Reconstruction of a fluttering flag from a single image

Tao Hu, Jun Li and Guihuan Guo

Abstract
Reconstructing a 3D object from a single image is a challenging task because determining useful geometric structure information from a single image is difficult. In this paper, we propose a novel method to extract the 3D mesh of a flag from a single image and drive the flag model to flutter with virtual wind. A deep convolutional neural fields model is first used to generate a depth map of a single image. Based on the Alpha Shape, a coarse 2D mesh of flag is reconstructed by sampling at different depth regions. Then, we optimize the mesh to generate a mesh with depth based on Restricted Frontal-Delaunay. We transform the Delaunay mesh with depth into a simple spring model and use a velocity-based solver to calculate the moving position of the virtual flag model. The experiments demonstrate that the proposed method can construct a realistic fluttering flag video from a single image.

Keywords
Mesh reconstruction, depth map, restricted frontal-Delaunay, simple spring model

Introduction
Image enhancement, which is a very important field of augmented reality, has attracted attention from both academia and industry. Increasingly more people want to obtain abundant visual information and a friendlier visual experience by processing images. Additionally, many scholars are investigating how to produce a video or animation by using a single image. For example, Carlos Castillo et al. implemented the style transfer of an object in a single image with semantic segmentation. Menglei Chai et al. designed the AutoHair method to model hair from a single image based on deep neural network.

Since a single image only provides color and shape information on 2D planes, it is difficult to quickly reconstruct the 3D structures of objects in the image. If we can reconstruct the 3D structure of an object of interest from a single image, then a wealth of visual content can be generated. Therefore, how to generate the 3D structure of an object from an image is one of the core problems of image enhancement.

(1) For the scene driving a flag fluttering from a single image, we should reconstruct the 3D mesh of the flag firstly. Due to the performance of deep convolutional networks, we can estimate the depth map of the single image. If we directly segment the depth information of the flag from the depth map, it is very hard to use the depth information to generate the 3D mesh. Because the estimated depth map is uncertain and not very accurate. And the flag material is soft, the depth map is also difficult to map the details of the flag surface such as wrinkles. Therefore, this paper will construct a 3D mesh of the flag by combining 2D mesh with depth map, so as to realize the 3D detail reconstruction of the flag. On the other hand, how to drive the flag fluttering in a single image is also a very challenging problem. This is because of the lack of animation-driven information in a single image. On the basis of reconstructing the 3D mesh of flag, we will introduce a virtual wind force as the driving force of the flag animation.

In this paper, we focus on how to drive a flag to flutter from a still image. The flag is very thin...
according to a large number of observations; thus, the flag can be considered as a curved mesh in the 3D reconstruction process. The flag in the image is presented in a stereoscopic form, where the depths of field in different areas of the flag are different. Therefore, we employ the depth information to first reconstruct the 3D mesh of the flag from the image. To improve the efficiency of mesh reconstruction, same points sampled from the single image are used in our work. Then, we use a simple linear spring model\(^7\) to drive the reconstructed flag mesh to flutter.

The major contributions of this paper are summarized as follows:

1. A new animation method is proposed to generate a flag video animation form a single image. The proposed method uses a deep convolutional neural network to generate the depth map of input image. It generates a coarse 2D mesh of flag in the input image. And then, it combines the depth map of image and 2D mesh of flag to reconstruct the 3D mesh of flag. Finally, it drives the flag to Flutter by the virtual wind force.

2. An innovative reconstruction method is designed to generate a 3D mesh of a flag in the single image. The coarse 2D mesh is generated by Alpha Shape and sampling points in the image firstly. And the Restricted Frontal-Delaunay method is used to optimize the 2D mesh. We transform the 2D mesh into 3D space based on the depth map. Finally, we use the linear subdivision method to refine the 3D mesh with high density.

3. We propose an animation driving model to simulate the fluttering of the 3D flag mesh. The reconstructed 3D mesh is mapped into a simple linear spring model, where the vertex is transformed into a particle. We construct a joint vector model of internal and external constraints between each particle pairs. The unfired dynamics solver is used to solve the object equation of driving flag fluttering by virtual wind force. An batch rendering technique is used to map the texture of the flag into the 3D mesh.

The remainder of this paper is organized as follows. We review the existing works related to our proposed method in the next section, such as animation based on a single Image etc. Then proposed method is thoroughly presented. We conduct some experiments and analyze the proposed method in the subsequent section. Finally, we conclude our work with conclusion section.

**Related works**

The method proposed in this paper is related to generating an animation from a single image, which consists of depth estimation from a single image,\(^8,9\) mesh reconstruction\(^10,11\) and generating an animation of flag fluttering. Representative works that address these four issues are briefly reviewed below.

**Animation based on a single image**

Driving a flag to flutter from a single image is a research work of generating an animation based on a single image. Chuang et al.\(^12\) animated a single image using stochastic motion textures. Sun et al.\(^13\) employed the same method to extract the parameters of wind and water animation. Xu et al.\(^14\) constructed a video of moving animals from a single image based on the existing moving order. Jhou et al.\(^15\) designed a cloud appearance model to simulate cloud flow. These methods did not reconstruct the 3D shape of the object being driven from the single image, but our work constructs the 3D geometric structure before generating the animation.

**Depth estimation**

Computing the depth information from a single image is another challenging work in image processing. Ashutosh Saxena et al.\(^3\) employed supervised learning to train a Markov random field to infer the 3D location and orientation of a single image. Hongwei Qin et al.\(^9\) used parameter transfer from the depth maps and corresponding image database to estimate the depth of a still image with a lightweight model. Fayao Liu et al.\(^5\) proposed a deep convolutional neural fields (DCNF) method to estimate the depth of each pixel in a single image by exploring a convolutional neural network (CNN) and continuous conditional random fields (CRF) to infer a maximum a posteriori. Evan Shelhamer et al.\(^2\) also employed a full convolutional network to estimate the depth of a single image. Our work will use the existing method to calculate the depth of a single image.

**Mesh reconstruction**

Currently, 3D mesh reconstructions based on multi-images and RGB-D images are the popular model reconstruction methods. Jakob Engel et al.\(^16\) and RaúlMur-Artal et al.\(^17\) employed a monocular SLAM method to reconstruct a 3D scene. Michal Jancosek and Tomas Pajdla\(^18\) used Visual-Hull to generated the surfaces for the multi-view reconstruction problem. Andrew Owens et al.\(^19\) employed a combination of recognition and multi-view cues to generate dense 3D reconstructions. Michael Goesele et al.\(^20\) introduced a multi-view stereo method for large online images by estimating high-quality depth maps. Peter Henry et al.\(^21\) designed an RGB-D mapping system to construct dense 3D maps of indoor environments. Deep learning has also been applied for 3D reconstruction.
from images. For example, Haoqiang Fan et al. introduced generative networks to generate 3D geometrical structure from a single image. There is an obvious difference between our work and these methods. Our work only extracts some sampling points from the most common single image to generate a 3D mesh based on depth.

**Flag animation**

There are many novel algorithms for fluttering a virtual flag, which is one type of cloth. For example, Robert Bridson et al. described the collision, contact and friction methods for cloth animation. Rahul Narain et al. proposed a cloth simulation method based on dynamically refining and coarsening meshes. Ning Jin et al. introduced an inequality cloth paradigm to realize the bending and folding of virtual cloth with high expansion. Those methods consider the cloth animation in virtual 3D space, in which the cloth or flag has a definitive 3D mesh structure. Chen et al. proposed a mesh super-resolution to enrich low resolution cloth meshes with wrinkles based on deep learning, which uses the SRResNet to train an image synthesizer. However, our work focuses on driving the virtual flag to flutter with a reconstructed approximate mesh from a single image. Additionally, we employ a simple spring model to generate the video of a flag fluttering.

**Proposed method**

The contribution of this paper is that a fluttering flag video is generated from a single image, while the 3D mesh is reconstructed at the same time. Figure 1 presents an overview of the proposed method. The entire process can be summarized in three steps: generate the depth map of a single image, reconstruct the 3D mesh of the flag from the image, and drive the virtual flag to flutter in the image.

In the first step, we employ DCNF (F. 5) to generate the depth map of a single image. Inspired by Liu et al., the DCNF is trained in NYU v2 dataset. The NYU v2 dataset consists of 1449 RGBD images of indoor scenes. The standard training images with the NYU v2 dataset are 795. And 654 images are used for testing.

For example, given a single image \( I \) (Figure 1(a)), we can generate the depth map \( D_I \) of \( I \) (Figure 1(b)). The depth map \( D_I \) does not completely refer to the entire structure of the flag in image \( I \); thus, reconstructing the 3D mesh \( T \) of the flag directly based on the depth is very difficult. Considering the above, this paper employs the depth map \( D_I \) to reconstruct the coordinate value of the vertex \( V_i \) of the virtual flag mesh \( T \) on the Z axis. We introduce the next steps in the following.

**Reconstruct the 3D mesh of a flag from a single image**

| Algorithm 1. Mesh reconstruction algorithm based on Alpha Shape and sampling points in a single image |
|---------------------------------------------------------------|
| **Input:** Sampling point set: \( Q \); Probe radius: \( R \); **Output:** |
| **2D mesh structure:** \( S \); Vertex set: \( V \) |

1. **procedure** MESHRECONSTRUCTION(\( Q \), \( R \))
2. **for** each sampling point: \( Q_i \) **do**
3. \( V_i^x = Q_i^x \), \( V_i^y = Q_i^y \), \( V_i^z = 0 \)
4. **end for**
5. \( \triangle tri = \text{Delaunay}(V) \)
6. **for** each simplex Delaunay triangle:
7. \( r_k = \frac{L_{\triangle tri}}{(4 \times D_{\triangle tri})} \Rightarrow L_{\triangle tri_k} \) is the perimeter of \( \triangle tri_k \), and \( D_{\triangle tri_k} \) is the area of \( \triangle tri_k \)
8. **if** \( r_k > R \) **then**
9. Delete \( \triangle tri_k \) from \( \triangle tri \)
10. **end if**
11. **End for**
12. Extract the boundary \( b \) for Delaunay triangle set \( \triangle tri \) based on region growing method
13. \( S.tri = \triangle tri \), \( S.bou = b \)
14. return \( S, V \)
15. **end procedure**

Figure 1. The overview of the propose method.
The shape of the flag in image 1 is irregular, and there are some shadow areas and folded regions. Our method samples the feature points \( Q \) in the shadows, folds and boundary areas of the flag in image 1 manually. The pixel coordinates of each sampling point \( Q_i \) in the image are set as \( X \) and \( Y \) axis coordinates of each vertex \( V_i \) in the reconstructed mesh. Based on the sampling point set \( Q \), the Alpha Shape method is employed to reconstruct a Delaunay mesh of the flag. We define a 2D mesh structure \( S \) to represent the simplex Delaunay triangle set \((S.tri)\) and boundary \((S.bou)\) of the virtual flag.

The mesh reconstruction method based on Alpha Shape and sampling points is described as Algorithm 1. A Delaunay mesh \( S \) is generated based on \( Q \) firstly. Then, the radius \( r_k \) of the circumcircle of each simplex Delaunay triangle \( \triangle v_i \) is used to optimize mesh \( \triangle tri \). Finally, the region growing method is employed to extract the boundary of \( tri \). Figure 1(d) presents an example of the reconstructed mesh based on Alpha Shape and sampling points.

**Algorithm 2. Mesh optimization algorithm**

**Input:** 2D mesh structure: \( S \); Vertex set: \( V \); Output: Depth map of image \( I \); \( D_i \)

1. procedure 3D mesh: \( T \);
2. \( \{ N, P \} = \text{ResFrontalDel}(S.tri, S.bou, V) \)
3. for each vertex \( N_i \) do
4. \( N_i^2 = D_i(N_i^1, N_i^2) \) ▶ Transform mesh into 3D with depth
5. end for
6. \( \{ N', P' \} = \text{linearSubdivision}(N, P) \)
7. \( T = \{ N', P' \} \)
8. return \( T \)
9. end procedure

The mesh \( S \) of the virtual flag is a planar coarse mesh, and it cannot represent the complete structure of the flag in image 1. The proposed method employs the Restricted Frontal-Delaunay method, which combines the classical Delaunay-refinement and advancing-front type approaches, to generate a new 2D mesh of the flag with a more reasonable distribution of internal vertices. According to the off-center Steiner points and the boundary of mesh \( S \), the Restricted Frontal-Delaunay method improved the quality of \( S \) in terms of shape and size. We define the new 2D mesh of the flag as \( T = \{ N, P \} \), where \( N \) is the array of vertices and \( P \) is the array of new triangles of mesh \( T \).

Subsequently, we transform the 2D mesh \( T \) into 3D space based on depth map \( D_i \). Although the mesh \( T \) has been optimized, mesh \( T \) does not display the details of the flag very well because the density of mesh \( T \) is not sufficiently high. The linear subdivision method is used to refine mesh \( T \) with high density. Algorithm 2 describes the entire process of optimizing the coarse mesh \( S \) into 3D refined mesh \( T \). Figure 1(e) presents an example of an optimized mesh from a single image.

**Drive the virtual flag to flutter from single image**

To simulate the fluttering of the flag mesh, the reconstructed 3D mesh \( T \) is transformed into a simple linear spring model \( TS = \{ TP, TL \} \). \( TP \) is the particle set, which is composed of position \( u \) and velocity \( v \), and \( TL \) is the link set. The particles are connected with links in linear spring, and the link is employed to constrain the displacement of each particle under internal and external forces. The edge \( E_r \) of each triangle \( P_i \) is defined as \( E_r = \{(N_i, N_j), (N_j, N_k), (N_k, N_i)\} \), and the transformed formula is described as equation (1). Figure 2 demonstrates the difference between the mesh and the linear spring model.

\[
\begin{align*}
TP &= \{ u, v \} \text{ with } u = \{ N_i^x, N_i^y, N_i^z \}, v = 0 \\
TL &= \text{unique}\{ E_r \}
\end{align*}
\]

In phase space \((u, v)\), a complex number \( z = u + iv \) is introduced to simplify the simple linear spring model.
Thus, based on Newton second law, the spring model is defined as equation (2), where \( z(t) = z_0 e^{-\frac{t}{\tau}} \). With a fixed time step \( h \), the velocity-position space between two consecutive states is approximated by equation (3).

\[
\begin{align*}
\dot{z} &= -iZ(t) \\
Z(0) &= Z_0 \\
\begin{pmatrix}
y_i \\ u_i
\end{pmatrix} &= \begin{pmatrix} 1 & -\frac{h}{\tau} \\ h & 1 - \frac{h^2}{\tau^2} \end{pmatrix} \begin{pmatrix} y_0 \\ u_0 \end{pmatrix}
\end{align*}
\]

Considering the existence of internal and external constraints when driving the flag to flutter, we use a vector \( C = f_{\text{stretch}}(i, \ast) + f_{\text{gravity}}(i) + f_{\text{air}}(i) + f_{\text{wind}}(i) \) to represent all constraints, such as stretch force between two particles, gravity force, a virtual air force and external wind force. We employ equation (4) to describe the constraints of particle \( i \) in the mesh. Figure 3 describes the force analysis of a vertex in the reconstructed mesh.

\[
\begin{align*}
f_{\text{stretch}}(i, \ast) &= \text{dis}_s - \text{dis}_i - L(i, \ast) \\
f_{\text{gravity}}(i) &= m_i g \\
f_{\text{air}}(i) &= -A \frac{v_i}{h} \\
f_{\text{wind}}(i) &= (W_{\text{wp}} m_i / h) \cdot W_{\text{dir}}
\end{align*}
\]

where \( \text{dis}_s \) is the displacement of particle \( \ast \) during the step \( h \), and \( L(i, \ast) \) is the length between particles \( \ast \) and \( i \). \( m_i \) is the mass of particle \( i \), which is set as a constant. \( A \) is also an arbitrary constant. \( W_{\text{wp}} \) is the speed of the external wind force, and \( W_{\text{dir}} \) is the direction of the external wind force. Finally, the object equation of driving flag fluttering based on the 3D mesh is described as follows:

\[
C(u + hv) + \nabla C(u + hv) h \Delta v = 0
\]

where \( v = v + \nabla v \) and \( u = u + hv \).

The proposed method employs a unified dynamics solver, which was described in the article, to solve the object equation (5). The unified dynamics solver uses the importance to specify the solution times of each constraint. Additionally, the order is employed to determine the solution sequence in each iteration process. Specifically, the unified dynamics solver solves the constraints vector \( C \) in a sequential manner one at a time in a Gauss-Seidel method. And it uses a line searching along a direction to satisfy each constraint, where the search direction is chosen by the gradient of the constraint. The gradient is calculated by the direction orthogonal to all transformations that modify the constraint. Finally, we use the Newton iteration to solve the equation (5). Therefore, based on equations (2) to (5), we can implicitly calculate the velocities and explicitly calculate the positions of all particles at time \( t \).

Based on the new position \( u \) of each particle, we can generate a new 3D mesh \( T' \) of the flag with the edge set \( E \). To create a video from a single image, the proposed method employs batch rendering technique to map the texture of the flag to the 3D mesh \( T' \) of the flag at time \( t \). The illumination recovery optimization method is used to remove the shadow of the flag in the input image before rendering the texture of the flag. Finally, the proposed method generates a flag fluttering video by combining a series of virtual frames.

**Experiments and analysis**

In this section, we describe the experiments conducted on single images that are accessed from the Internet. We analyze the advantages of the proposed method. We also present the video frames of a virtual fluttering flag created from two single images.

In this paper, we evaluate our method with some traditional mesh reconstruction methods, such as Restricted Frontal-Delaunay. The baseline of our experiments is the mesh based on depth map directly. We firstly evaluate the number of vertex and face of the reconstructed mesh and optimized mesh. We also evaluate the influence of input parameter \( R \) in Algorithm 1 on the performance of mesh reconstruction. This paper evaluates the visualization flag fluttering with different speed \( W_{\text{wp}} \) and direction \( W_{\text{dir}} \) of the virtual wind. And we compare the animation results with the animations generated by third-party tools, such as ‘Maya’.

However, there are some disadvantages when we employ the depth map of an image to generate the 3D structure directly. The first disadvantage is that it is very difficult to generate an accurate depth map from a single image. Each pixel of the flag is regarded as a vertex in the reconstructed mesh. Thus, the virtual mesh, which is generated from the depth map, is very
dense. The dense mesh will greatly increase the time cost of the solver, the instantaneous displacement and the velocity of each point in the virtual flag mesh.

Based directly on depth map $D_I$ (Figure 1) of image $I$, we can reconstruct the 3D mesh of the single image (Figure 1). The reconstruction result is shown in Figure 4(a). There are clearly some holes in Figure 4(a), and the mesh density of the reconstructed mesh is very high. Figure 4(b) is the simplification result of Figure 4(a), but its density is still very high. We count the vertices and faces of the reconstructed meshes based directly on depth map for comparison with our method, which are described in Table 1.

From the statistical results, we can infer that our method can minimize the number of vertices on the premise of guaranteeing a grid structure because we only sample pixels at the specified areas of the flag in the image. The results also explain why we do not employ the depth of each pixel to reconstruct mesh directly. The initial input parameters of the proposed method are summarized in Table 2. We will focus on the following two parameters: the radius $R$ and the virtual wind speed $W_{sp}$, which have the greatest impact on the flag fluttering animation.

**Mesh reconstruction**

We first sample the key points in the single image, as demonstrated in Figure 1(c). Then, Figure 5 shows the mesh reconstruction results based on the proposed method. Figure 5(a) presents the results based on Alpha Shape and sampling points, which is a 2D constructed mesh; Figure 5(b) shows the optimization results based on Restricted Frontal-Delaunay method, and Figure 5(c) is the final linear subdivision mesh of the flag from the single image. There are clear deformations on the edge of the reconstruction results (Figure 5(b) and (c)), which are caused by the depth map. There are some significant changes in the boundary of reconstruction meshes (Figure 5(b) and (c)) over the input image (Figure 1(a)) because the distance between any two vertices in a reconstructed triangle has changed based on the different depth of each vertex. Thus, how to reconstruct a mesh without deformation is our future research work. We also use a popular tool ‘Maya’ to simulate the mesh of flag in Figure 1(a), which are shown as (Figure 5(d) to (f)). The reconstruction meshes in Figure 5(d) to (f) are shown perfect boundary shapes of the virtual flag. And in especial the Figure 5(d) and (e) do not guarantee that the triangles are as average as possible. However, the shape of each triangle has high in similarity the reconstruction meshes based on the proposed method. In other words, the virtual forces can act on each vertex in Figure 5(a) to (c) to generate fluttering Flag videos as equally as possible.

We also count the number of vertices and faces of each reconstructed mesh in Figure 5, and the results are shown in Figure 6. The number of each vertex and face are very close between our method and Maya tool in Figure 6. This illustrates that the reconstructed meshes based on the proposed method have certain usability and can better restore the shape of the flag. And the number and distribution of final optimized triangles can guarantee that the proposed method can more evenly drive each part of the flag in the image.

The position of each vertex in the meshes has changed compared with Figure 5(a) and (b), as shown in Figure 7. Based on the Restricted Frontal-Delaunay method, the optimized flag mesh covers more regions in the image relative to the initial rough mesh.

**Table 1.** The vertex and face numbers of reconstructed.

| Mesh based on depth directly | 73579 | 65897 | 67594 | 60657 |
|-----------------------------|-------|-------|-------|-------|
| Dilution Mesh               | 3164  | 1507  | 5842  | 2830  |
| Ours                        | 737   | 373   | 1312  | 648   |

**Table 2.** The initial parameters.

| Notation | Meaning in our algorithm |
|----------|--------------------------|
| $I$      | Original single image    |
| $R$      | The probe radius in Alpha Shape |
| $D_I$    | The depth map of $I$     |
| $W_{sp}$ | The speed of virtual wind|
| $W_{dir}$| The direction of virtual wind |
There is another input parameter $R$ in Algorithm 1, which is employed to determine whether the constructed triangle is an internal triangle with the range of all sampling points. We set $R$ with different values to determine a proper value for the input single image (Figure 1(a)).

Figure 8 analyzes the quality of the reconstructed mesh based on the proposed method with a different value of $R$. We analyzed the number of reconstructed triangles based on Algorithm 1 with the different value of $R$ and fixed sampling points in the image, as shown in Figure 8 (a). The number of triangles in the reconstructed mesh, which concentrated at approximately 240, increases with increasing $R$. To determine the best $R$, we then calculate the area variance of the triangle, which can indicate the area difference of all triangles for each reconstructed mesh Figure 8(b)). The area variance tends to stabilize at approximately 70 when $R$ is in $[90,130]$. To guarantee that each part of the flag can be as large as possible in the same size of a triangle mesh when the flag is driven to flutter, $R$ must be set as the value when the area variance of the reconstructed mesh as large as possible on average. We use Algorithm 2 to generate a more suited mesh for the original image based on the result of Algorithm 2. Thus, we also count the numbers of vertices and
Figure 7. The different vertex’s position of each mesh.
(a) The number of triangle of reconstruction meshes. (b) The area variance of all triangles. (c) The number of vertex and triangle of optimized reconstruction meshes.

Figure 8. Mesh reconstruction analysis with difference $R$.

Figure 9. The reconstruction 3D mesh with texture in different angle views.
Figure 10. The generated frames of virtual fluttering flag.

Figure 11. The fluttering frames of flag with different speeds of virtual wind.
triangles for each optimized mesh (Figure 8(c)), which are most concentrated when $R$ is in $\left[90, 130\right]$. Based on the observations of Figure 8(a) to (c), the value of $R$ is set to 130, which can construct a more stable mesh in our experiments.

The final reconstructed 3D mesh from the single image based on the proposed method is shown in Figure 9, which was rendered with texture after removing shadow and shown with different view angles. As indicated by the reconstructed 3D mesh result, the proposed mesh reconstruction method can generate an approximate mesh from a single image. Although the reconstructed mesh does not completely reflect all the details of the flag in the image, it reconstructed the entire structure, partial shadows and folds in the same image.

**Fluttering flag video reconstruction**

Based on the reconstructed 3D mesh of the flag from the single image, we employ the simple spring model to drive the fluttering of flag’s mesh. We initialize the speed $W_{sp}$ and direction $W_{dir}$ of the virtual wind before generating the flag fluttering video. To simplify the virtual video production process, the $W_{dir}$ is set as a constant: $[1, 0, 1]$. In other words, the direction of the virtual wind is a $+45^\circ$ angle between the X and Z axes. The speed $W_{sp}$ of the virtual wind is set from 0 to 40. Figure 10 presents the frames of the virtual fluttering flag ($W_{sp} = 40$, $W_{dir} = [1, 2, 1]$), and the reconstructed 3D mesh is shown in Figure 9.

Figure 11 describes the animation results from Figure 1(a) with different wind speeds based on the proposed method. We extract three frames (50, 100, 150) from the flag fluttering video with different $W_{sp}$. From the results, we can observe that the flag fell down because of gravity when $W_{sp} = 0$. Additionally, the magnitude of flag fluttering increases with the increasing $W_{sp}$.

To further verify the applicability of the proposed method, we generate another flag fluttering video (Figure 12) from a second single image, which is also accessed from the Internet. Thus, we can confirm that the proposed method is capable of producing a virtual flag fluttering video from a single image.

**Conclusions**

In this paper, a smart method for generating a flag fluttering video from a single image is proposed. The depth information of the image is calculated using the (DCNF) method, which is an inaccurate estimation value. We sample some pixels in the image to generate a rough mesh of the flag based on Alpha Shape. To generate a more uniform and reasonable mesh distribution, the Restricted Frontal-Delaunay method is employed to optimize the rough mesh. Then, the 2D reconstructed mesh is mapped into 3D space by the depth information of each vertex. Finally, a continuous video of the flag fluttering from the image is generated using a simple linear spring model.

Although the reconstructed 3D mesh of the flag based on the proposed method cannot completely reflect all the details of the flag in the image, our method can generate a satisfactory flag fluttering video. The method also has the characteristics of simple processing and fast speed. In the future, we will improve the quality of the reconstructed mesh based on deep learning, and a more complex spring model will be used to process the interrelation between each vertex of the flag.

**Declaration of Conflicting Interests**

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

**Funding**

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by National Natural Science Foundation of China under Grants 61562025 and 61962019, and supported by Hubei technical innovation special project (key project) of China under Grant 2018AKB035.

**ORCID iD**

Tao Hu  
https://orcid.org/0000-0002-6586-231X

**References**

1. Castillo C, De S, Han X et al. Son of zorn’s lemma: Targeted style transfer using instance-aware semantic
segmentation. In 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA (2017.3.5-2017.3.9), pp. 1348–1352. IEEE.
2. Chai M, Shao T, Wu H, et al. AutoHair: fully automatic hair modeling from a single image. ACM Trans Graph 2016; 35: 1–12.
3. Saxena A, Sun M and Ng AY. Make3D: learning 3D scene structure from a single still image. IEEE Trans Pattern Anal Mach Intell 2009; 31: 824–840.
4. Shelhamer E, Barron JT and Darrell T. 2015. Scene intrinsics and depth from a single image. In: IEEE international conference on computer vision workshop, Santiago, 2015, pp.235–242.IEEE.
5. Liu F, Shen C, Lin G, et al. Learning depth from single monocular images using deep convolutional neural fields. IEEE Trans Pattern Anal Mach Intell 2016; 38: 2024–2039.
6. Qin H, Li X, Wang Y, et al. Depth estimation by parameter transfer with a lightweight model for single still images. IEEE Trans Circuits Syst Video Technol 2017; 27: 748–759.
7. Hairer E, Lubich C and Wanner G. Geometric numerical integration: structure-preserving algorithms for ordinary differential equations. Ser Comput Math 2006; 25: 805–882.
8. Schiopu I and Munteanu A. Deep-learning-based depth estimation from light field images. Electron Lett 2019; 55: 1086–1088.
9. El-Shazly EH, Zhang X and Jiang J. Improved appearance loss for deep estimation of image depth. Electron Lett 2019; 55: 264–266.
10. Ganapathi II, Prakash S, Dave IR, et al. Ear recognition in 3D using 2D curvilinear features. IET Biometr 2018; 7: 519–529.
11. Wang X, Yan K and Liu Y. Automatic geometry calibration for multi-projector display systems with arbitrary continuous curved surfaces. IET Image Process 2019; 13: 1050–1055.
12. Chuang Y-Y, Goldman DB, Zheng KC, et al. Animating pictures with stochastic motion textures. ACM Trans Graph 2005; 24: 853–860.
13. Sun M, Jepson AD and Fiume E. Video input driven animation (vida). IEEE Int Conf Comput Vision 2008; 1: 96–103.
14. Xu X, Wan L, Liu X, et al. Animating animal motion from still. ACM Trans Graph 2008; 27: 1–8.
15. Jhou WC and Cheng WH. Animating still landscape photographs through cloud motion creation. IEEE Trans Multimedia 2016; 18: 4–13.
16. Engel J, Schöps T and Cremers D. LSD-slam: large-scale direct monocular slam. In: European conference on computer vision, Zurich, Switzerland, 2014, pp. 834–849. Berlin: Springer.
17. Mur-Artal R, Montiel JMM and Tardós JD. ORB-slam: a versatile and accurate monocular slam system. IEEE Trans Robot 2015; 31: 1147–1163.
18. Jancosek M and Pajdla T. Multi-view reconstruction preserving weakly-supported surfaces, CVPR 2011, Providence, RI, 2011, pp. 3121–3128. IEEE.
19. A. Owens, J. Xiao, A. Torralba and W. Freeman, Shape Anchors for Data-Driven Multi-view Reconstruction, 2013 IEEE International Conference on Computer Vision, Sydney, NSW, 2013, pp. 33-40. IEEE.
20. M. Goesele, N. Snavely, B. Curless, H. Hoppe and S. M. Seitz, Multi-View Stereo for Community Photo Collections, 2007 IEEE 11th International Conference on Computer Vision, Rio de Janeiro, 2007, pp. 1-8.IEEE.
21. Henry P, Krainin M, Herbst E, et al. RGB-d mapping: using kinect-style depth cameras for dense 3D modeling of indoor environments. Int J Robot Res 2012; 31: 647–663.
22. Fan H, Su H and Guibas L. A Point Set Generation Network for 3D Object Reconstruction from a Single Image, 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 2463-2471. IEEE.
23. Bridson R, Fedkiw R and Anderson J. Robust treatment of collisions, contact and friction for cloth animation. ACM Trans Graph 2002; 21: 594–603.
24. Narain R, Samii A and O’Brien JF. Adaptive anisotropic remeshing for cloth simulation. ACM Trans Graph 2012; 31: 1–10.
25. Jin N, Lu W, Geng Z, et al. Inequality cloth. In: Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation, Los Angeles California, USA, 2017.
26. Chen L, Ye J, Jiang L, et al. Synthesizing cloth wrinkles by CNN-based geometry image superresolution. Comput Anim Virtual Worlds 2018; 29: e1810.
27. Silberman N, Kohli P, Hoiem D, et al. Indoor segmentation and support inference from RGBD images. Eur Conf Comput Vision 2012; . Florence, Italy, pp.746-760. Berlin Heidelberg: Springer .
28. Xu X and Harada K. Automatic surface reconstruction with alpha-shape method. Vis Comp 2003; 19: 431–443.
29. Engwirda D. JIGSAW-Geo (1.0): locally orthogonal staggered unstructured grid generation for general circulation modelling on the sphere. Geosci Model Dev 2017; 10: 2117–2140.
30. Engwirda D and Ivers D. Off-centre Steiner points for Delaunay-refinement on curved surfaces. Comput Aided Design 2016; 72: 157–171.
31. Stam J. Nucleus: Towards a unified dynamics solver for computer graphics, 2009 11th IEEE International Conference on Computer-Aided Design and Computer Graphics, Huangshan, 2009, pp. 1-11. IEEE.
32. Moreau JJ. On unilateral constraints, friction and plasticity. In : Capriz G and Stampacchia G (eds) New variational techniques in mathematical physics. Berlin: Springer, 2011, pp. 171–322.
33. Zhang L, Yan Q, Liu Z, et al. Illumination decomposition for photograph with multiple light sources. IEEE Trans Image Process 2017; 1–1.
34. autodesk. Maya free trial, https://www.autodesk.com/ (2000, accessed 23 December 2020).