media).

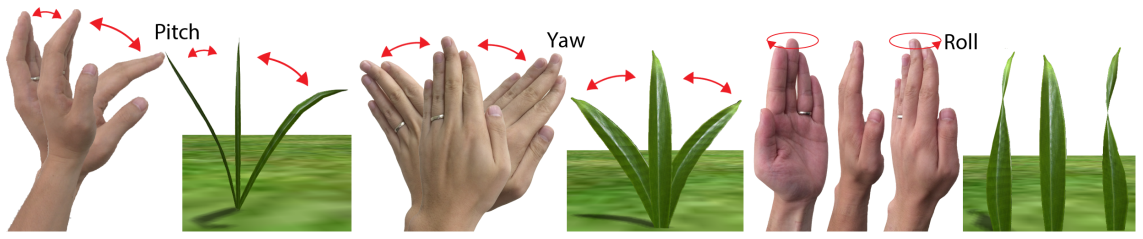
1 Introduction

Vegetation (a field of plants) animation has many applications such as in games, movies, and cartoons. Short plants, such as grass, flowers, and shrubs, are common vegetation species and can be found in many scenes such as grass fields, prairies, gardens, and forests. Under the influence of external forces, short plants usually exhibit bending and twisting motions. Simulating a dynamic and believable scene with short plants, such as a meadow with grass and flowers swaying in wind, provides users with immersive feelings. Moreover, designing and simulating specific motions of short plants are required in many scenarios, especially in cartoons and animation movies where artistic (non-physically realistic) motions are required, such as the examples in Figures 1 (a) and (b). The motions of short plants reflect the character’s actions and the story, thus these specific motions should follow the animator’s design ideas. Furthermore, under the influence of an object, certain forms of pattern, pace, and trajectory are usually expected in the motions of short plants. For example, when a car passes, short plants may suddenly sway crazily and then gradually recover (Figure 1 (c)). This is an important scenario in applications such as driving simulator and racing game. The viewers may feel uncomfortable if the short plants do not react according to the passing

object's motions. Efficiently authoring the motions of short plants is important:
(1) For non-expert users, such as when teaching animation to primary pupils, it would be beneficial if they are able to intuitively and efficiently produce the motions of short plants that they observe in real-life such as a swaying grass in wind or they imagine in mind, e.g. a dancing flower following the music.
(2) For users to conduct fast prototyping, it is important to rapidly realize or record their design ideas. Later at the production stage, the users can further adjust the details based on the prototype.

In these applications, intuitively and efficiently realizing visually plausible motion effects of short plants is more important than the physical accuracy. Unfortunately, it can be difficult to achieve this using existing techniques. In the conventional animation pipeline, the rigging stage establishes a control structure (handles) for the 3D models of short plants, then the animation stage generates poses of the rigged models by computing the shape based on the transformations of the handles. These transformations of handles can be obtained from manual posing at many key frames, procedural methods or physical simulations.

Manual posing requires tedious work and art skills to move the handles through the traditional mouse and keyboard interfaces. Procedural methods can achieve visually plausible animations, however it is not always easy to design a procedure for a specific motion. Existing physics-based methods can generate physically realistic animations, however they usually require a high computational cost. For many applications such as real-time animation editing, there are usually many important computations need to be performed at the same time, such as rendering, character animation and fluid simulation, so it is helpful to lower the computational cost of plant animation while still making the animation looks visually plausible. Moreover, both the procedural and physics-based methods usually require setting a number of parameters. Additional knowledge and tedious tuning are usually required to relate these abstract settings with the actual desired effects. Therefore, they are inconvenient and unintuitive. For instance, in the "Kung Fu Panda 2" (©DreamWorks Animation) example (Figure 1 (a)), it is difficult to generate the artistic motions using the procedural and physics-based methods. In the passing car example (Figure 1 (c)), it is in general a challenging task to produce the characteristics of the desired motions. Performing simulation of



(a) (b) (c)

Figure 2: A hand can bend and roll which can represent the essential bending and twisting motions of short plants. Thus, it is natural to use a hand to author the motions of short plants. The followings are authoring examples of a grass blade. (a) Bending a hand forward and backward: grass bending forward and backward (pitch bending). (b) Bending a hand left and right: grass bending left and right (yaw bending). (c) Rolling a hand: twisting of grass.

the wind generated by the car and its interactions with short plants as well as obtaining a number of properties (e.g. the mass of a blade) and tuning a number of parameters require tedious efforts. Another approach is by using motion capture data. However, in general, it is tedious to setup the motion capture system (e.g. markers and sensors) for short plants and process the captured data. In this paper, we demonstrate that the animation of short plants can be authored in a simpler and more efficient manner.

The goal of our research is to provide users with an intuitive interface for effective transition from user's targets or design ideas to actual animations. Instead of physical accuracy, our method focuses on creating essential visual effects of short plants which are perceived as either natural looking or artistic by viewers. We tackle this problem based on the following observations and our real-life experiences:

(1) Under the influence of external forces, short plants usually exhibit bending and twisting motions. Similarly, a hand can perform bending and rolling which can sufficiently represent the bending and twisting motions of a short plant (Figure 2). This is because mobility-wise, different from tall plants, short plants usually have lower degrees of freedom, which makes it possible for a hand (consists of wrist and fingers) to imitate the motions of a short plant.

(2) Human is proficient in using a hand to mimic motions: when we are describing the mo-

tions of short plants to others, we naturally use hand gestures, for example we usually use our bending hand to describe a bending grass blade (Figure 2(a)).
(3) In recent years, hand gesture based controller devices have become widely available, such as [1] and [2], which enable us to interact with a computer using our hands and fingers.

Many 3D hand gestural user interfaces have been developed to help us to perform our daily computer tasks such as switching between slides by swiping a hand as well as scrolling a page by moving a finger up and down. In these interfaces, a hand is used as a tool to trigger commands. Different from these interfaces, we demonstrate that a hand also can be used as a 'puppet' to model and mimic a motion. In this paper, we employ the acting capability of a hand and propose a natural and intuitive interface to interactively author animation of short plants. Our method has the following features:
(1) We propose a novel hand gestural interface, facilitating rapid authoring of the animation effects that short plants usually exhibit: bending and twisting. Our approach does not require tedious effort to set the poses of short plants at many frames and physical simulations which usually require high computational cost. Our authoring process usually takes only less than one minute.
(2) Our approach transfers the motions of a hand to the motions of a plant element (grass blade or leaf). Thus, our approach allows users to create their desired animation of short plants by acting or mimicking the motions using a hand as well as easily control the characteristics of the motions on the fly. Our method is not simply a straightforward direct mapping. We consider the limited (one) degrees of freedom the finger joints usually have, the user's ability to bend his/her wrist and fingers, and the ease of use. It allows users to efficiently author short plants bending to any directions and recovering as well as smoothly bending from one direction to another.
(3) We propose a two-level coarse-to-fine approach to author the animation for one short plant, e.g. users can first author the animation for the whole plant (e.g. one flower), then for its secondary elements (e.g. the flower's leaves). An implicit animation alignment method is also introduced to align the animations between secondary elements and whole plant.
(4) We propose an efficient framework to reuse, refine, and control the authored animation results. Users only need to create the animation for one short plant, the results can

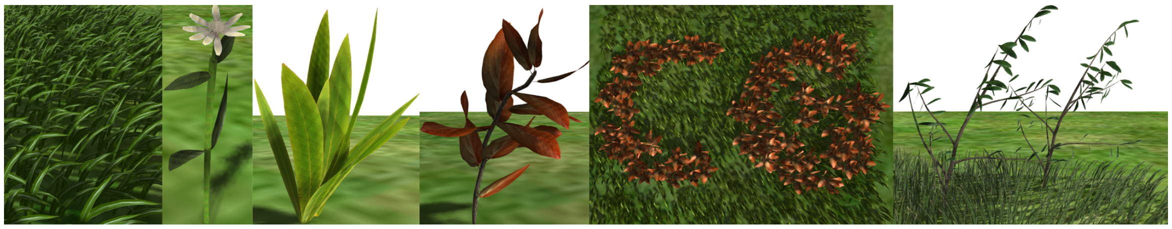


Figure 3: Our method can be used to animate (from left to right) grass, flower, shrub (two examples), a scene of short plants (example of a 'CG' scene), and a scene of grass and short trees.

then be applied to animate a field of short plants. To simulate a controllable dynamic field, we propose to utilize the authored animation results by incorporating an animation looping technique and a motion propagation method. Users can further adjust the motion effects according to the properties of the short plants.
(5) We also propose a semi-automatic method by authoring only the overall (root) motions of the plant element by a hand and using them to guide a procedural method to automatically synthesize the motions of its other parts. This is an option suitable for users who only want a high-level design of the overall motions of the short plants, omitting the design and control of the detailed motions, such as modeling swaying grass in natural wind.

Our method is suitable to simulate one short plant or one field of short plants, such as grass and flowers, under the influence of external forces (Figure 3). Especially, it is helpful when simulating specific motions of short plants is required, such as in cartoons. Our method has a similar target as a concurrent work [3], which is the easy accessibility to non-expert users. Moreover, it is helpful as a practical tool for users to illustrate their ideas and conduct fast prototyping. Most importantly, it opens the door of developing interactive hand gestural interfaces that is based on using a hand to perform acting jobs instead of issuing commands. In this paper, we introduce this expressive design paradigm in the domain of vegetation animation.

2 Related work

2.1 Animation of plants

Physics-based methods: In these methods, explicit or simplified integration for equations of motion is usually applied on the joints of the underlying skeletal structure. Shinya and Fournier [4] introduced modal analysis to animate plants. Stam [5] proposed to perform the simulation in the frequency domain. The spectral method is employed to reduce the computational cost. The spectral method is also used in [6], where they proposed to animate plants using a motion spectrum based on a damped harmonic oscillator. Habel et al. [7] proposed to use a similar spectral method as in [6] and beam deformation to create physics guided animations for trees. Diener et al. [8] computed a wind basis using modal analysis and projected directional wind at run time. Akagi et al. [9] performed an explicit fluid simulation to simulate the wind effect. Zhao and Barbič [10] presented a method to process plant model for a physics-based simulation. Furthermore, Weber [11] used a cloth simulation approach to animate plants. Spring simulation is also used to model grass animation [12, 13].
Procedural methods: In these methods, the appearance of tree motions are usually heuristically modeled based on procedural functions [14–16]. Trigonometric and noise functions have been widely used to simulate the wind effect [17, 18]. Chen and Johan [19] proposed to apply continuum simulation to model grass animations. Chen and Johan [20] proposed to animate vegetation in 2D images by combining 2D fluid simulation, wave simulation, and grid-based image warping. This method was also extended to animate vegetation in 3D scenes using view-dependent 2D billboard layers [21]. SpeedTree [22] is the state-of-the-art solution for games and movies. Users have to set some procedures in SpeedTree to generate the wind effect, for example, setting the frequency to control the oscillating speed of leaves.
Data-driven methods: In these methods, the plants are animated based on pre-computed or pre-captured motion data. The major methods include: pre-computed motion graphs [23–26], motion capture methods [27], and 2D video-based methods [28–30].
Summary: Physics-based methods [7, 10] can achieve physically realistic animations, however they usually lack intuitive direct control and require a high computational cost. Procedural methods [14, 18] can achieve natural looking animations, however it is in general

difficult to design a procedure to achieve a given specific effect. Both the physical and procedural methods require to set and tune a number of parameters (some are not intuitive), unfortunately it is not always easy to relate these parameters to the actual effects. SpeedTree [22] can achieve visually realistic animations for many types of plants, ranging from grass to big trees. However, SpeedTree also requires the users to set and tune a number of parameters. Different from existing techniques, in this paper, we demonstrate that the animation of short plants can be created in a more intuitive and efficient manner. In data-driven methods, physical simulations or procedural methods need to be performed in order to pre-compute the data, such as constructing a motion graph [25]. Generating motion data from a 2D video may lack some 3D information [29, 30]. Motion capture plants in general require tedious effort to setup the system and process the captured data [27]. In contrast to the existing methods, our method provides users with an intuitive interface based on hand gestures for rapid 3D animation authoring of short plants based on user's design ideas. Using the proposed system, users can easily and interactively generate their desired bending and twisting motions of short plants.

2.2 Authoring and controlling motions of articulated objects

Animating an articulated model with skeletal structural is a time-consuming task [31]. Motion capture methods can be used to animate skeletal characters, however it usually requires motion re-targeting which includes complex computation [32]. Computer puppetry methods [33–35] can directly control and animate a character using a specific input device. Most recently, Jin et al. [3] proposed algorithms to map and transfer motions between skeletons. However existing methods mainly focus on character motions. For indirect mapping, it requires to select features and map them to another character rig with different structure. Hecker et al. [36] proposed to pre-record skeletal motions and map them to specific characters with different morphologies. Lockwood and Singh [37] used finger touches to generate motion data to control leg movements. Seol et al. [38] proposed to puppet characters with a different topology. Coarse-to-fine design [39, 40] is a common approach to create animations. Dontcheva et al. [41] proposed to use layered multiple passes of acting to create and edit character animations. Their layered acting is similar to our proposed framework, our framework can be considered as consisting of multiple layers. Different from their approach, we also focus on the interaction between layers, e.g. the root influences the upper parts. Rhodin et al. [42] also proposed a method to map different input motions to different character motions, such as mapping a moving human arm to a running horse. Oshita et al. [43] proposed to use hand motions to control human character animations. The mappings from the captured motions to the character motions are not intuitive. In contrast to the existing methods, our method is based on the observation that it is natural to use hand gestures to describe the motions of short plants, hence we can transfer hand motions to plant motions while still maintaining the general intuitiveness.

2.3 Hand gestures

Hand motion capture devices have become widely available. Hand gestures are important interaction techniques. Please refer to [44] for a survey.
The major methods to capture hand motions include: marker-based methods [45], glove-based methods [46], and image and depth sensor based methods [1]. The main applications of hand motions are:
(1) Triggering commands: Hand gestures can be used to triggering commands such as moving fingers to scrolling web pages or swiping a hand to switch through open browser tabs [1].
(2) Manipulation: Hand gestures such as grasping can be used to performing tasks such as moving objects [47]. They can also be used to play instruments, such as a guitar [48], using complex positioning of fingers.
The existing applications of hand motions are mainly for triggering commands and manipulating objects. In contrast to the existing applications, in this paper the acting capability of a hand is employed. We use a hand as a ‘puppet’ to act the desired motions of plants. Since human are proficient in using a hand, using a hand to mimic the motions of short plants can the motions of plants can be efficiently realized.

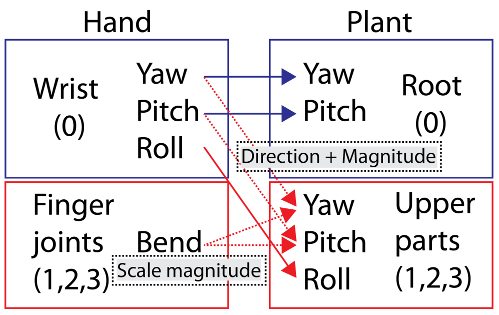


Figure 4: Transferring the motions of a hand to a short plant. The pitch and yaw motions of the wrist are transferred to the pitch and yaw motions of the root (joint 0) of the grass. They also determine the direction of bending for the upper parts (joints 1, 2, 3) of the grass. The roll motion of the wrist is transferred to the twisting of the upper parts. Bending angles of the finger joints control the magnitudes of the bending for the upper parts of the grass.

3 Overview of the proposed system

3.1 Basic idea

Given a 3D model of a short plant, we aim to create animation effects (bending and twisting) that users desired: grass, flowers and shrubs swaying under the influence of the external forces such as a passing object or natural wind. We focus on authoring these effects at the animation stage in the conventional animation pipeline. Similar to the global motions of a short plant, a hand can perform bending and rolling. Unlike tall plants, short plants have lower degrees of freedom, therefore using a hand can sufficiently represent the global motions of a short plant. Our authoring approach transfers the motions of a hand to the motions of a short plant (Figure 4).

We use the grass blade example in Figure 7 to introduce our basic idea. A wrist has three degrees of freedom (pitch, yaw and roll, refer to Figure 2), the pitch and yaw motions of the wrist are transferred to the pitch and yaw motions of the root (joint 0) of the grass, that means the grass can bend towards any directions. The roll motion of the wrist is transferred to the twisting of the upper parts (joints $1 \ldots 3$) of the grass, that means the upper parts of the grass can twist clockwise or counter-clockwise.

Note that the root and the upper parts of the grass, each have three degrees of freedom (pitch, yaw and roll). A straightforward mapping directly maps the wrist to the root and the finger joints to the upper parts of the grass. This is not a good solution, since the finger joints usually have only limited (one) degrees of freedom. In our system, all the joints of a finger have one degree of freedom (bend inwards to the palm and recover, refer to Figure 8). We propose to utilize this one degree of freedom: we let the bending angle of the finger joints to control the magnitudes of the bending for the upper parts of the grass. We make the direction of bending of the upper parts of the grass to be determined by the pitch and yaw motions of the wrist. This means the direction of the upper parts is the same as its root. This utilization enables the grass to bend to any directions. Moreover, connecting root with its upper parts also enables the grass to smoothly bend from one direction to another which enhances the ease of use of the system. When designing our system, we also take into account the users' abilities to move their joints. As such, the motions are fully controllable by the users and various natural looking motion effects of short plants can be achieved. Additionally, we propose to use root motions of the grass to guide a procedural animation method to automatically generate the motions of its upper parts. This is suitable for conducting a high-level authoring of the overall motions of the grass.

We can further generalize our model from a simple blade to a more complex short plant such as a flower (Figure 7), since it also has the same essential motion effects: bending and twisting. We treat the whole short plant (flower) as one element and author the global animation. We then work on the secondary elements (leaves) of the short plant and author the secondary animations. These two levels of animations are superimposed to create the final animation. Introducing secondary motions will add richness to the overall animation effects. Moreover, we propose this two-level approach to make the task in each step simpler. Furthermore, if users only want a simpler motion, they can choose to perform only one level authoring. We also introduce a method to align the secondary animations with the global motion.

Because different parts of a vegetation field have similarities, we propose to utilize the authored animation results of one short plant to animate the whole field. To realize the motion effects generated by external forces (such as passing objects or natural wind), we propose to use a motion propagation method base on backtracking in time. The tracking

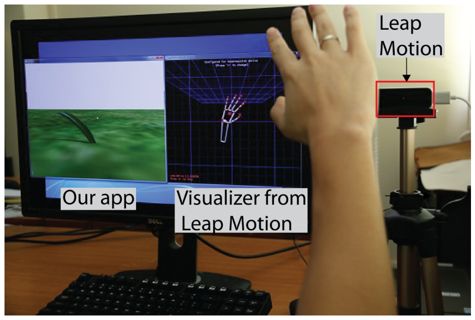


Figure 5: Our system setup.

path is determined via an animation looping technique. Users can further control and adjust the motion effects according to the properties of the short plants. The backtracking in time is also used to model the rigidity of motions.

3.2 Steps

Our system (Figure 5) consists of a PC with a monitor and a hand tracking device from which we can get the information of a hand: the bending angles of the finger joints and wrist. In our implementation, we choose to use Leap Motion controller with its SDK. As shown in Figure 6, we first set up the rig for the input 3D short plant model manually (Section 4). Then, during the animation stage, we perform the authoring of animation for one single short plant using hand gestures (Section 5). This authoring process is performed for the global animation first. This step may be repeated for the secondary elements (leaves). After authoring, the results are used to animate all the short plants in a field (Section 6).

4 Animation model

As shown in Figure 7, for the underlying animation model, we use a skeletal structure to model a grass blade (simple short plant). For a plant with a more complex structure (a complex short plant, such as the flower in Figure 7), we consider it as one trunk and use a skeletal structure to model the whole short plant. Its secondary elements (leaves) are treated as simple short plants. Since the underlying model is very simple, we can easily set

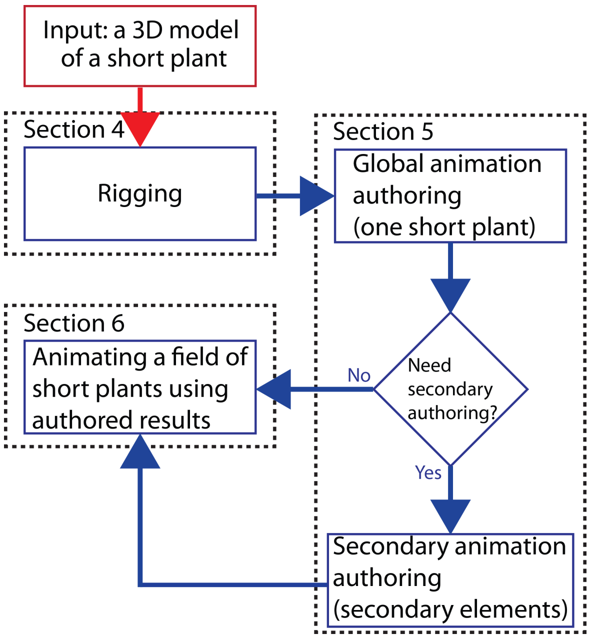

Input: a 3D model of a short plant
Section 4
Rigging
Section 5
Global animation authoring (one short plant)
Section 6
Animating a field of short plants using authored results
No
Need secondary authoring?
Yes
Secondary animation authoring (secondary elements)

Figure 6: The steps. Dotted boxes: the corresponding sections in the paper. We first do the rigging for the input 3D short plant model manually (Section 4). Then, we perform the animation authoring for one single short plant using hand gestures (Section 5). This authoring process is performed for the global animation first. This process may be repeated for the secondary elements (leaves). After authoring, the results of one short plant are utilized to animate the whole field (Section 6).

the skeleton manually using an interactive 3D modeling tool such as [49]. The number of joints is set as the number of joints of the principle finger plus one for the wrist. In our implementation, we use a thumb as the principle finger (we will explain the reasons later in the next section), and it contains three joints. So in our skeleton articulated model, there are four joints in total.

For the 3D deformation, we adopt the most common linear blend skinning (LBS) method. The articulation parameters are affine transformations, each of them is associated with a bone as the handle. For a vertex v in the skin local space, its updated position v' in the world space is computed as:

$$v' = \left(\sum_{i=0}^{n-1} w_i M_i \right) v, \tag{1}$$

where M_i is the affine transformation matrix for joint i, and w_i ($i = 0, \ldots, n-1$) represents how much influence the i-th joint has on vertex v. n is four in our implementation, 0 is the root (bottom) joint and 3 is the top joint. We define a simple linear method to assign the weights for the vertices, please refer to Appendix. The weights are computed based on the distances from the vertex to the joints. Alternatively, the skinning weights also can be painted through an interactive 3D modeling tool such as [49].

5 Animation authoring of a short plant using hand gestures

We design the following authoring interface based on the transferring the motions from a hand to a short plant. We focus on the two essential motions of short plants: bending (Section 5.1) and twisting (Section 5.2). The additional semi-automatic method is introduced in the next section. An authoring is performed using one hand, while another hand is also

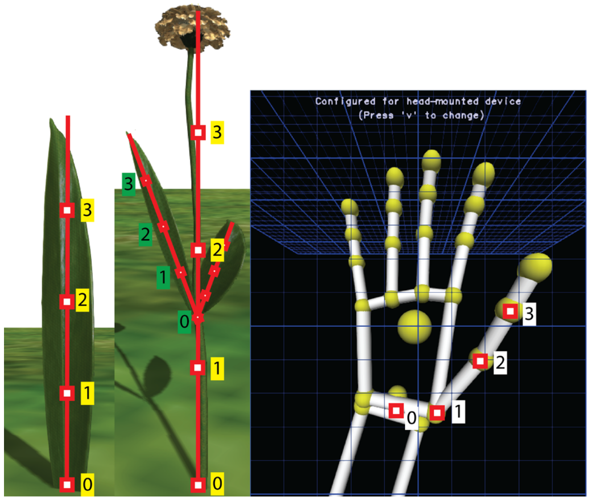


Figure 7: The underlying animation models for a grass blade and a flower. Their joints are labeled with respect to the joints of a hand. The yellow labels show the joints of the grass blade and the whole flower. The green labels show the joints of a secondary element (leaf) which is considered as a simple short plant.

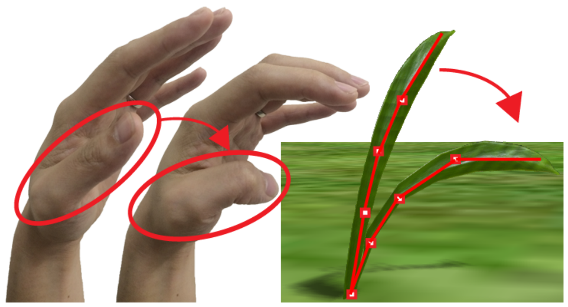


Figure 8: Bending the upper parts of a short plant.

involved for triggering the keys to start and end the recording of the authoring. We record the rotational motion data $\vec{P_i}$ which includes pitch, yaw, and roll angles ($P_{pitch_{(i)}}$, $P_{yaw_{(i)}}$, and $P_{roll_{(i)}}$) of the joints ($i = 0, \dots, 3$) of the short plant's skeleton. At each frame, we record the authored motion data $\vec{F} = \{\vec{P_0}, \dots, \vec{P_3}\}$. The recording is performed at a frame rate of 30 FPS. In Equations 2, 3, and 4, we introduce a scaling factor $\vec{s_i}$ ($s_{pitch_{(i)}}$, $s_{yaw_{(i)}}$, and $s_{roll_{(i)}}$) for the joints ($i = 0, \dots, 3$). $\vec{s_i}$ is set smaller for a more rigid short plant. We will explain how to set these scaling factors in Section 5.4.

5.1 Bending

(1) **Root bending:** We author the root bending of the short plant using the wrist by simply bending or waving the hand. The root bending is actually the overall bending of the entire short plant. It is mapped as the wrist's pitch and yaw motions (Figures 2(a) and (b)). The wrist can bend towards any directions, thus its motions are suitable to model the overall bending of the short plant, moreover it determines the bending direction of the entire short plant. The pitch and yaw motions of the root joint of the short plant's skeleton are set as:

$$P_{pitch_{(0)}} = s_{pitch_{(0)}} W_{pitch}, \;\; P_{yaw_{(0)}} = s_{yaw_{(0)}} W_{yaw}, \tag{2}$$

where W is the rotational data of the wrist. The short plant can bend to any directions and recover as well as smoothly bend from one direction to another according to the wrist motion.

(2) **Upper bending:** We author the bending of the upper parts of the short plant using the principle finger by simply bending the finger joints (Figure 8). The bending angles of the

joints of the principle finger are used to determine the bending amount of the upper joints of the short plant's skeleton. The upper parts of a short plant usually bend in the same direction as the bottom part, and they are different only in the magnitude of the bending. Fingers can only easily bend inwards to the palm or recover to its rest position, performing other motions may not be feasible for fingers. However, this one degree of freedom is sufficient to control the bending amount for the upper parts of the short plant while following the root bending direction of the short plant. The pitch and yaw motions of the upper joints ($i = 1, \ldots, 3$) of the short plant's skeleton are computed as:

$$\begin{aligned} P_{pitch_{(i)}} &= s_{pitch_{(i)}}(P_{pitch_{(0)}})G_i/G_{R_i}, \\ P_{yaw_{(i)}} &= s_{yaw_{(i)}}P_{yaw_{(0)}}G_i/G_{R_i}, \end{aligned} \tag{3}$$

where G_i is the bending angle of i-th joint of the principle finger, and G_{R_i} is its bending range. For short plants, the motions of the root and upper parts are in general coherent and linked. The pitch and yaw motions of the upper parts are affected by its root bending (pitch and yaw) in terms of not only the direction but also the magnitude. Thus, we incorporate the wrist (root) bending to compute the bending of upper parts. Basically, we scale G_i/G_{R_i}, and the root bending can be considered as scaling factors. Intuitively, this means if the overall (root) motion is little, the added motion for upper parts is also little. This simple equation helps to make the following process efficient: in order to bend the upper parts of the short plant from one bending direction to another, the users can only perform one step: (1) bending the wrist to the target direction while the principle finger can be kept bent, instead of performing three steps: (1) recovering the principle finger to its rest position, (2) bending wrist to the target bending direction and (3) bending the principle finger to achieve the target bending amount.

5.2 Twisting

The twisting of the short plant is mapped to the wrist's roll rotation, for example in Figure 2(c), the blade of grass is aligned with the palm. The roll motion (P_{roll}) of the joints ($i =$

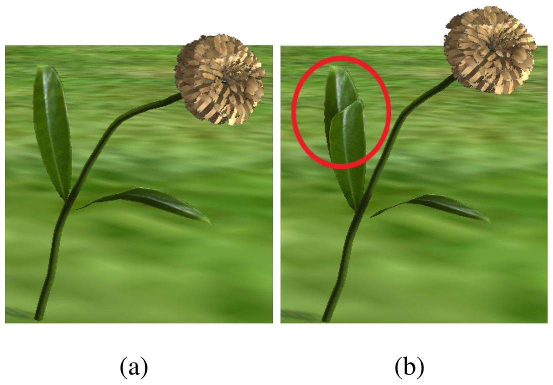


Figure 9: Authoring (a) the global animation for the whole flower, then (b) the secondary animation for the leaf.

$0, \ldots, 3$) of the short plant's skeleton is computed as:

$$P_{roll_{(i)}} = \begin{cases} 0 & , \ i = 0, \ldots, 3 - \tau, \\ s_{roll_{(0)}} W_{roll} & , \text{ otherwise.} \end{cases} \tag{4}$$

We set the number of twist-able joints (τ) based on the types of short plant, for example, a rigid short plant is difficult to twist, thus it has less number of twist-able joints. For short plants on the ground or the leaves, the root is usually not twist-able ($P_{roll_{(0)}} = 0$). In most of our experiments, we set $\tau = 2$.

5.3 Secondary animation authoring

The authoring process is performed for the whole short plant. If the short plant contains only one element (e.g. one grass has one blade), the authoring is finished. However, if it contains some secondary elements such as leaves and the users would like to superimpose some secondary motions (Figure 9(b)), they can repeat the authoring step once or several times to create the secondary animation(s). The secondary animation results are used to animate all the leaves of this short plant. For the secondary animation results, we usually randomly assign them to the secondary elements (leaves) if they are all similar, however for short plants with different type of secondary elements, such as the petal and leaf in a flower, we author and assign the secondary animations respectively.

During the authoring of the secondary animation, the animation result for the whole short plant can be viewed and looped (Section 6). This provides users the option to get on-the-fly feedback of the overall motion effects they are creating, e.g. users can instantly anticipate if the motions of leaves still look good when the flower bends to different directions. Moreover, this also provides users with a reference for animating the secondary elements, as the secondary motions and global motions are usually correlated. For example, we observed that when a short plant bends to one direction, the secondary elements (leaves) also bend accordingly. By looking at the global motion, users can author the secondary motions while taking into account the global motion's properties such as the swaying frequency and direction. This is helpful to ensure the global and secondary motions are coherent.

5.4 Hand gesture capturing setup

In Leap Motion controller's setup, the hand's palm is facing the camera. The thumb's bending direction is almost perpendicular to the camera's viewing direction, however the other fingers' bending direction is parallel to the Leap Motion camera's viewing direction. Hence, the thumb's bending is captured more accurately than others' because the tracking of the thumb has less occlusion artifacts. Therefore, we choose to use the thumb as the principle finger.

Using the Leap Motion controller (with SDK), we are able to track the wrist's motions and the thumb joints' bending angles. However, our approach is device independent, for example if we use some glove (marker) based hand motion capture system, we can also choose the forefinger as the principle finger. That means any hand motion capture devices can be adopted, as long as the bending angles of the wrist and some fingers can be obtained.

People usually have different capabilities in bending the wrist and the fingers. Hence, when we set the scaling factor $\vec{s_i}$ in Equations 2, 3, and 4, we also consider the user's ability to bend his/her wrist and thumb. This means, we also set $\vec{s_i}$ for calibrating the bending of wrist and thumb in order to map the user's comfortable ranges of bending wrist and thumb joints (G_{R_i}) with his/her desired bending ranges of plant's joints. This step only needs to be done once prior to the authoring.

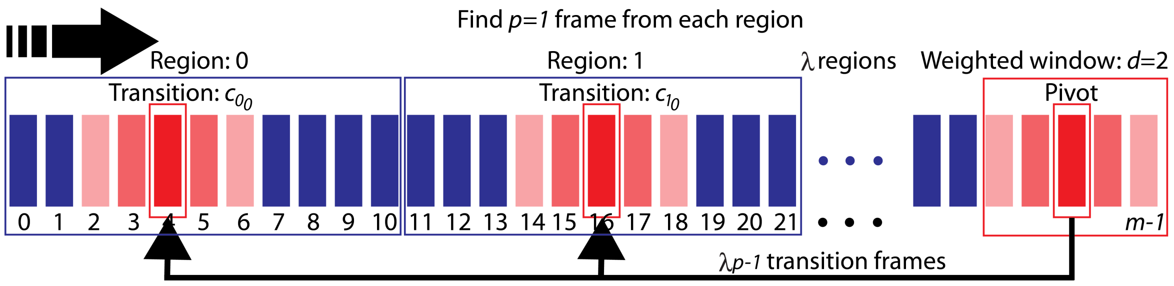


Figure 10: Creating an animation sequence with infinite duration.

6 Using and controlling the authored animations

We fetch the recorded animation data and feed it into the underlying skeleton model to animate the short plant based on the LBS skinning method (Equation 1), and M_i is the rotation matrix and it is computed by compositing the rotation matrices that correspond to the recorded $P_{pitch_{(i)}}$, $P_{yaw_{(i)}}$, and $P_{roll_{(i)}}$ bending. We also provide user with the options to create seamless infinite animation sequences (Section 6.1) and use the animation result to simulate a dynamic vegetation field (Section 6.2) as well as some high level controls such as speeding up or slowing down the animation (Section 6.3).

6.1 Animation looping

We propose a method similar to the idea of video texture [28] to animate short plants using the authored animations. As shown in Figure 10, we first select some transition frames from the m frames of the recorded motion data ($\vec{F}_0, \ldots, \vec{F}_{m-1}$). We then play the frames sequentially, if we encounter a transition frame (not the last one), we can randomly select one of the other transition frames to proceed or continue to proceed. If we encounter the last transition frame, we randomly select one of the other transition frames to proceed. As a result, we can achieve a seamless infinite animation while retaining the authored animation pattern.

To achieve this, we choose one frame as a transition pivot and use it to search for other transition frames. In our implementation, we use the frame at the back ($\vec{F}_{m-1-d}$) as the pivot. The recorded m frames are equally divided into λ regions, at each region we find p

most similar frames comparing to the pivot. By doing so, the distribution of the transition frames can be more uniform. The comparison is performed based on the adjacent neighborhood with a weighted window size $(2d + 1)$ and the Euclidean distance. At region r ($r = 0, \ldots, \lambda - 1$), the transition frames of indexes $c_{r_0}, \ldots, c_{r_{p-1}}$, are computed as:

$$c_{r_k} = \underset{j}{\arg} \underset{k}{\min} \sum_{i=-d}^{d} b_i \left\| \vec{F}_{j+i} - \vec{F}_{m-1-d+i} \right\| , \tag{5}$$

where b_is are the binomial weights and $j = 0, \ldots, \frac{m}{\lambda}$, $k = 0, \ldots, p - 1$. Small λ and small p (less transition frames) mean the looping effect is less fuzzy, in practice we set λ in the range of $[2, 5]$ and p in the range of $[1, 3]$. During transition if the two frames differ a lot, we use linear interpolation to insert some frames.

6.2 Motion propagation

Suppose we aim to create a dynamic field of short plants under the influence of external forces whose direction is from left to right. The external forces can be a wind generated by a passing object, a natural wind blowing across the field, or a character pressing the short plants. We generate the field by defining 2D grids to model the ground and randomly selecting some grids and placing one short plant at a random location inside each selected grid. The short plants can be sorted and indexed via the 2D grids. To create the animation, we first author one animation of a single short plant. In general, the bending direction should be mostly towards right in order to be perceived as the direction of the external forces is rightward, however some oscillations as well as a little swaying towards other directions can be added. Animation looping method (Section 6.1) is then used to create an infinite animation. Since different parts of the field have similarities in terms of motions, we propose to use this animation result of a single plant for animating the short plants in the whole field. For a field with multiple types of short plants, we consider all the short plants of one type as one field, we author and animate each field respectively, then combine these fields by superimposition.

To initiate the motion propagation effect, for the left most short plant at each row (left boundary) of the field, we randomly select one frame from the authored data as the starting

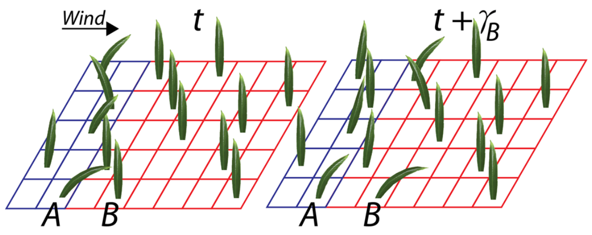


Figure 11: During time γ_B, the movement propagates from position A to B.

frame from which we play forward the recorded frames to animate the short plant. Note that, during the animation looping, the left most short plant determines the frame sequences for its row, and the animation of the rest of the short plants of this row are derived based on it. When the direction of the external forces is from left to right, the short plant on the right reacts later than the short plant on the left, so we use the previous motion of the short plant on the left to model the current motion of the short plant on the right. We propose a method similar to the idea in continuum grass [19]. In Figure 11, consider the movement ($\vec{F}$) propagates from position A to B during time γ_B, the motion of short plant ($\vec{F_B}$) at position B is actually the motion of short plant ($\vec{F_A}$) at position A at γ_B time ago.

$$\vec{F}(t)_B = \vec{F}(t - \gamma_B)_A, \tag{6}$$

$$\gamma_B = \|B - A\| / V_M, \tag{7}$$

where V_M is the propagation speed of the motion. Based on the time difference γ_B and the recording frame rate (30 FPS), we can easily compute the frame difference ($30\gamma_B$) and fetch the motion data based on the frame index. By propagating the rightward bending effects of the short plants from the left to the right of the field, a rightward turbulence or natural wind can be synthesized. The above approach can also be used to simulate a dynamic field under the influence of external forces in arbitrary directions.

6.3 Animation control

Control functionalities: We provide users with the following controls which are categorized with respect to the effects that they are related to:

(1) **Direction of the external forces:**

(a) Direction of motion propagation in a field: It is achieved by searching A in Equation 6 in the opposite direction of the external forces.
(b) Direction of bending: It is achieved by rotating the bending direction which is defined as (P_{pitch}, P_{yaw}). This direction follows the direction of the external forces. If users want to introduce more twisting effects to incorporate the new external forces direction, they need to re-author the motions.
(2) **Strength of the external forces:**
(c) Speed of motion propagation in a field: For a field under the influence of stronger external forces, the motions propagate faster across the field than a field under the influence of lighter external forces. It is achieved by setting the V_M in Equation 7.
(d) Speed of motion: For a short plant under stronger external forces, its motion speed is usually higher than the short plant under lighter external forces. Speeding up and slowing down the motion speed are realized respectively by decreasing and increasing the time step to forward the animation frames which are originally captured in 30 FPS by default.
(e) Scale of motion: For a short plant under stronger external forces, its bending amount is usually larger than the short plant under lighter external forces. We also provide users the option to up or down scale the bending motions by setting $\vec{s}_i$ in Equations 2, 3, and 4.
(3) **Rigidity of the motion:** As shown in [19], a soft short plant's motion usually exhibits the property that its upper parts are dragged by the bottom parts, in other words, the upper parts react later than the bottom parts. By using the proposed hand gesture based method, we can already mimic and synthesis this effect, on top of that, we also propose a method to further adjust the rigidity of the motion based on backtracking in time. At time t, instead of using $\vec{P}_i(t)$, we use $\vec{P}_i(t - \delta_i)$, $(i = 1, \ldots, 3)$, where δ_i is the time delayed for each upper part, a soft short plant has larger δ_i. In implementation, this is realized by fetching motion data $\vec{F}$ $(\{\vec{P}_0, \ldots, \vec{P}_3\})$ from the recorded frames based on the delay time δ_i, this is actually the same mechanism as we handle γ_B in Equation 7.
Control interface: We propose a hand gestural interface for the above controls: users can swipe hand in arbitrary directions to the Leap Motion's camera to set the corresponding direction of external forces. We map the speed of swiping to the speed of motion propagation as well as the speed of motion and map the length of swiping to the scale of motion.

We compared this interface with keyboard control: users can control the direction of the external forces by pressing the left, right, up, and down keys as well as increasing or decreasing the strength of the external forces. In our experiments, we observed that, in this case of animation control, using hand gestures maybe interesting and intuitive, but it has no advantages over using traditional keyboard in terms of efficiency and effectiveness. Therefore, in our framework we keep the hand gestural interface as one option for animation control. We believe in the near future, an environment with only hand gestural input (e.g. no keyboard) may emerge and this interface can be useful then.

Animation editing: We propose a simple approach to edit the authored animation. If users are not satisfied with a certain part of the animation, they can mark the start and end frames of that part and re-author an animation sequence to replace it. To insert the newly authored animation, we use linear interpolation to blend the boundary frames.

6.4 Guided procedural animation

In addition, we also propose a semi-automatic approach to animate a short plant by using hand motions to guide a procedural animation method [19]. During the recording stage, we only acquire the motions of wrist and use it to author the animation of the root as in Sections 5.1 and 5.2. We then propagate the motions of the root to the upper parts as in [19]. At time t, the motions of the upper joints ($i = 1, \ldots, 3$) of the short plant's skeleton are computed as:

$$\vec{P}_i(t) = \vec{P}_0(t - s_{g_{(i)}} G_3 \sigma_i) \tag{8}$$

where σ_i is the time delayed for each upper part, a soft short plant has a larger σ_i. We handle it using the same mechanism as we handle γ_B in Equation 7. We also propose to use the thumb bending to control the rigidity of the short plant, by introducing $s_{g_{(i)}} G_3$ in this equation, where $s_{g_{(i)}}$ is a scaling factor and G_3 is the bending angle of the top joint of the thumb which is generally the most flexible joint of the thumb. This approach can be integrated with the proposed animation looping and motion propagation methods to animate a field of short plants. This approach only authors the overall motions of the short plants which are actually the root motions of the whole short plant, the motions are then propagated to its upper parts automatically. Thus, it is suitable when only a high-level design of overall motions of the short plant is required, such as when modeling a simple grass swaying in natural wind. It is not suitable to author motions for short plants require detailed design for upper parts, such as for flowers, the petals (top parts) may require detailed control and design.

6.5 Alignment of secondary animation

Based on the observation that when the motions of the whole short plant (global motions) are severe, the motions of the secondary elements (leaves) are usually also severe at the same time (the opposite is usually not true). To model this effect, the secondary motions need to be aligned with the global motions. Users can author the secondary motions to achieve this. In many scenarios, the animated vegetation scene may contain a number of secondary animations and they may be out of such alignment. Thus, we also propose an automatic method to implicitly align the secondary motions with the global motions to maintain the consistency. The overall effects of the authored secondary motions are still maintained. Please refer to the supplemental video for the comparison of out-of-alignment and aligned results.

This motion alignment is achieved by aligning three important motion characteristics: motion direction, motion frequency, and motion amount. The motion direction can be aligned by adjusting the direction of secondary motions as introduced in Section 6.3 (1). In this section, we focus on the alignment of motion frequency and motion amount. Since one motion is in general perceived as one sequence of poses, at each time we process one subsequence of the authored motion data. Each subsequence corresponds to the same fixed time range. We usually use 2 seconds per subsequence (60 frames for 30 FPS). After one subsequence of the secondary motion data is processed to be aligned with its corresponding subsequence of global motion, it still corresponds to 2 seconds, but the number of frames within this subsequence may change (speed up or slow down). We count another 2 seconds (60 frames) secondary motion data after the end of the processed subsequence as the next to process. The process path is determined by the animation looping (Section 6.1).

We first compute the total motion frequency and motion amount (bending amount) within the subsequence with index j respectively, for the global motions of the whole short plant ($f_G(j)$, $A_G(j)$) and for the motions of the k-th secondary element (($f_{S_k}(j)$, $A_{S_k}(j)$)). To compute the total motion frequency, we count the number of times that the bending amount of the short plant (or the secondary element) achieves a local maximum or minimum within the subsequence. We perform this counting for all parts (root and upper parts) of the short plant (or the secondary element), and sum the counting results as the total motion frequency. Similarly, to compute the total motion amount within the subsequence, we sum the bending amount (the amplitude of the bending) for all parts.

We then align the secondary motions with the global motions. The basic idea of the alignment is based on matching their total motion frequencies and amounts via scaling. By doing so, we can align the motions while still maintaining the overall effects of the authored secondary motions. For example, if the global motions are getting faster, the secondary motions are also getting faster, while still maintaining their trajectories. For the subsequence with index j, to align the motion frequencies, we speed up the motion of all parts of the k-th secondary element as mentioned in 2(d) in Section 6.3 based on $r\frac{f_G(j)}{f_{S_k}(j)}$, where r is a random scaling factor between 0.9 to 1.1. Similarly, to align the motion amounts, we scale the motion of all parts of the k-th secondary element as mentioned in 2(e) in Section 6.3 based on $r\frac{A_G(j)}{A_{S_k}(j)}$. This method also can be used to align the motions of different types of short plants in a vegetation field.

7 Results and evaluations

We applied our method to animate 3D short plants: grass, flowers, and shrubs as well as short trees (Figure 3, also please refer to the supplemental video). Due to submission size limit, the submitted supplemental video has a lower resolution. The original captured video of our methods, results, and comparisons can be found at
https://www.dropbox.com/s/e0hnx8c3ijib5ur/cavwPlant.avi?dl=1.

Our method was implemented on a workstation with Intel Xeon E5-2650 2.6 GHz CPU, 16 G Memory and Nvidia Quadro K6000 GPU. All the operations for controlling an animation in Section 6.3 can be performed in real-time, users can adjust the animation on the fly. Users can author animation sequences with arbitrary lengths. In our experiments, the users only need to author 10 to 30 seconds animation for the global animation and each secondary animation. Although the users may author several times to achieve their desired results, each authoring session takes only 10 to 30 seconds.

We evaluate our method by comparing our results with (1) short plants in cartoons, (2) real short plants, and (3) the state-of-the-art award-winning toolkit SpeedTree (please refer to the supplemental video). Real and cartoon short plants exhibit bending and twisting motions. Our approach can effectively generate such motions while addressing the characteristics of the motions in the original videos (observations in real-life) and cartoons (design ideas). This is important in many scenarios such as teaching animation lessons to pupils and fast prototyping. This is not a trivial task using existing methods (such as SpeedTree), but since human are proficient in using a hand, such tasks can be efficiently done using our method.

Based on our knowledge, in the conventional 3D animation pipeline, during the animation stage, animators move the handles of the rigged 3D model to create poses. For instance, in the "Kung Fu Panda 2" (©DreamWorks Animation) example, in order to achieve such a nice artistic motion of a shrub interacting with a water drop which is driven by "Master Shifu": receiving the water drop, making an interesting turn, dropping it to the pond and recovering, we believe it requires tedious efforts and art skills to set the poses by manually moving relative handles. Such artistic motions also cannot be easily achieved using the existing physics-based and procedural methods by setting and tuning a number of parameters.

We also consulted with a professional artist. He told us that in general there are two approaches. The first one is based on carefully setting a specific rigging, as the animator moves the body of the shrub, other parts will follow. However, this rigging is not easy to set. The second approach is based on the normal rigging and key frames. It may require several steps and iterations to author the primary motions and secondary motions, during each step, the animator needs to set and tune the poses and the motion curves. These two approaches require tedious efforts and art skills. However, using our method, a similar animation can be generated in 7 seconds. If the result is not satisfying, the user can efficiently re-author or edit it. Based on the authored result, details can be added later during the production stage. Moreover, using our method, even non-expert users are able to generate desired animations of short plants.

SpeedTree is able to create realistic animations for a range of different types of plants. For short plants, we demonstrate that our method can achieve comparable results and our method provides a more intuitive and efficient approach. For example, in our experiment, authoring a grass animation (15 seconds) using SpeedTree took about 10 minutes, however using our method only took about 20 seconds.

7.1 User study and evaluation

Participants: The participants include 3 professional artists with average of 6 years experiences in game/animation industry and 7 researchers of visual computing (4 are from the same research center of the authors, 3 are from other labs). Out of 10 participants, 7 of them are familiar with 3D gestural interfaces (e.g. Leap Motion). In our study, the participants performed animation authoring tasks on a workstation with a Leap Motion controller.

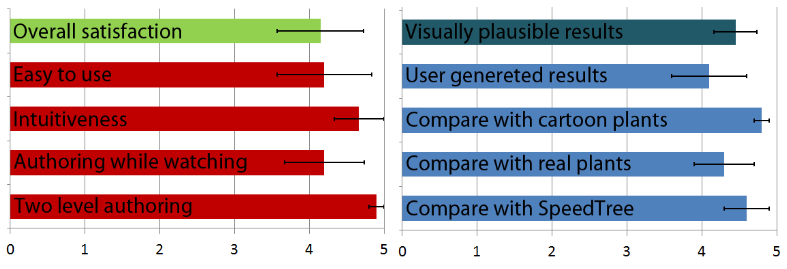
Procedure: The study includes four sections: warm-up, target animation authoring, open animation authoring, results evaluation, and final interview.

(1) Warm-up: We first give participants an explanation of our proposed interface and the procedure of our user study. Then, they are asked to try pitch, yaw, and roll motions using hand. Meanwhile, the scaling parameters in Equations 2, 3, and 4 may be set based on their preferences. After they are familiar with all the functions of our system, we proceed to the next section. This section took around 5 minutes.

(2) Target animation authoring: After warm-up, the participants are asked to author an animation of a short plant based on a given 14 seconds reference video footage of a short plant. They are asked to try to reproduce the characteristics of the motions in the given video. It took 2 to 5 minutes for them to create their satisfied animation results.

(3) Free authoring: In this section, the participants are asked to use our system for free animation authoring for short plants.

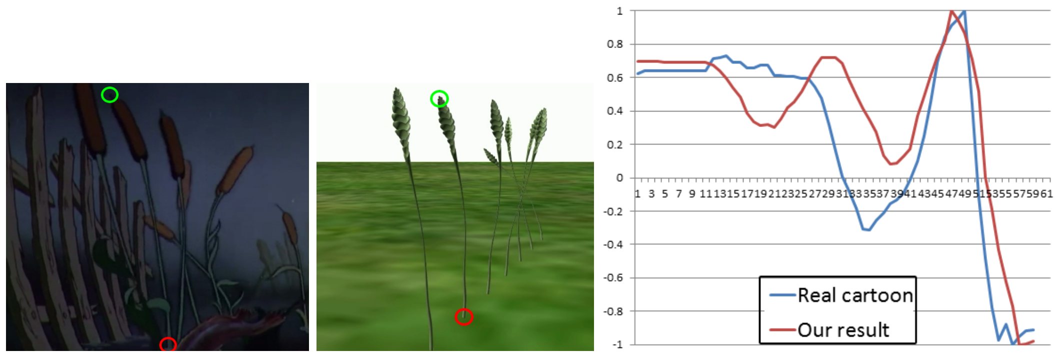
(4) Animation result evaluation: In this section, we invite the participants to evaluate various animation results created using our system, including (a) each other's authoring results as well as comparison results with (b) real and (c) cartoon short plants as well as (d) SpeedTree.



(a) (b)

Figure 12: User feedback. (a) Feedback of the system. (b) Feedback of the animation results.

System evaluation results: System evaluation results: Figure 12 (a) shows the feedback from the participants about our system. In general, all the participants are satisfied with our system. They think it is natural to use hand motions to represent the motions of short plants and our hand gestural interface is intuitive. They consider our system to be helpful for non-expert users to easily realize their desired motions. They think the two-step approach is natural, since during each step they can only focus on one task (the average rating is 4.9, all the scores are from 1 to 5, 1 for bad and 5 for good). They also think the option of authoring the secondary motion while watching the global motion is helpful (the average rating is 4.2). For the 3 professional artists, they think our method is more efficient than existing 3D software for creating a specific animation for short plants. They feel our system is very helpful especially when conducting fast prototyping and in the early stage of design where the illustration and communication of ideas intensively occur. We can efficiently support this since our method provides a solution for rapid authoring of animations for short plants.
Animation evaluation results: The step (4) is a qualitative study to evaluate if our method can create motions effects that are perceived as natural looking by the viewers (Figure 12 (b)). The participants think they can create visually plausible results similar to the reference motions (including each other's authoring results, the overall average rating is 4.1, the average rating is 4.0 evaluated by the three professional artists and 4.1 evaluated by the other users). Comparing with real and cartoon short plants, they think our animation results are visually plausible and the characteristics of the motions in the original cartoons and videos can be well captured, especially for the short plants in cartoons (the average rating, cartoon:



(a) (b) (c)

Figure 13: Motion similarity. (a) Cartoon wheat from “The Old Mill” from Walt Disney’s “Silly Symphonies”. (b) Our result. (c) Motion similarity chart.

4.8, real: 4.3). They think our authored animation result is comparable with the animation result using SpeedTree (the average rating is 4.6).

Motion similarity: Moreover, we also perform a motion similarity experiment to evaluate the animation results. This experiment is to test if the important characteristics of the reference motions can be captured: the frequency, pace and amplitude change of the motion. We select one feature point of a short plant, e.g. we usually use the tip of the grass or leaf (Figure 13 (a, b)). We track the feature point and the root of the short plant in each frame for both the authored result and reference video. The tracking is done manually: at each frame, we look for the feature point (P_x, P_y) and the root (R_y, R_y) and record their positions. Based on these two positions, we calculate the approximated bending angle at each frame using $\arctan(\frac{P_x - R_x}{P_y - R_y})$. Then we normalize the bending angles and plot the curves of bending angles versus frames for both the authored result and reference video. As shown in Figure 13 (c), we put the two curves together in one chart and observed that their shapes are similar, e.g. both curves generally have 2 peaks (frequency), the distances between the peaks are similar (pace), both curves show small bending amplitudes followed by a big bending amplitude (amplitude change). This shows that our method is able to capture the important characteristics of the reference motions. Please refer to the video for the chart of each example.

Our results show that using the proposed method to author bending and twisting motions, we are able to rapidly create the desired natural looking motions for many scenarios for short plants such as grass, flowers, and shrubs, under the influence of external forces, such as passing object and natural wind.

7.2 Generalization and limitation

Generalization: The proposed approach is an efficient solution for authoring bending and twisting motions. It can be used to animate other objects which also exhibit such motions, such as fur, short hair and some body parts of virtual characters such as short tail.
Our method can be integrated in to the traditional animation pipeline, for example in the pre-production stage, using our method can do fast prototyping and illustrate/exchange design ideas, and during the actual production, animators can start based our authored results.
Limitation: Our method may not achieve physically realistic results for real plants. However, it has the advantage of being fast. As such, users can efficiently perform re-authoring to improve the animation results. We also provide users with the tools to edit and fine tune the effects.

In our method, there are basically two types of parameters. The first type is to calibrate the hand gesture capturing device. This is usually necessary and only needs to be done once prior to the authoring. The second type of parameters aims at providing an option to control or to fine tune the animation results. This is usually optional. First, since our method is fast, users can always choose to re-author the animation if it is not satisfying. Second, if users think that only a small part is not satisfying, this type of parameters provides the tool to adjust it.

The Leap Motion controller used in our implementation can detect slight hand jitter. These detailed motions can be utilized to add richness to the final animation results, however, in some cases, such detailed motions may be considered as unnatural looking (noises) especially for cartoonish motions. To reduce unwanted noises, we can apply a low-pass filter (or a smooth kernel) as in the conventional signal processing when processing the hand gestures. This low-pass filter is an anti-aliasing filter to generate smoother animation results.

This filter can be set on-the-fly during capturing and authoring. Since our method has the advantage of being fast and effective, users can efficiently perform re-authoring to achieve the desired smoothness for the animation results. The users can also choose to apply the low-pass filter as a post processing step, when performing the editing.

Currently we do not explicitly perform collision detection and response. As a result, short plants may intersect each other when collisions happen. However, in the generated animation, short plants are continuously bending and twisting and thus it is not obvious to see the intersections when they occur.

8 Conclusion and future work

In this paper, we have proposed to use a hand as a 'puppet' to author animations of short plants. Because of the intuitiveness and efficiency of our method, it is suitable for non-expert users and helpful for fast prototyping and authoring specific motions of short plants. The proposed method is based on the observations: different from tall plants, short plants usually have lower degrees of freedom of bending and twisting motions, thus these motions of short plants can be well mimicked with the movements of a hand. In addition, it is natural to use hand gestures to describe the motions of short plants. Our method is based on transferring the motions of a hand to the motions of a short plant. Since human are very proficient in using hands and fingers, the essential motions of short plants can be efficiently realized. Moreover, the properties of these motions can be effectively and easily controlled using a hand, such as the bending directions, frequencies, and trajectories. Therefore the characteristics of the targeted motions can be well addressed. To author the animation for one short plant, we have employed a coarse-to-fine authoring approach. An implicit animation alignment method is also introduced. We have also provided users with a semi-automatic method suitable for users who only want a high-level design of the overall motions of short plants.

We have proposed an efficient framework to reuse the authored animation results of one single short plant to animate a field of short plants. As a result, users can intuitively generate their desired motions of short plants under the influence of external forces. Our method does not need the tedious manual posing or the expensive physical simulation. Our authoring process usually takes only less than one minute.

In the future, we plan to explore using a hand to author other objects which also exhibit bending and twisting motions, such as fur and short hair. Another future work is to extend our method to animate tall and complex plants. We also plan to incorporate physics-based simulation and use hand gestures to impose constraints.

Acknowledgment

This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its International Research Centres in Singapore Funding Initiative.

References

[1] Leap Motion. Leap Motion, Inc.

[2] Nimble VR. Nimble VR, Inc.

[3] Ming Jin, Dan Gopstein, Yotam Gingold, and Andrew Nealen. Animesh: Interleaved animation, modeling, and editing. *ACM Transactions on Graphics*, 34(6):207:1–207:8, 2015.

[4] Mikio Shinya and Alain Fournier. Stochastic motion—motion under the influence of wind. *Computer Graphics Forum*, 11(3):119–128, 1992.

[5] Jos Stam. Stochastic dynamics: Simulating the effects of turbulence on flexible structures. *Computer Graphics Forum*, 16(3):159–164, 1997.

[6] Yung-Yu Chuang, Dan B Goldman, Ke Colin Zheng, Brian Curless, David H. Salesin, and Richard Szeliski. Animating pictures with stochastic motion textures. *ACM Transactions on Graphics*, 24(3):853–860, 2005.

[7] Ralf Habel, Alexander Kusternig, and Michael Wimmer. Physically guided animation of trees. *Computer Graphics Forum*, 28(2):523–532, 2009.

[8] Julien Diener, Mathieu Rodriguez, Lionel Baboud, and Lionel Reveret. Wind projection basis for real-time animation of trees. *Computer Graphics Forum*, 28(2):20–28, 2009.

[9] Yasuhiro Akagi and Katsuhiro Kitajima. Computer animation of swaying trees based on physical simulation. *Computers and Graphics*, 30(4):529–539, 2006.

[10] Yili Zhao and Jernej Barbič. Interactive authoring of simulation-ready plants. *ACM Transactions on Graphics*, 32(4):84:1–84:12, 2013.

[11] J.P. Weber. Fast simulation of realistic trees. *IEEE Computer Graphics and Applications*, 28(3):67–75, 2008.

[12] Orthmann Jensn, Christof Rezk Salama, and Andreas Kolb. GPU-based responsive grass. 17(1-3):65–72, 2009.

[13] Zengzhi Fan, Hongwei Li, Karl Hillesland, and Bin Sheng. Simulation and rendering for millions of grass blades. In *I3D '15: Proceedings of the 19th Symposium on Interactive 3D Graphics and Games*, pages 55–60, 2015.

[14] Shin Ota, Tadahiro Fujimoto, Machiko Tamura, Kazunobu Muraoka, Kunihiko Fujita, and Norishige Chiba. 1/f β noise-based real-time animation of trees swaying in wind fields. In *CGI '03: Proceedings of Computer Graphics International 2003*, pages 52–59, 2003.

[15] Tiago Sousa. Vegetation procedural animation and shading in Crysis. *GPU Gems 3*, pages 373–385, Hubert Nguyen, 2007.

[16] Shaojun Hu, Norishige Chiba, and Dongjian He. Realistic animation of interactive trees. *The Visual Computer*, 28(6-8):859–868, 2012.

[17] Kurt Pelzer. Rendering countless blades of waving grass. *GPU Gems*, pages 107–121, Randima Femando, 2004.

[18] Renaldas Zioma. GPU-generated procedural wind animation for trees. *GPU Gems 3*, pages 105–120, Hubert Nguyen, 2007.

[19] Kan Chen and Henry Johan. Real-time continuum grass. In *VR '10: Proceedings of the 2010 IEEE Virtual Reality Conference*, pages 227–234, 2010.

[20] Kan Chen and Henry Johan. A simple method to animate vegetation in images using simulation-guided grid-based warping. In *CAD/Graphics '13: Proceedings of the 14th International Conference on Computer-Aided Design and Computer Graphics*, pages 244–251, 2013.

[21] Kan Chen and Henry Johan. Simple and efficient example-based texture synthesis using tiling and deformation. In *I3D '15: Proceedings of the 19th Symposium on Interactive 3D Graphics and Games*, pages 69–76, 2015.

[22] SpeedTree. Interactive Data Visualization, Inc.

[23] Lucas Kovar, Michael Gleicher, and Frédéric Pighin. Motion graphs. *ACM Transactions on Graphics*, 21(3):473–482, 2002.

[24] Doug L. James, Christopher D. Twigg, Andrew Cove, and Robert Y. Wang. Mesh ensemble motion graphs: Data-driven mesh animation with constraints. *ACM Transactions on Graphics*, 26(4):17:1–17:21, 2007.

[25] Long Zhang, Chengfang Song, Qifeng Tan, Wei Chen, and Qunsheng Peng. Quasi-physical simulation of large-scale dynamic forest scenes. In *CGI '06: Proceedings of Computer Graphics International 2006*, pages 735–742, 2006.

[26] Long Zhang, Yubo Zhang, Zhongding Jiang, Luying Li, Wei Chen, and Qunsheng Peng. Precomputing data-driven tree animation. *Computer Animation and Virtual Worlds*, 18(4-5):371–382, 2007.

[27] Jie Long, Bryce Porter, and Michael Jones. Animation of trees in wind using sparse motion capture data. *The Visual Computer*, 31(3):325–339, 2015.

[28] Arno Schödl, Richard Szeliski, David H. Salesin, and Irfan Essa. Video textures. In *SIGGRAPH '00: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, pages 489–498, 2000.

[29] Julien Diener, Lionel Reveret, and Eugene Fiume. Hierarchical retargetting of 2d motion fields to the animation of 3d plant models. In *SCA '06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer animation*, pages 187–195, 2006.

[30] Chuan Li, Oliver Deussen, Yi-Zhe Song, Phil Willis, and Peter Hall. Modeling and generating moving trees from video. *ACM Transactions on Graphics*, 30(6):127:1–127:12, 2011.

[31] Michael Gleicher, F Sebastian Grassia, Zoran Popovic, S Rosnthal, and Jeffrey Thingvold. Motion editing: Principles, practice and promise. In *SIGGRAPH '00: ACM SIGGRAPH 2000 Courses*, 2000.

[32] Michael Gleicher. Retargetting motion to new characters. In *SIGGRAPH '98: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, pages 33–42, 1998.

[33] David J. Sturman. Computer puppetry. *IEEE Computer Graphics and Applications*, 18(1):38–45, 1998.

[34] Hyun Joon Shin, Jehee Lee, Sung Yong Shin, and Michael Gleicher. Computer puppetry: An importance-based approach. *ACM Transactions on Graphics*, 20(2):67–94, 2001.

[35] T. Shiratori, M. Mahler, W. Trezevant, and J.K. Hodgins. Expressing animated performances through puppeteering. In *3DUI '13: Proceedings of the IEEE Symposium on 3D User Interfaces*, pages 59–66, 2013.

[36] Chris Hecker, Bernd Raabe, Ryan W. Enslow, John DeWeese, Jordan Maynard, and Kees van Prooijen. Real-time motion retargeting to highly varied user-created morphologies. *ACM Transactions on Graphics*, 27(3):27:1–27:11, 2008.

[37] Noah Lockwood and Karan Singh. Finger walking: Motion editing with contact-based hand performance. In *SCA '12: Proceedings of the 2012 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 43–52, 2012.

[38] Yeongho Seol, Carol O'Sullivan, and Jehee Lee. Creature features: Online motion puppetry for non-human characters. In *SCA '13: Proceedings of the 2013 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 213–221, 2013.

[39] Chongyang Ma, Li-Yi Wei, and Xin Tong. Discrete element textures. *ACM Transactions on Graphics*, 30(4):62:1–62:10, 2011.

[40] Rubaiat Habib Kazi, Fanny Chevalier, Tovi Grossman, Shengdong Zhao, and George Fitzmaurice. Draco: Bringing life to illustrations with kinetic textures. In *CHI '14: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 351–360, 2014.

[41] Mira Dontcheva, Gary Yngve, and Zoran Popović. Layered acting for character animation. *ACM Transactions on Graphics*, 22(3):409–416, 2003.

[42] Helge Rhodin, James Tompkin, Kwang In Kim, Varanasi Kiran, Hans-Peter Seidel, and Christian Theobalt. Interactive motion mapping for real-time character control. *Computer Graphics Forum*, 33(2):273–282, 2014.

[43] Masaki Oshita, Hayato Oshima, Yuta Senju, and Syun Morishige. Character motion control by hands and principal component analysis. In *VRCAI '14: Proceedings of the 13th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, pages 171–179, 2014.

[44] Nkenge Wheatland, Yingying Wang, Huaguang Song, Michael Neff, Victor Zordan, and Sophie Jörg. State of the Art in Hand and Finger Modeling and Animation. *Computer Graphics Forum*, 34(2):735–760, 2015.

[45] Vicon. Vicon, Inc.

[46] CyberGlove. CyberGlove Systems.

[47] Paul G. Kry and Dinesh K. Pai. Interaction capture and synthesis. *ACM Transactions on Graphics*, 25(3):872–880, 2006.

[48] George ElKoura and Karan Singh. Handrix: Animating the human hand. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 110–119, 2003.

[49] 3DS Max. Autodesk.

Appendix: Procedural weights

```
// The weights are computed based on the distances to the joints
// The distances are measured based on height
for $i = 0, \ldots, n-1$ do
    $w_i \leftarrow 0$
    if $v_y < 0$ then
        $w_0 \leftarrow 1$ // Under ground part
    end
    else
        // Find which bone it is related based on the height
        if $J_{i_y} < v_y < J_{(i+1)_y}$ then
            T $\leftarrow \frac{v_y - J_{(i-2)y}}{J_{i_y} - J_{(i-2)y}}$ $w_{i-1} \leftarrow \begin{cases} \text{n/a}, & \text{if } i == 0. \\ T, & \text{if } i == n-1. \\ \frac{1.5}{T+0.5} - 1, & \text{otherwise.} \end{cases}$
            $w_i \leftarrow \begin{cases} 1 - T, & \text{if } i == 0. \\ 1 - T, & \text{if } i == n-1. \\ 1 - \frac{0.5}{T+0.5}, & \text{otherwise.} \end{cases}$
            $w_{i+1} \leftarrow \begin{cases} T, & \text{if } i == 0. \\ \text{n/a}, & \text{if } i == n-1. \\ 1 - \frac{1}{T+0.5}, & \text{otherwise.} \end{cases}$
        end
    end
end
```

Algorithm 1: Procedural weights