Motion Capture and Retargeting of Fish by Monocular Camera

Xiangfei Meng
Beihang Univeristy, China

Junjun Pan*
Beihang Univeristy, China

Hong Qin
Stony Brook University, USA

***Abstract*—Accurate motion capture and flexible retargeting of underwater creatures such as fish remain to be difficult due to the long-lasting challenges of marker attachment and feature description for soft bodies in the underwater environment. Despite limited new research progresses appeared in recent years, the fish motion retargeting with a desirable motion pattern in real-time remains elusive. Strongly motivated by our ambitious goal of achieving high-quality data-driven fish animation with a light-weight, mobile device, this paper develops a novel framework of motion capturing and retargeting for a fish. We capture the motion of actual fish by a monocular camera without the utility of any marker. The elliptical Fourier coefficients are then integrated into the contour-based feature extraction process to analyze the fish swimming patterns. This novel approach can obtain the motion information in a robust way, with smooth medial axis as the descriptor for a soft fish body. For motion retargeting, we propose a two-level scheme to properly transfer the captured motion into new models, such as 2D meshes (with texture) generated from pictures or 3D models designed by artists, regardless of different body geometry and fin proportions among various species. Both motion capture and retargeting processes are functioning in real time. Hence, the system can simultaneously create fish animation with variation, while obtaining video sequences of real fish by a monocular camera.**

Keywords — Fish Animation, Markerless Motion Capture, Monocular Camera, Motion Retargeting

I. Introduction

Motion capture (or Mo-cap) has been widely used in video game production and film industry for the last two decades, with the goal of creating natural character animation and special effects. Realistic motion data of an actor can be captured by a mo-cap system and then retargeted to new characters, creating animation with the same motion as the actor but with new appearances. Nonetheless, traditional mo-cap systems suffer from noisy motion data [1] [2] and specialized high-end devices which are too expensive to afford by the general public [3]. Moreover, it is difficult to attach markers to certain types of characters such as underwater creatures (e.g., fish).

To address the limitations above, we develop a markerless motion capture technique to record the motion of an underwater fish in this paper. Since the most movement of a fish takes place in the horizontal plane, we simply employ a monocular camera to record the fish movement from the top of a fish tank. The swimming fish (i.e., foreground region) is segmented by a background subtraction algorithm. Based on the extracted fish contour, the head and tail can be located successively. However, the accurate medial axis of a fish body serving as a spine is difficult to represent, due to the asymmetric motion of the fish fins. To solve this problem, we employ elliptical Fourier coefficients [4] to convert the fish contour from the original physical domain to the frequency domain, and then, reconstruct it with fewer coefficients. Similar to the low pass filter, it gives rise to a smoothed contour. Then a smooth medial axis, which is considered as the deformable spine, can be generated for the soft fish body. Finally, the remaining feature points can be located based on the medial axis and the original contour. By retargeting such feature points as fish motion to 2D/3D meshes with texture information, the final fish animation can be obtained. With our proposed data-driven method, even the general public without specially-trained artistic skills is capable of creating fish animation with low-cost mobile devices, such as a cellphone with a camera.

Essentially, fish motion can be decomposed into two parts: the global motion including position and orientation, and the local motion which is the deformation under local coordinates. The estimation and recovery of global motion are straightforward. As for the local deformation, simply scaling the motion w.r.t. the skeleton according to the fish body ratio will cause unnatural geometry distortion, especially in the vicinity of the fins area. To address this problem, we propose a two-level motion retargeting scheme, in which the local shape transferring process is separated into two steps. At the first step, we only transfer the body motion into the target model, during which the junction points between the fish body and two fish fins are regarded as a part of the fish body and can be accurately retargeted. Afterwards, we transfer the fin motion through the junction points, serving as both local control points and relative positions of fin motion. With this two-level motion retargeting scheme, we can properly transfer the motion of a real fish into a target model. In addition, our system also allows users to edit the retargeted frames to interactively fine tune the final animation. In particular, the innovative contributions of our research can be summarized as follows:

- We develop a mo-cap technique for fish animation with low-cost device. Using a monocular camera, the motion sequences of a real fish are captured and retargeted to a 2D/3D fish or fish-like model in real time. With comprehensive functionalities, our system also supports interactions from users to edit retargeted frames for final animation production.
- The smooth medial axis serving as the description of a

soft fish body is hard to acquire due to the asymmetry of the fins during fish swimming. We incorporate elliptical Fourier coefficients into the contour-based feature extraction. Then, it is possible to reconstruct the contour with low-frequency harmonics, which can effectively attenuate asymmetry of the fins while preserving the global shape of a fish body. Finally, a preferable medial axis can be extracted.

- We propose a two-level motion retargeting scheme to transfer the motion from an original fish sequence acquired in a video to new models. The major challenge is to handle various body proportions and fin locations. In particular, we decompose the local motion retargeting process into two steps. First, the motion of a fish body is transferred to the target model, with the positions of junction points determined. At the second step, the motion of fins can be accurately retargeted with the junction points, serving as both local control points and relative positions of the fin motion.

II. Related Work

Motion capture and motion retargeting have been studied widely in computer vision and computer graphics. However, both of them remain open and challenging problems. Given the significant literature in these areas, we focus on the most relevant researches.

Video-based Motion Capture. The mainstream mo-cap system usually employs a number of synchronized cameras to acquire multiple views of the character with markers or sensors, which has been well studied in the existing literature [5]. Motion capture from the video is more challenging for the lack of depth information and occlusion of feature points or markers. Most video-based approaches estimate the pose by searching from the state space [6] or relying on Bayesian filtering with the prior models of dynamics [7] and focus on human or articulated characters only [8] [9] [10]. In contrast, our mo-cap system is designed for fish or fish-like creatures. It is unworthy to pre-define the motion pattern for particular non-human species. Therefore, we propose a feature-based markerless motion capture approach which saves the costs of prior dynamics construction or state-space searching.

In the field of motion capture for animals, Ju *et al.* [11] present a mo-cap system for a bird but their method is marker-based and needs multiple high-speed video cameras. Wilhelms *et al.* [12] propose an interactive video-based mo-cap system for character animation. They identify features and establish feature relationships from frame to frame, but their method is developed only for articulated characters such as horses. Lee *et al.* [13] present a video-based fish animation generating system which is the most similar work to ours, but their work relies on multi-view cameras and doesn't take the motion of fins into consideration. Yu *et al.* [14] present a synthetic motion capture to render plenty of fish at an interactive rate. However, the motion data they use is generated from the simulation of the biomechanical model, while we directly use captured videos as a reliable data source.

Video-based motion capture can be regarded as extracting motion from image sequences. There are some classic approaches for point tracking [15] or object tracking [16]. Shi *et al.* present the famous KLT tracking algorithm [15] to track good features which are selected under the principle that a good feature is the one that can be tracked well. The features selected by KLT tracker are mostly corners while the body of fish is too flexible to obtain stable tracking across a large number of frames. Comaniciu *et al.* [16] propose the mean-shift tracking approach for real-time non-rigid object tracking, which is used in the mo-cap system presented in [8] to track the motion of a human body. However, the mean-shift algorithm calculates the shift vector based on the histogram of the object region, which makes it hard to track the patch of a fish with its self-similar texture. In contrast, our contour-based feature extraction avoids the direct tracking to a specific point or patch, thus is able to provide robust motion information.

Motion Retargeting. Motion retargeting is a classic problem which aims to retarget motion from one character to another while keeping styles of the original motion. Most existing work is either offline or requires a large database of example motion. An offline method presented by Gleicher [17] uses spacetime constraints on example motion. The paper considers different segment lengths but identical structure or characters with fewer degrees of freedom (DOF). Similar limitations exist in many other offline methods. Shin *et al.* [18] present an online method using inverse kinematics(IK), but only take identical skeletal topologies into consideration. Kulpa *et al.* [19] propose an approach to support different numbers of bones in limbs by using IK in real time, but only on humanoid topologies. Popović [20] applies the principles of physical-based animation and constraint optimization formulation on motion data, but like most other approaches, retargets the motion to articulated figures only. An interesting method proposed by Bregler [21] is able to retarget the motion from a cartoon character to another, yet it needs the user to define each key-shape motion to synthesize the final animation. Hornung *et al.* propose a retargeting approach [22] similar to ours that can animate photos of 2D characters. However, the motion types are limited and the animation characters must be humanoid.

In terms of fish motion retargeting, it's hard to treat the fish as an articulated character and represent it with a traditional skeleton model, due to the flexibility of the fish body and the variable locations of the fins. The two-level retargeting scheme we propose treats the fish topology as two parts. The body part can handle the flexibility and the fin part is able to deal with the various fin locations.

III. System Overview

Our motion capture and retargeting technique aims at providing a framework to drive a 2D/3D fish or fish-like model to swim lively with the same motion style as the real one in the video. Fig. 1 shows the pipeline.

Motion Capture. After acquiring a swimming fish video, we employ an adaptive background subtraction method: ViBe [23] to detect the foreground of each frame. To eliminate

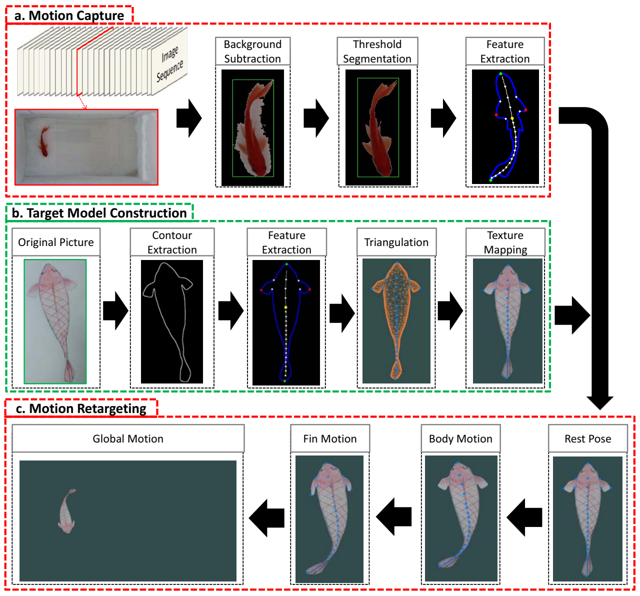

Fig. 1. Flow chart of our technique. (a)Motion capture of a swimming fish in water. (b) Target model is constructed from a hand-drawn cartoon fish. (c)Body motion, fin motion, and global motion are retargeted sequentially. Both (a) and (c) are processed synchronously in real time.

salt and pepper noise, a morphological filter is applied to the binary mask of the foreground pixels. Then the system marks the salient connected components as the foreground objects. Once the user selecting one object, the system will keep tracking on it and regard it as the swimming fish. While the fish is being tracked, its motion will be continuously recorded by the contour-based feature extraction procedure and then delivered to the motion retargeting module.

Target Model Construction. In Fig. 1(b), the target model is constructed from a hand-drawn cartoon fish. The aim is generating a target 2D/3D model with several control points to achieve the animation. This task is processed off-line and only once. Given a picture containing a cartoon fish, we preprocess the image to acquire the contour, then the control points will be extracted by the contour-based feature extraction process. Finally, the fish model is generated by constrained Delaunay triangulation [24]. For 3D model, we take a snapshot on the top view of the fish model, extract the control points and finally recover them to the 3D scene.

Motion Retargeting. The control points of the fish model can be divided into two groups: body control points and fin control points. The recorded motion consists of global motion and local motion. At the first stage, we transfer the local motion to the body control points of the fish model, then deform the model with body control points only. Thus we can obtain the positions of junction points, which are also deformed as a local part of the fish body. At the second stage, we transfer the local motion to the fin control points with the help of junction points, then deform the fish model with all control points. Afterwards, the final retargeted model is obtained by applying rotation and translation transformation according to the global motion. For shape deformation, we choose a moving-least-squares (MLS) based approach [25] due to its high efficiency for the closed-form solution and low distortion stemming from the least square error.

IV. Motion Capture

In each frame, the foreground pixels are identified with binary mask from ViBe [23]. The salt and pepper noise is eliminated through a morphological opening operation on the binary mask. Then we identify the connected components and remove the components whose areas are too small, leaving the salient components as foreground objects.

The system will start tracking once user selecting one object. As shown in Fig. 1(a), an adaptive threshold image segmentation method [26] is employed to eliminate the noise caused by shadow . After we get an accurate contour of the fish, the motion will be captured by our contour-based feature extraction process. Here we list the motion attributes (position or orientation) in Table I and label them in Fig. 2, where u_num and l_num are the sampling numbers of the upper segment and lower segment of the medial axis respectively.

TABLE I
Fish Motion Attributes

Attribute	Explanation
center	Fish center
m_dir	Major direction, representing the fish orientation
head	Fish head
tail	Fish tail
l_fin	The left pectoral fin tip
r_fin	The right pectoral fin tip
jun_tl	The top-left junction point
jun_tr	The top-right junction point
jun_bl	The bottom-left junction point
jun_br	The bottom-right junction point
$\textbf{up\_axis}_{\{1,2,\ldots,u\_num\}}$	Sample points in the upper segment of the medial axis, from center towards head
$\textbf{low\_axis}_{\{1,2,\ldots,l\_num\}}$	Sample points in the lower segment of the medial axis, from center towards tail

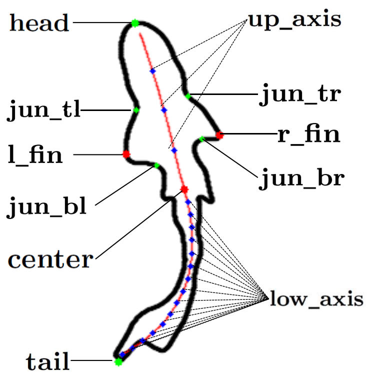

Fig. 2. Motion attributes of a fish.

The feature extraction process can be divided into four steps: global motion estimation, head and tail recognition, medial axis extraction, and fin recognition.

A. Global Motion Estimation

The aim of this step is to estimate the center position (**center**) and the orientation (**m_dir**) of the fish. We use PCA method to process the coordinates of all pixels inside the closed contour to obtain the mean value, eigenvalues and eigenvectors. The mean value that representing the barycenter is assigned to **center** temporarily. As the barycenter will sometimes drift apart from the medial axis when the fish forms "C-shape", we'll update the value of **center** later after the medial axis is extracted.

As for orientation estimation, we firstly choose the eigenvector with the larger eigenvalue as **m_dir**. However, it is possible the negative direction of the fish head. So the direction of the eigenvector will be adjusted by the orientation of the last frame so that the angle of the orientations between two consecutive frames won't be larger than $\pi/2$.

B. Head and Tail Recognition

The recognition of **head** and **tail** is straightforward. At first, we calculate all the distances between contour points and the barycenter. The point with the largest distance is treated as **tail**. Then we divide the contour point sequence into two equal-length segments such that **tail** is the midpoint of the one, leaving **head** in the other. Therefore we can locate **head** as the point with the largest distance in its' own segment. Occasionally, there could be a chance to misidentify **head** and **tail** in reverse order. In case of that, we will use the orientation **m_dir** as a calibration. For general purpose, considering some types of fish that have two tips in caudal fins, we will detect whether there is another local max-distance point near **tail** as the other tip. Then **tail** will be updated as the midpoint of the two tips if the other tip is found.

C. Medial Axis Extraction

The core part of the motion is the fish medial axis, which is drawn in red in Fig. 2. At the beginning, we divide the closed contour into two segments, which is the two point sequences from **head** to **tail** in different ways. Without loss of generality, assume the number of points in the two sequences is $l1$ and $l2$ with $l1 \leq l2$. Then the coordinates of the two sequences can be represented by $\mathbf{p}_{\{1,2,...l1\}}$ and $\mathbf{q}_{\{1,2,...l2\}}$. And the medial axis can be given by

$$\mathbf{m}_i = \frac{\mathbf{p}_i + \mathbf{q}_{\lfloor \frac{l2}{l1} i \rfloor}}{2}, \; i = 1, 2, ..., l1. \tag{1}$$

However, this strategy has one problem. As we can see in Fig. 3(a) and (e), the fish contour is not smooth enough thus is likely to produce a jagged axis. Under this circumstance, we believe that a contour containing only low-frequency information is more suitable to express the body shape, thus can extract a more desirable medial axis. Spatial filter could be a choice, but the parameters of the filter template need to be set elaborately and the convolution process is time-consuming.

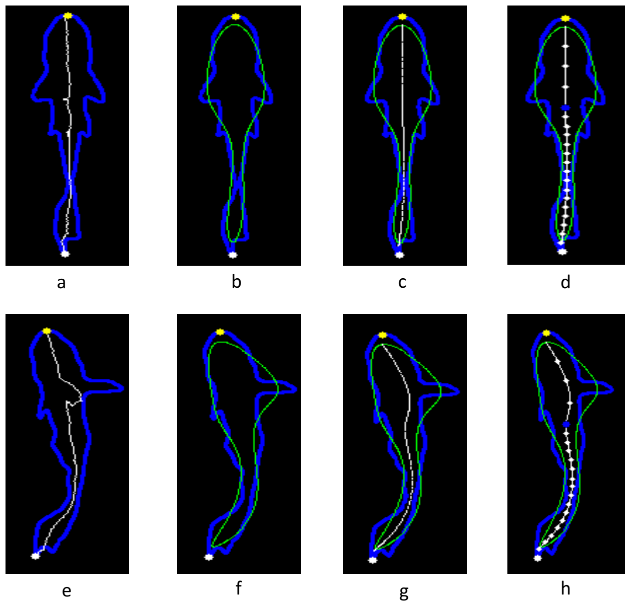

Fig. 3. The functionality of elliptical Fourier coefficients in medial axis extraction. (a)Jagged medial axis extracted directly from the original contour. (b)Smooth contour reconstructed by using elliptical Fourier coefficients. (c)Smooth medial axis extracted from the reconstructed contour. (d)Sampled points in the medial axis. (e)Biased medial axis caused by the asymmetric fins. (f)Asymmetry is largely eliminated by the reconstructed contour. (g)Smooth medial axis with little bias. (h)Sampled points in the medial axis.

Here we employ the elliptical Fourier coefficients technology to handle it.

Elliptical Fourier coefficients model closed contour as sums of elliptical harmonics. Each harmonic is described by four coefficients, interpreted geometrically as major axis length, minor axis length and the orientation of the ellipse. As shown in [4], we regard the contour as two periodic functions $x(t)$ and $y(t)$. For example, $x(t)$ can be written as

$$x(t) = a_0 + \sum_{k=1}^{\infty}(a_k \cos\frac{2k\pi t}{T} + b_k \sin\frac{2k\pi t}{T}), \tag{2}$$

where T is the perimeter of the contour, $t = 2\pi l/T$, l is the arc length from a preset starting point, and the coefficients can be calculated as

$$a_0 = \frac{1}{T}\sum_{p=1}^{K}\frac{\Delta x_p}{2\Delta t_p}(t_p^2 - t_{p-1}^2) + \zeta_p(t_p - t_{p-1}), \tag{3}$$

$$a_k = \frac{T}{2k^2\pi^2}\sum_{p=1}^{K}\frac{\Delta x_p}{\Delta t_p}(\cos\frac{2k\pi t_p}{T} - \cos\frac{2k\pi t_{p-1}}{T}), \tag{4}$$

$$b_k = \frac{T}{2k^2\pi^2}\sum_{p=1}^{K}\frac{\Delta x_p}{\Delta t_p}(\sin\frac{2k\pi t_p}{T} - \sin\frac{2k\pi t_{p-1}}{T}), \tag{5}$$

with K being the number of points in the contour, $\Delta x_p = x_p - x_{p-1}$, $\Delta t_p = \sqrt{(\Delta x_p)^2 + (\Delta y_p)^2}$, and

$$\zeta_p = \sum_{j=1}^{p-1}\Delta x_j - \frac{\Delta x_p}{\Delta t_p}\sum_{j=1}^{p-1}\Delta t_j. \tag{6}$$

$y(t)$ can be defined in terms of the coefficients c_0, c_k and d_k similarly.

After obtaining the coefficients from the original contour, we use the first S harmonics to reconstruct a smooth contour according to Eq. 2 with the same point number as origin. The reconstruction result is shown in Fig. 3(b) and (f) with S being 5. Afterwards, we use the reconstructed contour to update point sequences $\mathbf{p}_{\{1,2,...l1\}}$ and $\mathbf{q}_{\{1,2,...l2\}}$. Then the medial axis can be extracted according to Eq. 1, as shown in Fig. 3(c) and (g), which becomes much more smooth.

Since more feature points are required to represent the curved body deformation in the lower part of the fish, we sample the medial axis with different densities. We uniformly sampled u_num points in the segment of the medial axis from center point towards head ($\mathbf{up\_axis}_{\{1,2,...,u\_num\}}$), and l_num points in the segment from center point towards tail ($\mathbf{low\_axis}_{\{1,2,...,l\_num\}}$). The result is illustrated as Fig. 3(d) and (h) with u_num begin 3 and l_num being 15.

D. Fin Recognition

Besides describing the body shape, the medial axis can offer a great help to locate the fins. We divide the contour into two segments with **head** and **tail** as the breakpoints, then find the point with the largest distance to the medial axis in each segment as the left or right pectoral fin's tip (**l_fin** and **r_fin**). Note that we only extract the pectoral fins of the fish, for the pelvic fin and dorsal fin are usually occluded by the fish body thus can hardly be recognized consistently.

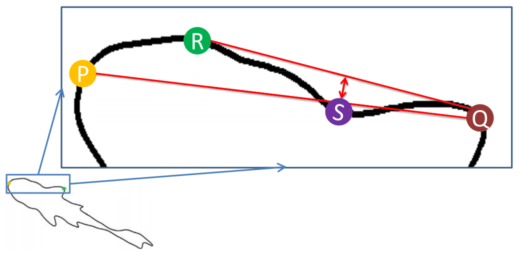

Fig. 4. The recognition process of **jun_tr**. P denotes the head, Q denotes the right fin tip, R is an assistant point, and S denotes the recognized **jun_tr**.

The remaining task is to locate four junction points (Fig. 2). The method to find each junction point is basically the same, so here we only describe how to find the upper junction between right fin and body (**jun_tr**). For the contour segment from **head** to **r_fin** (Fig. 4), supposing P represents **head** and Q represents **r_fin**, we search a point R on the contour segment between P and Q to ensure $\angle PQR$ is the maximum. Then the position of the point among the segment $\widetilde{QR}$ with the largest distance to the line $\overrightarrow{QR}$ is regarded as **jun_tr**.

V. Motion Retargeting

Before motion retargeting, We need to construct a target model from 2D pictures or 3D meshes for retargeting. As shown in Fig. 1, the 2D target model is constructed from a hand-drawn cartoon fish. We use a Canny edge detector to generate an edge image in binary mask, from which the contour is detected. Control points can be extracted from the contour-based feature extraction process which is basically the same as the detail in section IV. The mesh is generated from the contour by constrained Delaunay triangulation [24]. The texture of the model is also attached (Fig. 1(b)).

For 3D fish model, we can directly import the mesh and texture into our system. In order to generate the positions of control points, we use PCA to calculate three principle directions then apply a translation and rotation to the model, making it lying on the x-y plane, with its center at origin, head pointing to the positive y axis. A 2D snapshot is taken at the top view of the model, from which control points can be extracted with planar coordinates. Then we recover the control points to the 3D scene with $z = 0$.

The retargeting part consists of three stages—body motion, fin motion and global motion. The first two stages are the core part of the two-level retargeting scheme we proposed, and the last stage aims to retarget the global motion.

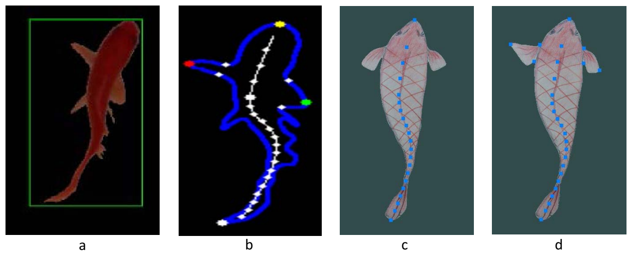

Fig. 5. Our two-level motion retargeting scheme. (a)The tracked fish. (b)Extracted contour and feature points. (c)The deformed implicit model with body control points. (d)The deformed explicit model with all control points.

In fact, we have two fish models constructed after target model construction. One is an implicit model which only works during body motion retargeting. The other is an explicit model which is utilized in the fin motion retargeting and displayed on the screen finally. The implicit model only has body control points while the explicit model contains all control points. In addition, the four junction points are added to the implicit model as ordinary vertices, thus can be transformed during body motion retargeting. For explanation, we define the implicit model I and explicit model E:

$$I = < V, C >, E = < V, C > \tag{7}$$

where V denotes the vertices

$$I.V = \{\mathbf{v}; \mathbf{jun\_tl}, \mathbf{jun\_tr}, \mathbf{jun\_bl}, \mathbf{jun\_br}\}, E.V = \{\mathbf{v}\}$$

and C denotes the control points

$$\begin{aligned} I.C =& \{\mathbf{center}, \mathbf{head}, \mathbf{tail}, \mathbf{up\_axis}, \mathbf{low\_axis}\} \\ E.C =& \{\mathbf{center}, \mathbf{head}, \mathbf{tail}, \mathbf{up\_axis}, \mathbf{low\_axis}, \\ & \mathbf{jun\_tl}, \mathbf{jun\_tr}, \mathbf{jun\_bl}, \mathbf{jun\_br}, \mathbf{l\_fin}, \mathbf{r\_fin}\} \end{aligned} \tag{8}$$

and $\mathbf{v}$ denotes the vertices generated from the triangulation process. Meanwhile, we have the recorded motion M

$$\begin{aligned} M =& \{\mathbf{center}, \mathbf{m\_dir}, \mathbf{head}, \mathbf{tail}, \mathbf{up\_axis}, \mathbf{low\_axis}, \\ & \mathbf{jun\_tl}, \mathbf{jun\_tr}, \mathbf{jun\_bl}, \mathbf{jun\_br}, \mathbf{l\_fin}, \mathbf{r\_fin}\}. \end{aligned} \tag{9}$$

To deform the target fish model, the motion structure of the target fish should be analyzed. Here we use a simplified curve skeleton L to represent the structure of the fish model.

$$L = \{body\_len, head\_len, tail\_len, l\_fin\_len, r\_fin\_len, up\_len_{\{1,2,...,u\_num\}}, low\_len_{\{1,2,...,l\_num\}}\} \quad (10)$$

The construction of L is described in Algorithm 1, which is executed only once after the target model is constructed.

Algorithm 1 Simplified Curve Skeleton Construction

```
Input: Explicit model $E$
Output: Bone length $L$
1: Compute $head\_len$ and $tail\_len$ of $L$ as the distances between $\mathbf{head}$, $\mathbf{tail}$ and the uppermost, the lowermost ending sample points in the medial axis of E.
2: Compute $up\_len$ and $low\_len$ of $L$ as the piecewise lengths of the sampled points in the medial axis of E.
3: Compute $l\_fin\_len$ and $r\_fin\_len$ of $L$ as the length between $E.C.\mathbf{l\_fin}$ and $E.C.\mathbf{jun\_tl}$ and the length between $E.C.\mathbf{r\_fin}$ and $E.C.\mathbf{jun\_tr}$
4: $L.body\_len = \sum_{i=1}^{u\_num} up\_len_i + \sum_{i=1}^{l\_num} low\_len_i + L.head\_len + L.tail\_len$
```

A. Body Motion Retargeting

The details of implicit model deformation are described in Algorithm 2. We choose angular variance as a motion parameter transferring between feature points of the actual fish and control points of the fish model.

Algorithm 2 Implicit Model Deformation

```
Input: Implicit model $I$, bone length $L$ and recorded motion $M$
Output: The deformed implicit model $I$
1: $head\_angle = angle(M.\mathbf{head} - M.\mathbf{up\_axis}_{u\_num})$
2: $tail\_angle = angle(M.\mathbf{tail} - M.\mathbf{low\_axis}_{l\_num})$
3: $\mathbf{v1} = [M.\mathbf{center}, M.\mathbf{up\_aixs}_1, ..., M.\mathbf{up\_aixs}_{u\_num-1}]$
4: $\mathbf{v2} = [M.\mathbf{center}, M.\mathbf{low\_aixs}_1, ..., M.\mathbf{low\_aixs}_{l\_num-1}]$
5: $\mathbf{up\_axis\_angle} = angle(M.\mathbf{up\_axis} - \mathbf{v1})$
6: $\mathbf{low\_axis\_angle} = angle(M.\mathbf{low\_axis} - \mathbf{v2})$
7: $I.C.\mathbf{center} = \mathbf{0}$
8: Sequentially construct $I.C.\mathbf{up\_axis}$ according to $L.up\_len$ and $\mathbf{up\_axis\_angle}$
9: Sequentially construct $I.C.\mathbf{low\_axis}$ according to $L.low\_len$ and $\mathbf{low\_axis\_angle}$
10: $I.C.\mathbf{head} = polarcart(L.head\_len, head\_angle) + I.C.\mathbf{up\_axis}_{u\_num}$
11: $I.C.\mathbf{tail} = polarcart(L.tail\_len, tail\_angle) + I.C.\mathbf{low\_axis}_{l\_num}$
12: Deform $I.V$ according to $I.C$
```

The function $polarcart(len, angle)$ is used to translate coordinate from polar coordinates system to Cartesian coordinates system. We employ a moving-least-squares based method [25] to deform the target model. Its closed-form solution can provide high computation performance. Meanwhile, unlike some other deformation approaches [27] [28] which require control points must be the existing vertices of the mesh, this method allows control points to be added at any positions, thus provides convenience for our sampled medial axis points (**up_axis** and **low_axis**) editing. After body motion retargeting, we can obtain an intermediate result of the target fish model (Fig. 5(c)), in which the fish body is deformed but the fins remain the same as the rest pose.

Algorithm 3 Explicit Model Deformation

```
Input: Explicit model $E$, bone length $L$ and recorded motion $M$
Output: The deformed explicit model $E$
1: Copy the positions of the body control points from $I.C$ to $E.C$.
2: Copy the positions of the deformed four junction points from $I.V$ to $E.C$.
3: $l\_fin\_angle = angle(M.\mathbf{l\_fin} - M.\mathbf{jun\_tl})$
4: $r\_fin\_angle = angle(M.\mathbf{r\_fin} - M.\mathbf{jun\_tr})$
5: $E.C.\mathbf{l\_fin} = polarcart(L.l\_fin\_len, l\_fin\_angle) + E.C.\mathbf{jun\_tl}$
6: $E.C.\mathbf{r\_fin} = polarcart(L.r\_fin\_len, r\_fin\_angle) + E.C.\mathbf{jun\_tr}$
7: Deform $E.V$ according to $E.C$
```

B. Fin Motion Retargeting

Once the implicit model I is deformed, the positions of the four junction points are determined. Then we can generate the target locations of fin tips and retarget the fin motion to the explicit model E. The details are described in Algorithm 3. The result is shown in Fig. 5(d).

C. Global Motion Retargeting

After the first two stages, we are able to retarget the local motion accurately to the fish model. In the end, we need to transform the deformed model E to world coordinates, including a rotation and a translation. The rotation angle can be directly determined by orientation $M.\mathbf{m\_dir}$, while the translation distance needs to be adjusted to meet the difference of the scales between the actual fish and the target model. Here we use a straightforward way to obtain the position of the fish model E after translation as

$$E.C.\mathbf{center} = \frac{L.body\_len}{init\_len} M.\mathbf{center}, \quad (11)$$

where $init\_len$ is the medial axis length of the actual fish which is recorded in the first tracking frame. Note that we let the user to select the first tracking frame in order to make such translation distance adjustment simpler and more intuitive.

Finally, for the smoothness of the final animation, we linearly interpolate 5 animation frames between two consecutive video frames, and use Kalman filters to estimate the control points' positions in each frame.

VI. Experimental Results

We have implemented the system using C++ and OpenCV, and all the experiments are run on an Intel(R) Core(TM) i5-6400 CPU (2.7GHz) with 8GB RAM. A video of a goldfish is captured by a monocular camera with a resolution of 1920x1088 pixels yet we resize it to 960x544 pixels to process. For 2D animation, we use a carp and a cartoon fish as the target models. For 3D animation, we use a whale as the target model. We also deliver a hand-drawn mermaid and a flower picture to the system, to test the motion retargeting effect of our technique for other fish-like characters. Fig. 6 shows the results. The parameter setting and performance statistics are listed in Table II. The memory consumption is in the range of 130MB to 200MB proportional to the number of vertices.

The current implementation is not time optimized and the entire pipeline can operate at around 20 FPS. We also compare our technique with others' work. The only research we have found in data-driven fish animation is presented by Lee *et al.* in [13]. However, they haven't shown any convincing figures or tables. Other similar work such as [29] and [30] concentrates on detection and trajectory tracking which is similar to the global motion estimation task under our framework (their methods can handle multiple fishes yet). Since we hardly find other valid researches for motion capture for fish, we compare our method with several video-based mo-cap systems for human [8] [9] [10]. The comparison result is shown in Table. III which is made with results published by peers. Given the image size and camera number, our time cost is competitive among video-based mo-cap systems.

VII. Conclusion and Discussion

In this paper we have presented a novel framework of fish motion capturing and retargeting. The fish motion can be captured by a monocular camera without resorting to any marker. We employ elliptical Fourier coefficients to analyze the swimming patterns of a fish. The fish motion was represented in a robust way, with smooth medial axis serving as the descriptor for the soft fish body. We also proposed a two-level motion retargeting scheme to properly transfer the captured motion into fish-like models. The system can then drive a static pictured fish to swim with the same motion style of the actual fish in real time. Besides, 3D models and fish-like characters can also be animated vividly in this framework.

Nevertheless, our technique still suffers from several limitations. First, our technique can only capture the fish motion from the top view, thus is unable to acquire the fish's visual information from the side view. Second, the simplified fish model we employ limits the generalization of our system. For example, we cannot model the complex fin motion of a lionfish, or the body motion of a porcupine fish. Finally, the fish needs to move properly in a small aquarium to be tracked.

Our grand motivation in this paper is to showcase a cheap yet effective way for the general public to create data-driven fish animation with a low-end mobile device. In the near future, we plan to migrate this system to a cellphone platform, so that any cellphone user could capture a swimming fish in a pool or aquarium and generate a fish character animation in an augmented-reality (AR) environment. The proposed method for medial axis extraction can be extended to describe the shape of flexible objects. Its high efficiency makes it competent in real-time applications. The two-level motion retargeting scheme can be considered as a novel combination of point-based deformation and forward kinematics, and can be easily generalized to a multi-scale scheme. Such combination greatly enlarges the range of retargeting models—making the motion retargeting process much more easily accessible between models that have similar structures at the corresponding scales.

Acknowledgment

We really appreciate Pu Ge for her cartoon design. This research is supported by National Natural Science Foundation of China (No. 61402025, 61672149, 61532002).

References

[1] J. C. Barca and G. Rumantir, "A modified k-means algorithm for noise reduction in optical motion capture data," in *International Conference on Computer and Information Science*, 2007, pp. 118–122.

[2] S. Corazza, L. Mndermann, A. M. Chaudhari, T. Demattio, C. Cobelli, and T. P. Andriacchi, "A markerless motion capture system to study musculoskeletal biomechanics: Visual hull and simulated annealing approach," *Annals of Biomedical Engineering*, vol. 34, no. 6, pp. 1019–1029, 2006.

[3] M. Oshita, "Motion-capture-based avatar control framework in third-person view virtual environments," in *ACM Sigchi International Conference on Advances in Computer Entertainment Technology*, 2006, p. 2.

[4] F. P. Kuhl and C. R. Giardina, "Elliptic fourier features of a closed contour," *Computer Graphics & Image Processing*, vol. 18, no. 3, pp. 236–258, 1982.

[5] J. Gall, C. Stoll, E. de Aguiar, C. Theobalt, B. Rosenhahn, and H. P. Seidel, "Motion capture using joint skeleton tracking and surface estimation," in *Computer Vision and Pattern Recognition*. IEEE, June 2009, pp. 1746–1753.

[6] M. Vondrak, L. Sigal, J. Hodgins, and O. Jenkins, "Video-based 3D motion capture through biped control," *ACM Transactions on Graphics*, vol. 31, no. 4, July 2012.

[7] R. Urtasun, D. Fleet, and P. Fua, "Gaussian process dynamical models for 3D people tracking," in *Computer Vision and Pattern Recognition*, 2006, pp. 238–245.

[8] C. Theobalt, M. A. Magnor, P. Schüler, and H.-P. Seidel, "Combining 2D feature tracking and volume reconstruction for online video-based human motion capture," *International Journal of Image and Graphics*, vol. 4, no. 04, pp. 563–583, 2004.

[9] B. Michoud, E. Guillou, and S. Bouakaz, "Real-time and markerless 3D human motion capture using multiple views." in *Human motion: understanding, modeling, capture and animation*, 2007, pp. 88–103.

[10] A. Shafaei and J. J. Little, "Real-time human motion capture with multiple depth cameras," in *Computer and Robot Vision*, 2016, pp. 24–31.

[11] E. Ju, J. Won, J. Lee, B. Choi, J. Noh, and M. G. Choi, "Data-driven control of flapping flight," *ACM Transactions on Graphics*, vol. 32, no. 5, p. 151, 2013.

[12] J. Wilhelms and A. V. Gelder, "Interactive video-based motion capture for character animation," in *IASTED Computer Graphics and Imaging Conference*, 2002.

[13] C.-N. Lee, W.-C. Hsieh, D.-J. Zhang-Jian, and Y. Yang, "3D fish animation with visual learning ability," in *Asia-Pacific Signal and Information Processing Association (APSIPA)*. IEEE, Dec 2014, pp. 1–4.

[14] Q. Yu and D. Terzopoulos, "Synthetic motion capture: Implementing an interactive virtual marine world," *The Visual Computer*, vol. 15, no. 7, pp. 377–394, 1999.

[15] J. Shi and C. Tomasi, "Good features to track," in *Computer Vision and Pattern Recognition*, June 1994, pp. 593–600.

[16] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Computer Vision and Pattern Recognition*, vol. 2, 2000, pp. 142–149 vol.2.

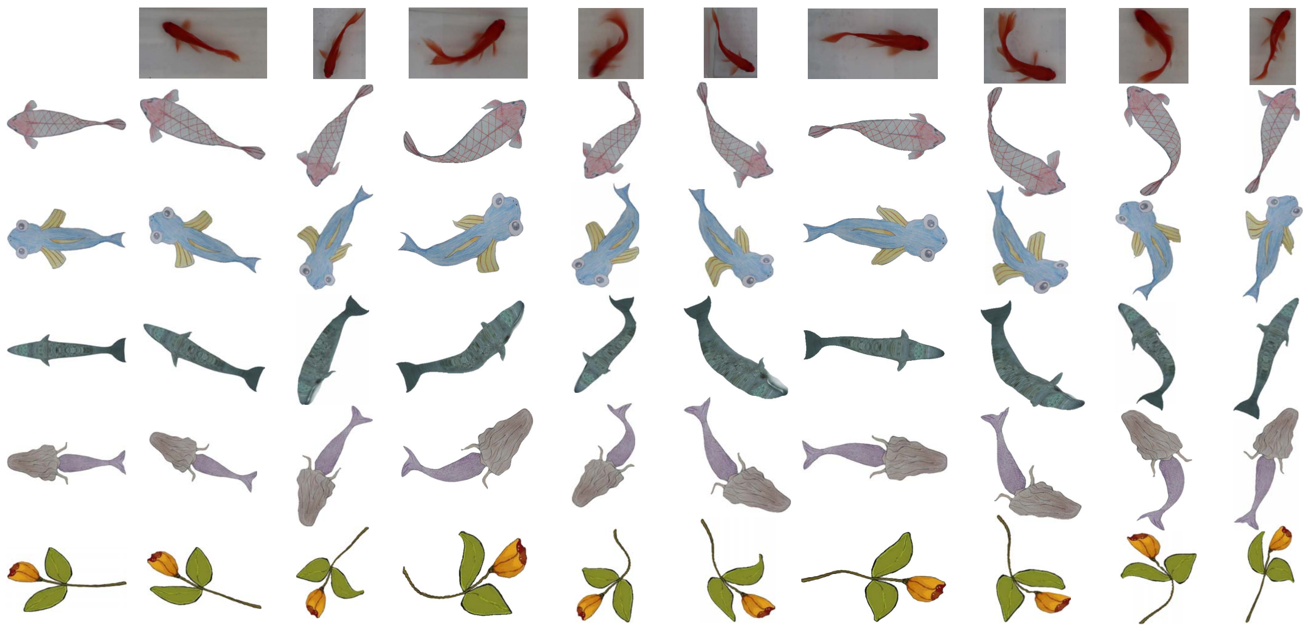

Fig. 6. Sample frames in our experiments. The top row contains nine clipped frames from the video. The remaining five rows are the corresponding animation frames of the target models. The target models from top to bottom are carp, cartoon fish, 3D whale, mermaid and flower. The first column on the left shows the rest poses of the target models.

TABLE II
THE EXPERIMENTAL PERFORMANCE STATISTICS.

Target Model	# Vertices	# Triangles	u_num	l_num	S	Tracking			Motion Retargeting	Rendering	Total Time Cost
						Background Subtraction	Filter	Feature Extraction			
Carp	2445	3827	3	15	5	16.51ms	10.03ms	8.10ms	1.13ms	12.92ms	48.69ms
Cartoon Fish	2667	4136	3	15	5				1.25ms	12.06ms	47.95ms
3D Whale	6003	9200	3	15	5				8.16ms	12.71ms	55.51ms
Mermaid	3033	4710	3	15	5				1.48ms	12.38ms	48.50ms
Flower	3492	5431	3	25	8				2.38ms	12.07ms	49.09ms

TABLE III
TIME COSTS FOR ONE-FRAME PROCESSING.

Method	[8]	[9]	[10]	Ours
Camera Number	4	3	4	1
Image Size	320x240	320x240	250x250	960x544
Average Time	236.5ms	33ms	33ms	49.94ms

[17] M. Gleicher, "Retargetting motion to new characters," in *Computer Graphics and Interactive Techniques*. ACM, 1998, pp. 33–42.

[18] H. J. Shin, J. Lee, S. Y. Shin, and M. Gleicher, "Computer puppetry: An importance-based approach," *ACM Transactions on Graphics*, vol. 20, no. 2, pp. 67–94, April 2001.

[19] K. R, M. F, and B. Arnaldi, "Morphology-independent representation of motions for interactive human-like animation," *Computer Graphics Forum*, vol. 24, no. 3, pp. 343–351, 2005.

[20] Popović, Zoran, and A. Witkin, "Physically based motion transformation," in *Computer Graphics and Interactive Techniques*. ACM, 1999, pp. 11–20.

[21] C. Bregler, L. Loeb, E. Chuang, and H. Deshpande, "Turning to the masters: motion capturing cartoons," *ACM Transactions on Graphics*, vol. 21, no. 3, pp. 399–407, July 2002.

[22] A. Hornung, E. Dekkers, and L. Kobbelt, "Character animation from 2D pictures and 3D motion data," *ACM Transactions on Graphics*, vol. 26, no. 1, p. 1, 2007.

[23] O. Barnich and D. M. Van, "Vibe: a universal background subtraction algorithm for video sequences," *IEEE Transactions on Image processing*, vol. 20, no. 6, pp. 1709–1724, June 2011.

[24] J. R. Shewchuk, "Delaunay refinement algorithms for triangular mesh generation," *Computational Geometry*, vol. 22, no. 1-3, pp. 21–74, 2002.

[25] S. Schaefer, T. Mcphail, and J. Warren, "Image deformation using moving least squares," *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 533–540, July 2006.

[26] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[27] T. Igarashi, T. Moscovich, and J. F. Hughes, "As-rigid-as-possible shape manipulation," *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 1134–1141, July 2005.

[28] Y. Weng, W. Xu, Y. Wu, K. Zhou, and B. Guo, "2D shape deformation using nonlinear least squares optimization," *The Visual Computer*, vol. 22, no. 9, pp. 653–660, 2006.

[29] M. C. Chuang, J. N. Hwang, J. H. Ye, S. C. Huang, and K. Williams, "Underwater fish tracking for moving cameras based on deformable multiple kernels," *IEEE Transactions on Systems Man & Cybernetics Systems*, vol. PP, no. 99, pp. 1–11, 2016.

[30] M. C. Chuang, J. N. Hwang, K. Williams, and R. Towler, "Tracking live fish from low-contrast and low-frame-rate stereo videos," *IEEE Transactions on Circuits & Systems for Video Technology*, vol. 25, no. 1, pp. 167–179, Jan 2015.