SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

September 2016, Vol. 59 092208:1–092208:15
doi: 10.1007/s11432-015-5497-1

Precise planar motion measurement of a swimming multi-joint robotic fish

Jun YUAN, Junzhi YU*, Zhengxing WU & Min TAN

State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190*, China*

Received May 17, 2015; accepted September 6, 2015; published online August 23, 2016

Abstract This paper presents a method for planar motion measurement of a swimming multi-joint robotic fish. The motion of the robotic fish is captured via image sequences and a proposed tracking scheme is employed to continuously detect and track the robotic fish. The tracking scheme initially acquires a rough scope of the robotic fish and thereafter precisely locates it. Historical motion information is utilized to determine the rough scope, which can speed up the tracking process and avoid possible ambient interference. A combination of adaptive bilateral filtering and k-means clustering is then applied to segment out color markers accurately. The pose of the robotic fish is calculated in accordance with the centers of these markers. Further, we address the problem of time synchronization between the on-board motion control system of the robotic fish and the motion measurement system. To the best of our knowledge, this problem has not been tackled in previous research on robotic fish. With information about both the multi-link structure and motion law of the robotic fish, we convert the problem to a nonlinear optimization problem, which we then solve using the particle swarm optimization (PSO) algorithm. Further, smoothing splines are adopted to fit curves of poses versus time, in order to obtain a continuous motion state and alleviate the impact of noise. Velocity is acquired via a temporal derivative operation. The results of experiments conducted verify the efficacy of the proposed method.

Keywords motion measurement, robotic fish, time synchronization, visual tracking, localization

Citation Yuan J, Yu J Z, Wu Z X, et al. Precise planar motion measurement of a swimming multi-joint robotic fish. Sci China Inf Sci, 2016, 59(9): 092208, doi: 10.1007/s11432-015-5497-1

1 Introduction

The past decade has witnessed increasing interest and a plethora of research studies on robotic fish. Of these, one critical research topic is motion measurement of the robotic fish, including acquisition of pose, linear (angular) velocity, and acceleration. This is because motion information is indispensable for motion performance evaluation [1], model parameter identification [2], and real-time motion control feedback [3]. In general, there are two kinds of sensors that can be used for motion measurement of robotic fish in a laboratory environment: image sensor and inertial measurement unit (IMU). However, civilian IMUs can not offer precise and reliable estimation of position and linear velocity, owing to temperature drift and accumulated error. A good alternative is to measure motion using cameras, as they can obtain more reliable, accurate, and abundant motion information. As a result, computer vision is widely used

* Corresponding author (email: junzhi.yu@ia.ac.cn)

as means for motion measurement of robotic fish. The contents of this paper are focused on computer vision-based methods. According to the positions in which the cameras are installed, the adopted vision systems can be divided into two categories: on-board vision systems and global vision systems.

As the term suggests, on-board vision systems [4, 5], refer to cases in which the robotic fish carry cameras on themselves. Such cameras are just like the eyes of the robotic fish and move together with them. Robotic fish can extract their motion information from the visual features captured by their eyes. Compared with global vision systems, the application of on-board vision systems is not constrained to indoor environments. On-board vision systems may also operate in natural waters. However, motion measurement by on-board vision systems is very challenging, because of lack of features, weak illumination and poor image quality. At present, the accuracy of on-board systems is lower than that of global vision systems.

Global vision systems are suitable for pose measurement in indoor experimental environments. In this kind of systems, cameras are installed above or around a water tank with their fields of view covering the range of motion of the robotic fish. Global vision systems can use either of two different approaches to obtain motion information. One kind of approach [6, 7] extracts and tracks the whole posture or silhouette of the robotic fish in each image. This kind of approach is able to obtain the full-body motion information of the robotic fish. However, as a sword has double edges, this kind of approach is time-consuming and therefore is not suitable for real-time applications. The other kind of approach only tracks specific regions on the robotic fish, rather than considering the entire body. So this kind of approach is much simpler, faster, and is consequently more commonly used. Various markers have been established to designate the regions, such as light-emitting diodes (LEDs) [8], punctate markers [2,9] attached along the midline of the robotic fish, and rectangular patches with specific colors [10]. Algorithms have also been developed to detect the regions, with most of them based on background subtraction [11,12] and threshold segmentation [8, 10, 11]. Tracking algorithms, e.g., Kalman filters [12], have also been applied to motion state estimation. It has been observed that fluctuations and reflections on the water surface caused by the motion of a robotic fish would lead to numerous false alarms after the background subtraction operation. With respect to threshold segmentation methods, parameter tuning is tedious, and image segmentation results are readily influenced by illumination changes. Yu et al. [10] proposed an adaptive thresholding method to alleviate the effect of ambient light by analyzing the color statistics of the background. However, appearance variations of markers are not only from ambient illumination changes, but also from attitude changes and immersion by water. Consequently, realization of accurate segmentation of markers through the above threshold-based methods is difficult.

In addition to pose measurement, the time information in a motion measurement system should also be accurate in order to obtain precise motion measurement via global vision. The time of the motion measurement system should definitely be synchronized with that of the on-board control system of the robotic fish. Although nonconformity with this condition has only a minor impact on the calculation of velocity and acceleration, it can lead to inaccuracy in the time information corresponding to a motion state. A minor inaccuracy in time information can cause significant differences in joint states or controlled quantities. Further, it can affect some motion information applications, e.g., dynamic simulation [8] and identification of hydrodynamic parameters [2]. Thus, time synchronization is another crucial technique in the measurement system. Distributed modules in sensory systems usually demand a common synchronized time reference for data integration [13], localization [14], and coordinated actuation [15], etc. In general, time synchronization is solved by dedicated hardware [16], network synchronization protocols [17], and timestamp filters [18]. For instance, Nakadai et al. [16] installed a potentiometer in their sound-tracking wheeled robot to synchronize the sound signal and angle information captured from a sound card and an encoder respectively. To implement a robot musical accompaniment, Lim et al. [17] synchronized audio and visual cues via network time protocol. Nevertheless, owing to the additional hardware and software synchronization capabilities needed in sensory systems, approaches based on hardware or communication are often undesirable, especially in low-cost flexible solutions [18]. Furthermore, filtering methods are often conducted under certain assumptions of noise probability distribution that may be impractical.

This paper proposes a method for precise planar motion measurement of a multi-link robotic fish. We summarize the contributions of this paper from the following three aspects. First, a simple and practicable tracking scheme is proposed for continuous robotic fish detection and tracking. The main idea underlying the tracking scheme is initial acquisition of a rough scope of the robotic fish followed by precise localization thereafter. Historical motion information is utilized to determine the rough scope. This can speed up the tracking process and avoid some ambient interference. Then, a combination of adaptive bilateral filtering and k-means clustering is applied to accurately segment out color markers. The pose of the robotic fish is calculated according to the centers of these markers. Second, an algorithm that synchronizes the time of the motion measurement system with the on-board control system, which is different from any traditional time synchronization methods, is presented. Exploiting both the multi-link structure of the robotic fish and the motion law of joints, the problem is converted into a nonlinear optimization problem with two unknown variables, and solved using the particle swarm optimization (PSO) algorithm. Third, smoothing splines are adopted to fit curves of poses versus time in order to obtain a continuous motion state and reduce the impact of noise. Velocity is further acquired by a derivative operation. The results of experiments conducted verify the efficacy of the proposed method.

The remainder of this paper is organized as follows. Section 2 gives basic information about the multi-link robotic fish and the motion measurement system. Section 3 elaborates on the tracking scheme. Section 4 outlines the time synchronization problem and our corresponding solution. Section 5 presents the proposed velocity estimation method. In Section 6, the experiments conducted on the robotic fish prototype are described. Finally, Section 7 concludes this paper.

2 The robotic fish and the motion measurement system

The motion measurement method is oriented to a multi-link robotic fish built in our laboratory [19]. The mechanical structure of the robotic fish is depicted in Figure 1(a). In brief, the robotic fish is composed of a rigid head and a self-propulsive body. The body is designed as a multi-link hinge structure, which is a popular style for robotic fish. This multi-link structure comprises four aluminum skeletons connected in series along the body. In addition, the body ends with a rigid caudal fin affixed to the last link. This configuration corresponds to four oscillating joints, which are actuated by servomotors with strong torque and high speed. More specifically, each servomotor has a maximum output torque of 1.6 N·m and maximum angular speed of 1000°/s. The servomotors are controlled by a controller embedded in the head of the robotic fish. This controller is equipped with an radio frequency (RF) transceiver to receive commands or send data. Waterproof markers are attached to the skin out of the rigid shell, as shown in Figure 1(b), which help to distinguish the robotic fish from the environment.

To measure the planar motion of the robotic fish, as illustrated in Figure 2, a measurement system was constructed. The robotic fish swims in a pool filled with water. A video camera is mounted above the pool and connected to a host computer. The camera has a resolution of 1292×964 and a maximum frame rate of 30 fps (frames per second). In order to avoid strong reflection from the water surface, lighting sources are shielded from the front cover. The side effect of this is that the illumination is weakened and so the camera requires longer exposure time. In actuality, the camera operates at a frame rate around 22 fps for adequate exposure time during measurements. The camera sends captured image sequences to the host computer via a USB interface. The computer stores these image data for off-line analysis, or processes them on-line. In addition, the computer communicates with the robotic fish via an RF transceiver, which operates at the same frequency. Thereby, commands or data are wirelessly exchanged between the robotic fish and the host computer.

3 Pose measurement via global vision

Attached markers designate the pose of the robotic fish. It is necessary to find and segment out the markers in each image as completely as possible for high accuracy. To reduce computation cost, searching in an entire image every time should be avoided. Historical motion data during swimming helps to narrow

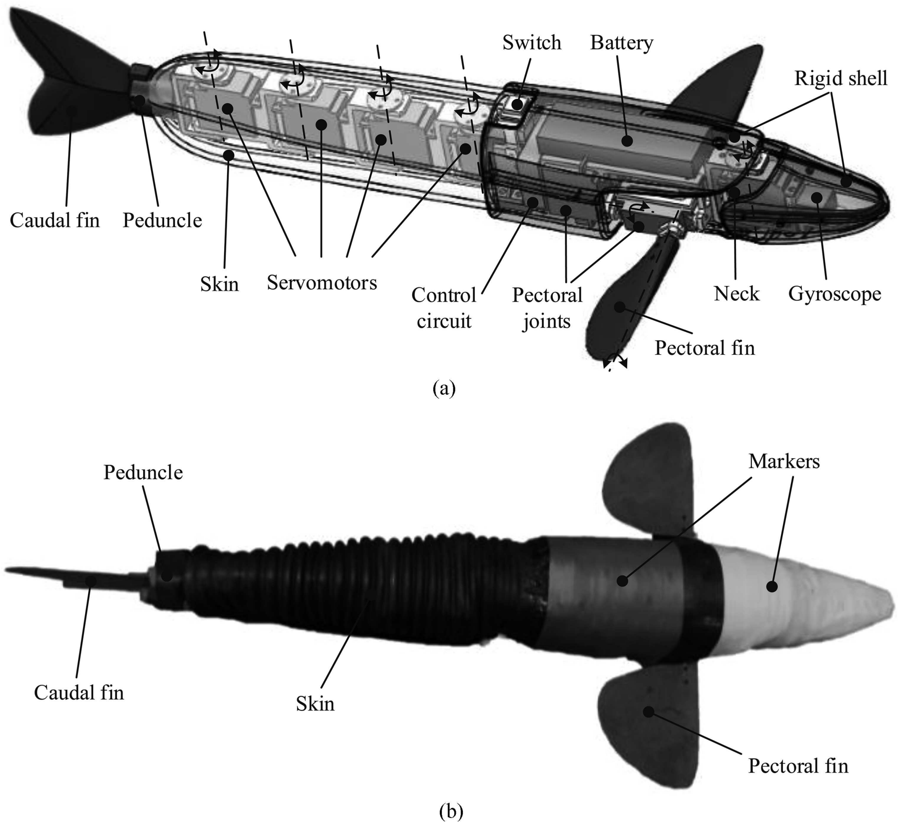

Figure 1 Multi-link robotic fish. (a) Internal structure; (b) prototype with markers.

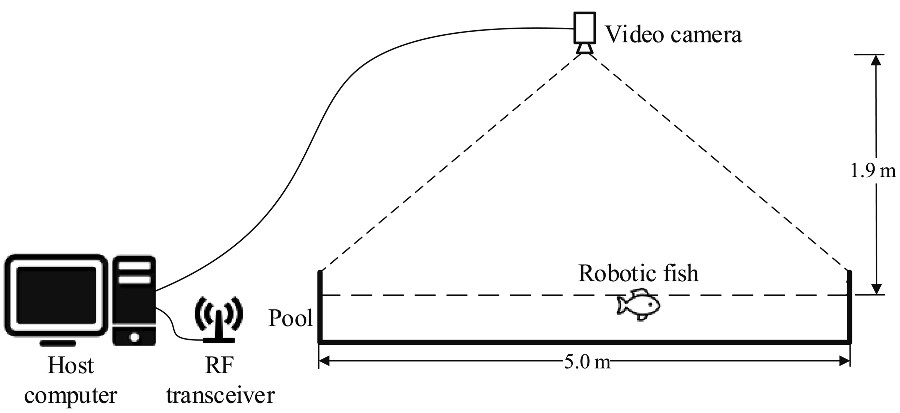

Figure 2 Schematic of the motion measurement system.

the search scope. Thus, as illustrated in Figure 3, a tracking scheme is adopted to detect and track the color markers continuously in captured image sequences. The main idea of the tracking scheme is to determine an approximate scope in which the robotic fish can first be located and then obtain an exact location. The motion information in a previous frame is applied to predict a possible region in the next frame. Further, the condition that the robotic fish may disappear from the "field of view" during tracking is considered in the tracking scheme, which is handled through a rediscovery mechanism. In essence, the tracking scheme can be regarded as a universal tracking framework. The concrete process of each part of the tracking scheme is detailed below:

(1) Search globally. In this step, the markers are searched in the entire scope of an image. Under normal circumstances, it is applied to initialize a region of interest (ROI) at the very first frame because

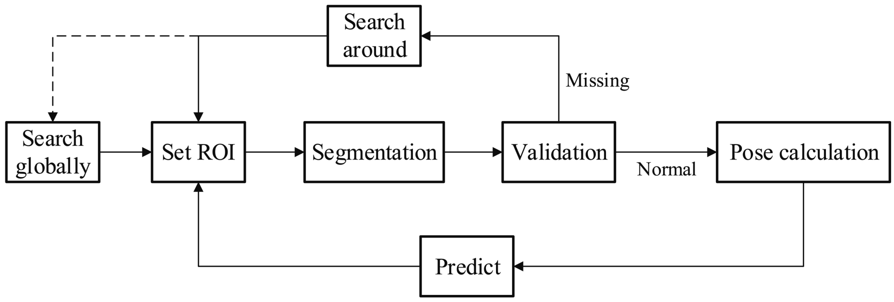

Figure 3 Diagram of the tracking scheme.

no historical motion data exists. Further, this step can also be invoked when the "search around" fails to find the markers. The searching process is mainly based on appearance discrepancies between the markers and the background. The rectangular marker has the characteristic that its pixel values are obviously higher than the background in the R channel of a normalized RGB image. In like fashion, it is also observed that the Cb components of the triangular markers are lower than the background in the YCrCb space. To reduce the impact of illumination, we propose a color histogram based adaptive threshold (CHAT) method specific to the markers. The following method is for the case of the rectangular marker. An RGB image is first converted to a normalized RGB image. Then, a cumulative histogram is built for the R channel of the normalized RGB image. Assume that the R components of the markers are higher than the background. Given a percentage of the pixel number of the markers to that of the whole image, a threshold value can be calculated according to the cumulative histogram built. The region of the markers can be obtained through a thresholding process using the threshold value. Once settled, the percentage value functions well although illumination changes in a normal range. It is not necessary to tune the percentage value repeatedly to split off the whole marker, because only a rough position of the robotic fish is required in this step. Area size information is also applied to filter out underlying false alarms. The coordinate of the marker's center, denoted by $(\bar{x},\ \bar{y})$, can be calculated as follows:

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

where n is the number of pixels in the region acquired, and $(x_i,\ y_i)$ is the coordinate of the ith pixel in the region. Finally, this step outputs $(\bar{x},\ \bar{y})$. The strategy above can also be applied to the other marker. The search process is more reliable when both markers are detected at the same time.

(2) Set ROI. Given a center $(\bar{x},\ \bar{y})$, an ROI is set as a rectangular region of an image:

$$\left\{(x,y)\middle| x \in \left[\max\left(\bar{x}-\frac{w}{2},0\right),\min\left(\bar{x}+\frac{w}{2},W\right)\right],\ y \in \left[\max\left(\bar{y}-\frac{h}{2},0\right),\min\left(\bar{y}+\frac{h}{2},H\right)\right]\right\}, \tag{2}$$

where W and H are the width and height of the image respectively, and w and h are the width and height of an ROI.

(3) Segmentation. Whether the markers can be segmented out accurately in this step determines the ultimate localization precision. As shown in Figure 4, two stages constitutes the segmentation process: adaptive bilateral filtering [20] and k-means clustering [21]. The segmentation process is implemented on each ROI. It is not conducive to subsequent processing that initial images are noisy. Thus, after an ROI is set up, an adaptive bilateral filter is adopted to smooth the ROI while preserving sharp edges. In essence, the image filtering process implements a weighted average operation to each pixel by its neighborhood pixels. Crucially, the weights here are determined by not only the Euclidean distance of pixels but also the discrepancies of color intensity. Figure 4 shows that the filtering process improves the quality of an ROI dramatically. Nevertheless, the markers still show considerable intra-regional diversity, owing to illumination angle differences and immersion by water. It is difficult for a thresholding process to segment out the markers accurately and completely. Therefore, k-means clustering, a simple unsupervised method,

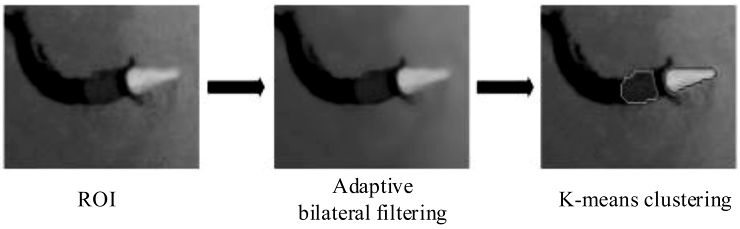

Figure 4 The segmentation process.

is adopted. Pixels constituting the markers are collected according to the similarity between each other rather than an absolute threshold. Each pixel p in the ROI is described by a feature vector composed of its RGB values, i.e., $p = [R\ G\ B]^{\mathrm{T}}$. All the pixels are partitioned to k sets, i.e., $Q = \{S_1,\ S_2, \ldots, S_k\}$, to minimize the within-cluster sum of squares:

$$\arg\min_{Q} \sum_{i=1}^{k} \sum_{p\in S_i} (p - u_i)^{\mathrm{T}} \cdot (p - u_i), \tag{3}$$

where u_i is the mean of the feature vectors in S_i. Compared with general k-means methods, the initialization manner of mean values is different here. Historical data is utilized to reduce iteration times: the mean vector at frame j is initialized by the corresponding one at frame $j-1$ if $j > 1$, i.e., $u_{i,j} = u_{i,j-1}$. Once clustering is accomplished, regions of the markers are among the k sets. In Figure 4, the circled regions are pixel sets corresponding to the two markers.

(4) Validation. On completion of the segmentation process, pixel sets corresponding to the markers are selected and checked in this step. The mean RGB values of the two markers in the latest m frames are recorded, i.e., $U_{i,j-1} = \{u_{i,j-m}, u_{i,j-m+1}, \ldots, u_{i,j-1}\}$. Pixel set S_i can be regarded as a candidate region if

$$\left|u_{i,j} - \bar{U}_{r,j-1}\right| \leqslant \lambda \cdot \sigma_{r,j-1} \text{ and } |a_{i,j} - A_r| \leqslant \varepsilon_r, \tag{4}$$

where $U_{r,j-1}$ is the record of the red marker; $\bar{U}_{r,j-1}$ and $\sigma_{r,j-1}$ are the mean vector and the vector of standard deviation, i.e., $\bar{U}_{r,j-1} = \frac{1}{m}\sum_{q=1}^{m} u_{r,t-q}$, $\sigma_{r,j-1} = \mathrm{std}(U_{r,j-1})$; λ is a coefficient; $a_{i,j}$ is the area of S_i; A_r is the real area of the red marker; and ε_r is the allowed area deviation of the red maker. The criteria described in (4) are also applied to find and check the other marker. If both the markers are available, this module outputs "normal" and goes to "pose calculation" next. Otherwise, the robotic fish is deemed to be "missing" and there is a need to search around.

(5) Search around. The scheme in this step is the same as that in "search globally" except for the search scope. The scope is also set as a rectangular region, whose width and height are doubled relative to an ROI. If both markers are found, a new ROI is set. Otherwise, "search globally" is activated.

(6) Pose calculation. The position of the head apex is utilized to represent the location of the robotic fish. The pose at frame j can be calculated as follows:

$$\alpha_j = \arctan((y_{r,j} - y_{e,j})/(x_{r,j} - x_{e,j}))(x_{a,j}, y_{a,j}) = (x_{r,j} + l \cdot \cos\alpha_j, y_{r,j} + l \cdot \sin\alpha_j), \tag{5}$$

where $(x_{a,j}, y_{a,j})$ is the coordinate of the head apex, α_j denotes the yaw angle, l is the distance from the center of the rectangular marker to the head apex, and $(x_{r,j}, y_{r,j})$ and $(x_{e,j}, y_{e,j})$ are the centers of the two markers, respectively. The centers can be calculated by (1) when regions of the markers are obtained.

(7) Predict. The rough location of the robotic fish in the next frame can be predicted based on historical motion data. A constant velocity motion model is adopted to describe the motion of the robotic fish, although it is inconsistent with the reality. It is easy to calculate the moving distance between two consecutive frames:

$$\Delta_j = (x_{c,j} - x_{c,j-1}, y_{c,j} - y_{c,j-1}), \tag{6}$$

where $(x_{c,j}, y_{c,j}) = \frac{1}{2}(x_{r,j} + x_{e,j}, y_{r,j} + y_{e,j})$. According to the assumption of constant velocity motion, we can predict the location of the robotic fish at frame $j+1$ as follows:

$$(\hat{x}_{c,j+1}, \hat{y}_{c,j+1}) = (x_{c,j}, y_{c,j}) + \Delta_j = (2x_{c,j} - x_{c,j-1}, 2y_{c,j} - y_{c,j-1}), \tag{7}$$

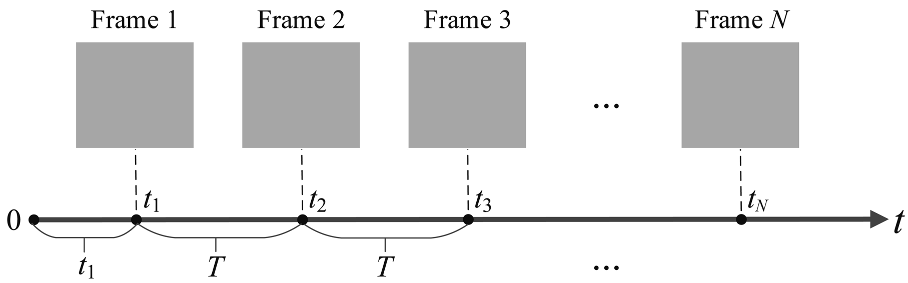

Figure 5 Illustration of timeline.

where $(\hat{x}_{c,j+1}, \hat{y}_{c,j+1})$ is the predicted position. Specifically, a prediction may be inaccurate, but it can be used to set the ROI in the next frame because the robotic fish can not move a long distance during a short time slice.

The tracking process outputs the pose of the robotic fish in each frame. More precisely, poses above are in the image coordinate system. Through camera calibration, we can obtain corresponding poses in the world coordinate system.

4 Time synchronization

A single image only reserves the static state of the robotic fish at a point in time. Connecting static states via a timeline creates the motion information. It is necessary to guarantee an accurate timebase for a precise motion measurement system. Because the motion measurement system and the on-board control system have independent time systems, the exact time point when the robotic fish executes a start command is not knowable by the motion measurement system. Thus, a problem of time synchronization inevitably arises. This problem has been ignored in previous studies on robotic fish. In this section, this problem is first analyzed. Then, a method that exploits the multi-link structure of the robotic fish and the motion law of its joints is proposed to solve the problem.

4.1 Problem analysis

The essence of time synchronization is determination of a precise corresponding time point when each state of the robotic fish appears. The motion measurement system captures frames one at a time, each of which records the static state at a time point on a timeline. As illustrated in Figure 5, the origin of the timeline is the time point where the robotic fish just begins to swim. Frame 1 denotes the first frame in which motion appears, and time intervals between two consecutive frames are equal. Therefore, we require the two key time parameters, t_1 and T, demonstrated in Figure 5, to obtain an accurate time point for each frame.

Although it is the motion measurement system that sends the start command, the exact time point at which the robotic fish starts is unclear to the system. This is because it is the on-board control system that parses and executes the start command. The start time point can be acquired easily and accurately in the on-board control system, because the system is dedicated to real-time control tasks. Nevertheless, the time systems of these two systems are independent and share no common reference. Consequently, even though an accurate timestamp of the first frame can be obtained, we are uncertain about the time length from the start time point to the first frame, i.e., t_1.

The time interval between two consecutive frames, T, is another important time parameter. The precise value of the parameter is also mysterious. There are two main reasons for this. Firstly, the timestamp attached to each frame is not sufficiently accurate to be used, owing to the non-realtime and non-deterministic scheduling specialties of the operating system (Microsoft Windows) installed on the host computer. An instance of time intervals is given, which are calculated by the timestamps acquired from the host computer, as shown in Figure 6. Specifically, the Windows API called query performance counter is utilized to acquire timestamps. Figure 6 shows that the timestamps attached to the frames are

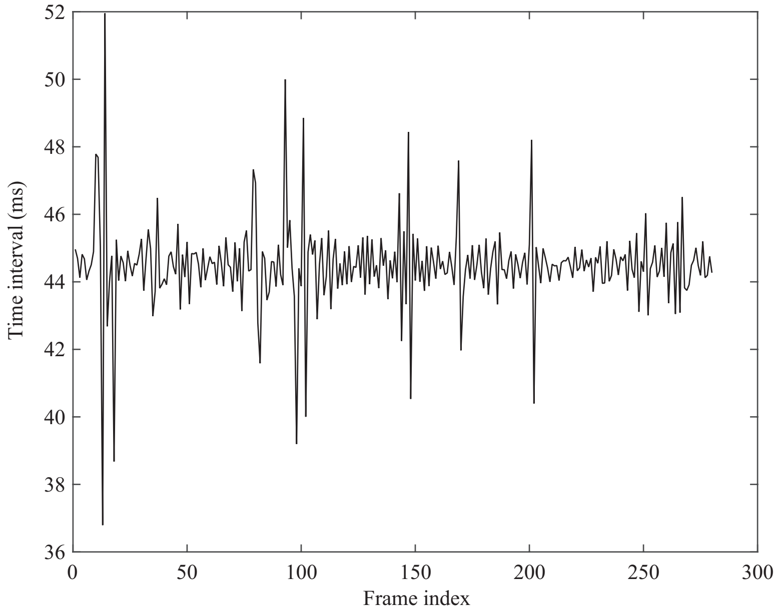

Figure 6 Time intervals calculated using timestamps from the host computer.

not consistent with the periodic characteristic of the sampling process of the camera. Secondly, the frame rate of the camera is related to the time of exposure and readout. Suppose that the readout time keeps constant for relevant parameters are fixed. Once the exposure time is fixed, the frame rate will remain constant. When the exposure time is adjusted to meet actual demand, the frame rate will change. A rough value of frame rate is clear, whereas it is difficult to acquire the precise frame rate of the camera. Estimating T using a rough frame rate will lead to deviation of T, denoted by ΔT. As a result, time errors in each frame will accumulate over time. For frame N, the error of its time point will reach $N \cdot \Delta T$. Note that the control period of the on-board control system is 40 ms. If $\Delta T = 1$ ms, the time errors will surpass a control period after 40 frames. The increasing time errors will cause significant deviation in the joint states. An instance demonstrates the angular deviation of the four joints and distance deviation of the caudal fin's endpoint when $\Delta T = 1$ ms, as shown in Figure 7. The instance indicates that even a millisecond error in T can result in a significant deviation. The cumulative time errors lead to significant mismatch between joint states and motion information, which will affect applications such as model parameter identification. Therefore, T should be as accurate as possible.

4.2 Solution

The real-time capability of the on-board control system makes for a promising solution to the problem of time synchronization. Owing to the high-speed characteristic and the innate scheme of closed-loop control, the servomotors are able to reach a target angle within a control period. The joints rotate according to a given rule, which provides the target joint angles at different time points. Thus, the joint states connote corresponding time information. The problem can be solved by matching the joint states captured by images with the given motion rule.

Specifically, the motion of the joints conforms to the Hopf oscillator-based central pattern generator (CPG) model [22] below:

$$\begin{cases} \dot{\xi}_i = -\omega_i(\psi_i - b_i) + \xi_i(A_i - {\xi_i}^2 - (\psi_i - b_i)^2) + h_1(\xi_{i-1}\cos\varphi + (\psi_{i-1} - b_{i-1})\sin\varphi_i), \\ \dot{\psi}_i = \omega_i\xi_i + (\psi_i - b_i)(A_i - {\xi_i}^2 - (\psi_i - b_i)^2) + h_2(\xi_{i+1}\sin\varphi + (\psi_{i+1} - b_{i+1})\cos\varphi_i), \\ \theta_i = c_i\psi_i, \end{cases} \tag{8}$$

where subscript i indicates the ith oscillator ($i = 1, 2, 3, 4$), ξ_i and ψ_i are the oscillation states, the "dot" over the head of ξ_i and ψ_i represents temporal derivative d./dt, ω_i and A_i denote the intrinsic

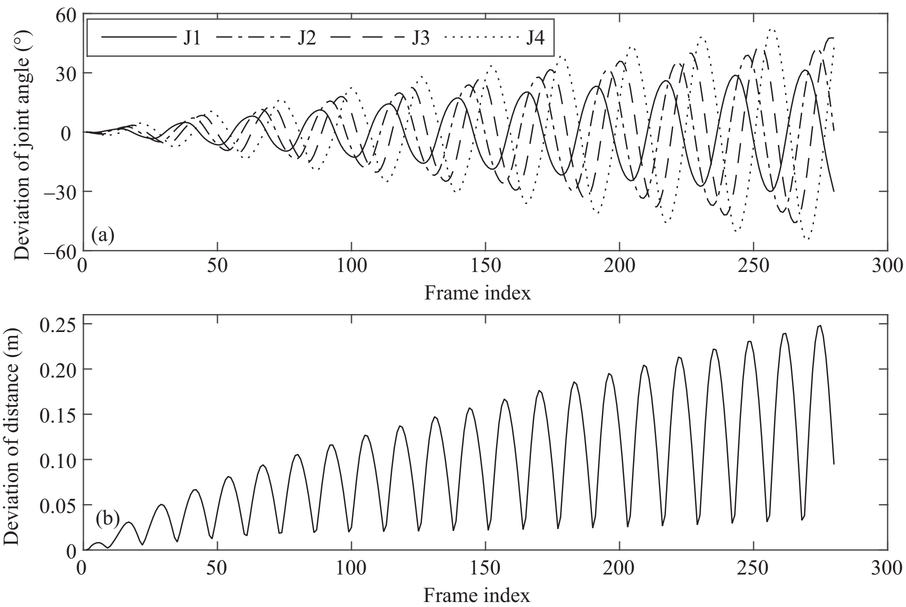

Figure 7 Deviation of joint states when $\Delta T = 1$ ms. (a) Deviation of joint angles; (b) distance deviation of the caudal fin's endpoint.

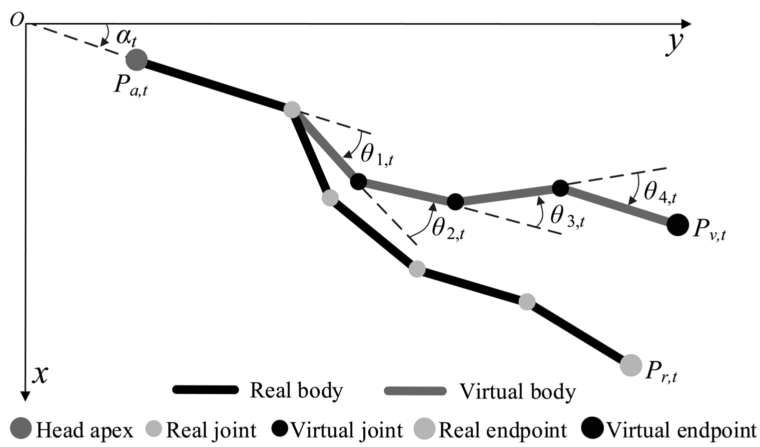

Figure 8 Matching a virtual skeleton with the real skeleton of the robotic fish.

oscillation frequency and amplitude, h_1 and h_2 are the coupling weights, φ_i is the phase difference between neighboring oscillators, b_i denotes the directional bias, c_i is the magnification coefficient, and θ_i is the output of the ith oscillator (i.e., the target angle of the ith joint). Given the parameters of the CPG model, the motion law of the joints described in (8) is a function of time t, which can be formalized concisely as follows:

$$\eta_t = f_{\mathrm{cpg}}(t|\gamma), \tag{9}$$

where γ denotes the parameters of the CPG model, and η_t is a vector composed of the four joint angles at time t, i.e., $\eta_t = [\theta_{1,t}, \theta_{2,t}, \theta_{3,t}, \theta_{4,t}]$. Consequently, joint angles corresponding to any time point can be calculated using (9).

In reality, the joints are covered with a waterproof skin. Consequently, extracting the joint angles directly in an image is difficult. Fortunately, the joint angles determine the position of the caudal fin's endpoint, which is relatively easy to locate. Therefore, we turn to matching of the caudal fin's endpoint instead of the joint angles. A virtual skeleton is estimated to fit the shape of the real fish in each image, as illustrated in Figure 8. The skeleton of the robotic fish is regarded as a multi-link structure. The virtual skeleton shares the same head with the robotic fish. The position of the head apex and the yaw angle in each frame can be obtained in the tracking process. With the head's pose settled, the position of the caudal fin's endpoint $P_{v,t}$ is determined by the joint angles η_t:

$$\begin{cases} x_{v,t} = x_{a,t} + l_0 \cos\alpha_t + \sum\limits_{i=1}^{4} l_i \cos\left(\alpha_t + \sum\limits_{j=1}^{i} \theta_{j,t}\right), \\ y_{v,t} = y_{a,t} + l_0 \sin\alpha_t + \sum\limits_{i=1}^{4} l_i \sin\left(\alpha_t + \sum\limits_{j=1}^{i} \theta_{j,t}\right), \end{cases} \tag{10}$$

where $(x_{a,t}, y_{a,t})$ is the coordinate of the head apex $P_{a,t}$, $(x_{v,t}, y_{v,t})$ is the coordinate of $P_{v,t}$, l_0 indicates the length of the head, and l_i denotes the length of the ith link. If t is the precise time point corresponding to a frame, then the endpoint of the virtual skeleton should overlap the endpoint of the real skeleton. The precise time point corresponding to Frame j can be calculated below:

$$t_j = t_1 + (j-1)T. \tag{11}$$

It is clear that parameters t_1 and T determine t_j, the joint angles η_{t_j}, furthermore the location of P_{v,t_j}, and eventually the distance between P_{v,t_j} and P_{r,t_j}. Given a set of real endpoints in several images, t_1 and T can be solved by minimizing the square summation of distance between the real and virtual endpoint pairs as follows:

$$\mathop{\arg\min}_{t_1,T} \sum_{j\in I} \left\| P_{r,t_j} - P_{v,t_j} \right\|_2^2, \tag{12}$$

where $\|\cdot\|_2$ denotes the Euclidean norm, I is a set containing indexes of frames at which real endpoints are given, and I should own at least two elements because there are two parameters to be solved for.

Consequently, the problem of time synchronization is converted to a nonlinear optimization problem, which can be formalized by a combination of (9)–(12) as follows:

$$\begin{cases} \mathop{\arg\min}\limits_{t_1,T} \sum\limits_{j\in I} \left\| P_{r,t_j} - P_{v,t_j} \right\|_2^2, \\ P_{v,t_j} = g(\eta_{t_j} | P_{a,t_j}, \alpha_{t_j}), \\ \eta_{t_j} = f_{\mathrm{cpg}}(t_j|\gamma), \\ t_j = t_1 + (j-1)T, \\ T_L \leqslant T \leqslant T_U, \\ 0 \leqslant t_1 < T, \end{cases} \tag{13}$$

where the function $g(\cdot)$ is a simplified representation of (10), and T_L and T_U are the lower and upper bounds of T, respectively. Considering the nonlinearity of the problem described in (13), we adopt the PSO algorithm [23] to solve for t_1 and T. PSO is a population-based stochastic search algorithm, in which both local and global best known data is used to adjust the action of each particle. An open source tool [24] that implements the PSO algorithm is applied to solve the problem. As stated above, a set of endpoints need to be extracted from several images. We mark the endpoints manually because detecting them automatically and directly is very difficult. Attaching a certain marker to the tailfin would make automatic detection easy and feasible. A more accurate solution will be obtained when the endpoints are extracted from images with large indexes. This is attributable to the cumulative effect of errors illustrated in Figure 7, which can only be terminated by accurate time parameters. Further, the more endpoints we supply, the more reliable the result will be, although two endpoints can also generate a solution.

5 Velocity estimation

With pose and time information ready, the velocity of the robotic fish can be estimated. First, we need to determine the continuous relationship between poses and time. Because the poses obtained through the tracking process are a discrete time series. Furthermore, it is inevitable that the pose data is contaminated

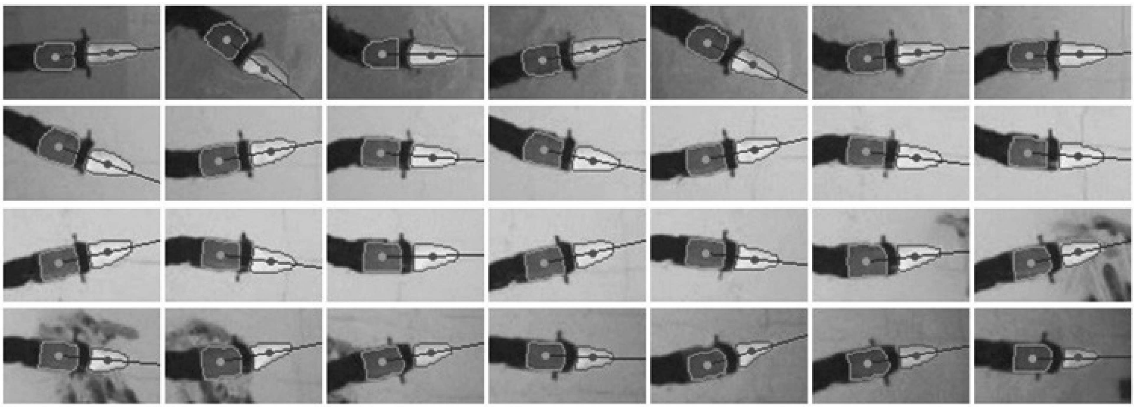

Figure 9 The segmentation results: from Frame 1 to Frame 298 with a uniform interval of 11 frames.

by noise. Nevertheless, velocity estimation should be implemented with derivative operations, which will amplify the noise. Thus, steps should be taken to relieve the impact of noise.

To satisfy these two requirements, we adopt smoothing splines [25] to fit the relationship between poses and time. The smoothing spline method is one of the most popular nonparametric regression methods. The essence of the regression process is to determine a smoothing spline $f(\cdot)$ that serves as a solution to the minimization problem below:

$$\min\left\{\rho\sum_{j=1}^{N}\left[z_j - f(t_j)\right]^2 + (1-\rho)\int_0^{t_N}\left[f''(t)\right]^2\mathrm{d}t\right\}, \tag{14}$$

where z_j denotes the pose extracted from frame j, ρ represents the smoothing parameter ($\rho \in [0,1]$), and $f(\cdot)$ is set as a cubic spline for continuous first and second derivatives. The first term in (14) appears as the residual summation of the squares, whereas the second term is a roughness penalty based on the second derivative of $f(\cdot)$. The smoothing parameter ρ adjusts the tradeoff between the goodness and smoothness of fit. After the parameters of the cubic spline are obtained, we calculate the linear (angular) velocity v using a temporal derivative operation, $v = f'(t)$.

6 Experiments and analysis

6.1 Pose measurement experiments

We captured video clips to assess the pose measurement performance. The segmentation results in a video clip are shown in Figure 9. The figure qualitatively illustrates the accuracy and robustness of the adopted segmentation method. Each image in Figure 9 is an ROI set automatically in the tracking scheme, which indicates that the prediction strategy is adequate for keeping the robotic fish in focus. The regions surrounded with enclosed contours are the segmented markers, and the points mark their centers. It can be seen that the markers can be segmented out virtually completely even though their appearance varies with ambient illumination and other factors.

For the quantitative evaluation of the accuracy of pose measurement, the position of the head apex and the yaw angle were taken as quality metric. Head apexes and yaw angles in 100 frames were marked manually as ground-truth. Further, the distances between the calculated and given head apexes were regarded as position errors, and the absolute values of the deviation of yaw angles were angular errors. The poses were expressed in the world coordinate system. Four approaches were accomplished for evaluation and comparison. Their performances are listed in Table 1. The CHAT method in the "search globally" module was used as a baseline. In addition, CHAT was utilized to locate ROIs for the meanshift [26] and the graph-cut algorithms [27]. Gaussian mixture models (GMMs) were applied to model the foreground and background in the graph-cut framework. The GMMs were trained off-line using collected samples.

Table 1 Pose measurement performance

Method	Position error (mm)	Angular error (°)	Average time cost per frame (ms)
CHAT	$(8.3, 4.1)$	$(2.0, 1.6)$	64.8
CHAT+meanshift	$(9.3, 5.2)$	$(1.9, 2.8)$	71.6
CHAT+graph cut	$(7.4, 3.5)$	$(2.2, 1.6)$	102.8
Our method	$(6.7, 3.4)$	$(1.9, 1.4)$	30.3

In Table 1, the position and angular errors are represented in a format $(\mu,\ \sigma)$, where μ and σ denote mean value and standard deviation, respectively. It is clear that our method achieves the best precision among the four methods. Furthermore, it is much faster than the others thanks to the utilization of the tracking scheme and the manner in which centers are initialized in k-means. In fact, the cost of the combination of the original k-means algorithm and CHAT, i.e., CHAT+k-means, is approximately 120 ms per frame. Thus, the speed-up effect is significant. The graph-cut method has good position precision, but it is time-consuming. The meanshift method generates less incomplete segmentation, which affects the precision. Furthermore, CHAT also exhibits good performance. Indeed, CHAT plays an important role for the meanshift and the graph cut because it narrows the search scope and thereby eliminates potential environmental interference.

6.2 Time synchronization experiments

For a video clip captured under fixed exposure time, the two time parameters were obtained by solving the optimization problem in (13). Three elements in several frames are requisite: the position of the head apex, the yaw angle of the head, and the position of the endpoint. The yaw angles and positions of the head apex are products of the pose measurement. Endpoints were marked manually here. To prevent PSO from getting stuck in a local optimum, the optimization was executed several times with different initial values for each case. Scattered and adequate marked endpoints (more than 15 from experience) are conducive to the acquisition of precise solutions. Further, it is beneficial to set the bounds of T as appropriate as possible. The frame rate of the camera ranged from 20 fps to 25 fps. Therefore, we set $T_L = 0.04$ and $T_U = 0.05$. We observed that the PSO with a population size of 50 usually converges to a solution within 40 iterations during a single run. The converged solutions of several runs were always the same value, which was regarded as the optimum. The time parameters calculated for the video clip shown in Figure 9 are $t_1 = 13.33$ ms and $T = 43.39$ ms.

The effect of time synchronization can be demonstrated intuitively through the virtual skeleton, i.e., the skeleton state estimated by (9). As shown in Figure 10, the virtual skeleton was superimposed on the robotic fish in the video clip. It can be seen that the virtual skeleton fits well with the shape of the robotic fish from the beginning to the end. Specifically, the virtual endpoint and the real endpoint remain overlapped virtually continuously. The deviation in a total of 81 frames was calculated by the manually marked endpoints, as shown in Figure 11. The endpoint deviation in the first frame is 4.7 mm, which indicates the accuracy of the parameter t_1. In Figure 11, the curve labeled $\Delta T = 0$ shows the deviation when T is calculated by the time synchronization method, and $\Delta T = 1$ ms denotes the condition when the calculated T is added with an offset of 1 ms. The case of $\Delta T = 1$ ms reveals the cumulative effect of time error, which is of a consistent tendency with the theoretical instance illustrated in Figure 7(b). By contrast, the deviation in the case of $\Delta T = 0$ fluctuates over a relatively small range. The deviation originates mainly from the three-dimensional (3D) movement of the robotic fish, i.e., roll and pitch, which were not in consideration. The virtual skeleton is always planar, whereas the robotic fish appearing in an image is a projection from 3D space. The projection is affected by the 3D attitude of the robotic fish. In addition, the pose error introduced by the pose measurement also contributes to the deviation. Even though the deviation exists, the experimental results still illustrate the accuracy of the estimated time parameters. We therefore conclude that the time synchronization approach realizes sub-millisecond precision for T.

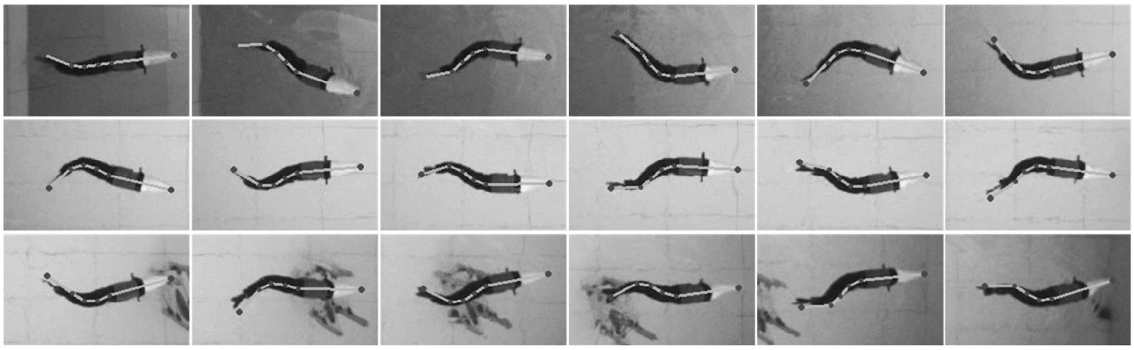

Figure 10   Virtual skeleton with time synchronized: from Frame 1 to Frame 306 with a uniform interval of 18 frames.

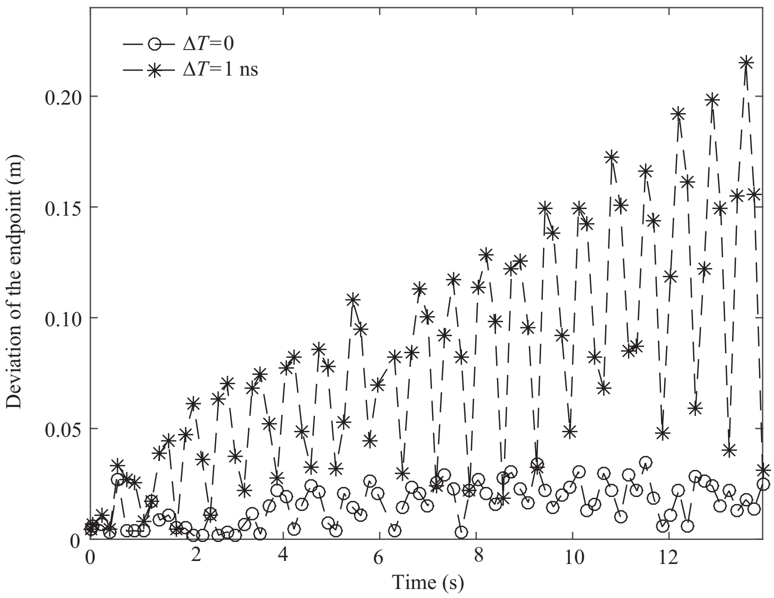

Figure 11   Deviation of the real endpoint and the virtual endpoint.

6.3 Velocity estimation experiments

We employed smoothing splines to fit the curves of poses versus time. Then, we estimated the velocity based on the obtained curves. An instance is shown in Figure 12, in which the smoothing parameter is set as $\rho = 0.999$. Virtually all of the discrete pose points are lying on the curves. Thus, it appears that the smoothness effect is not obvious for the curves of poses versus time. However, the derivative operation can amplify noise and make it visible, which is demonstrated by the case where $\rho = 1$ in Figure 12. Specifically, the condition $\rho = 1$ indicates that no roughness penalization is inflicted on the fitting process, and therefore no smoothness is exerted on the noise. By contrast, the estimated velocity is much smoother in the case where $\rho = 0.999$ in Figure 12. It is revealed that even a slight change in the smoothing parameter will result in different smoothness effects on the estimation of velocity. However, no criteria exist that can be applied to evaluate the accuracy of the estimated velocity owing to lack of standard data. The model of the noise is indeterminate. As a result, deriving a criterion to determine the optimal smoothing parameter is difficult. The tradeoff between the goodness and smoothness of fit needs to be accomplished via trial and error.

7 Conclusion and future work

In this paper, we proposed a method for planar motion measurement of a swimming multi-link robotic fish. The method is a global-vision based approach and mainly focuses on three motion measurement

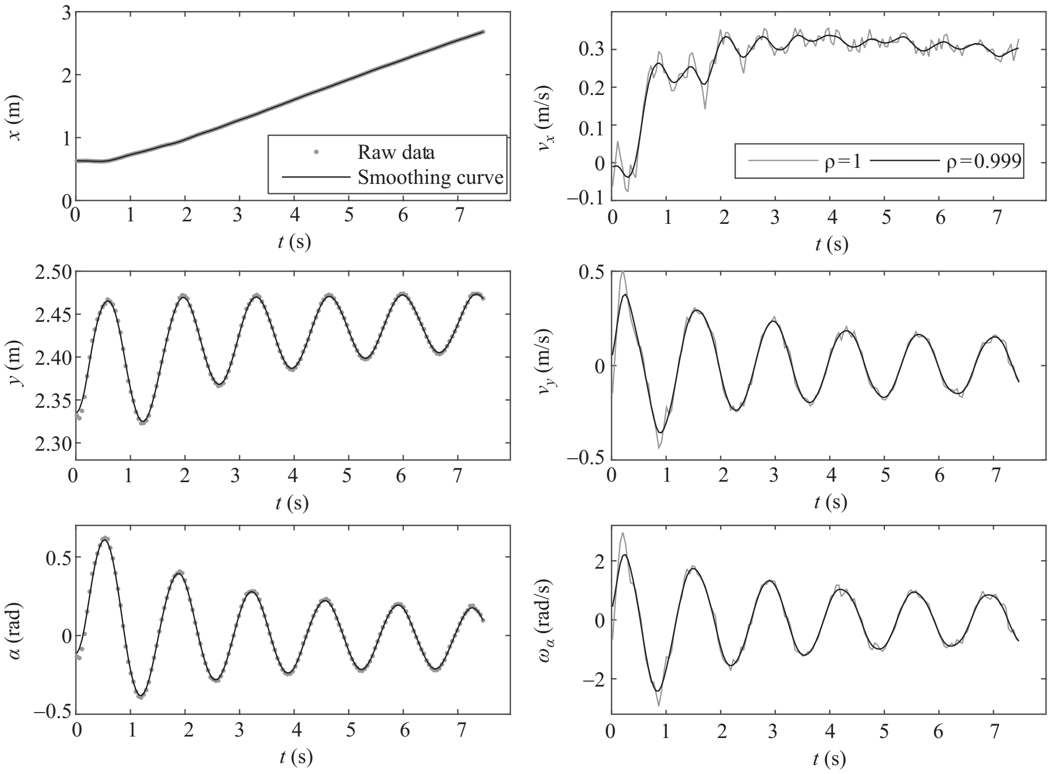

Figure 12 Smoothing curves of pose (left side) and estimated velocity (right side).

aspects. First, a systematic tracking scheme was developed to continuously detect and track the robotic fish in captured image sequences. Second, a synchronization algorithm that synchronizes the time of the motion measurement system and the on-board control system was proposed. Third, for the purpose of noise reduction, smoothing splines were applied to fit curves of poses versus time. Experimental results demonstrate that the tracking scheme achieves an average position error of 6.7 mm and an average angular error of 1.9°, with an average time cost per frame of 30.3 ms. Further, the proposed time synchronization approach realizes sub-millisecond precision for the frame period.

Future work will focus on improving the performance of the planar motion measurement by increasing the number of cameras and synthesizing shape features of the markers and their relative positions.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 61375102, 61333016, 61421004), Beijing Natural Science Foundation (Grant No. 3141002), and State Key Laboratory of Alternate Electrical Power System with Renewable Energy Sources (Grant No. LAPS16006).

Conflict of interest The authors declare that they have no conflict of interest.

References

1 Su Z S, Yu J Z, Tan M, et al. Implementing flexible and fast turning maneuvers of a multijoint robotic fish. IEEE/ASME Trans Mechatron, 2014, 19: 329–338

2 Kopman V, Laut J, Acquaviva F, et al. Dynamic modeling of a robotic fish propelled by a compliant tail. IEEE J Oceanic Eng, 2014, 40: 209–221

3 Swain D T, Couzin I D, Leonard N E. Real-time feedback-controlled robotic fish for behavioral experiments with fish schools. Proc IEEE, 2012, 100: 150–163

4 Takada Y, Koyama K, Usami T. Position estimation of small robotic fish based on camera information and Gyro sensors. Robotics, 2014, 3: 149–162

5 Wang W, Xie G M. Online high-precision probabilistic localization of robotic fish using visual and inertial cues. IEEE Trans Ind Electron, 2014, 62: 1113–1124

6 Fontaine E, Lentink D, Kranenbarg S, et al. Automated visual tracking for studying the ontogeny of zebrafish swimming. J Exp Biol, 2008, 211: 1305–1316

7 Butail S, Paley D A. Three-dimensional reconstruction of the fast-start swimming kinematics of densely schooling fish. J R Soc Interface, 2012, 9: 77–88

8 Porez M, Boyer F, Ijspeert A J. Improved Lighthill fish swimming model for bio-inspired robots: modeling, computational aspects and experimental comparisons. Int J Robot Res, 2014, 33: 1322–1341

9 Marchese A D, Onal C D, Rus D. Autonomous soft robotic fish capable of escape maneuvers using fluidic elastomer actuators. Soft Robot, 2014, 1: 75–87

10 Yu J Z, Wang S, Tan M. A parallel algorithm for visual tracking of multiple free-swimming robot fishes based on color information. In: Proceedings of IEEE International Conference on Intelligent Systems and Signal Processing, Changsha, 2003. 359–364

11 Kopman V, Laut J, Polverino G, et al. Closed-loop control of zebrafish response using a bioinspired robotic-fish in a preference test. J R Soc Interface, 2013, 10: 20120540

12 Butail S, Bartolini T, Porfiri M. Collective response of zebrafish shoals to a free-swimming robotic fish. PLoS ONE, 2013, 8: e76123

13 Huck T, Westenberger A, Fritzsche M, et al. Precise timestamping and temporal synchronization in multi-sensor fusion. In: Proceedings of IEEE Intelligent Vehicles Symposium, Baden, 2011. 242–247

14 Romer K. Time synchronization and localization in sensor networks. Dissertation for Ph.D. Degree. Zurich: Swiss Federal Institute of Technology Zurich, 2005. 97–137

15 Liu T, Hu Y, Hua Y, et al. Study on autonomous and distributed time synchronization method for formation UAVs. In: Proceedings of IEEE International Joint Conference of FCS, Denver, 2015. 293–296

16 Nakadai K, Nakajima H, Murase M, et al. Real-time tracking of multiple sound sources by integration of in-room and robot-embedded microphone arrays. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, 2006. 852–859

17 Lim A, Mizumoto T, Cahier L K, et al. Robot musical accompaniment: Integrating audio and visual cues for real-time synchronization with a human flutist. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, 2010. 1964–1969

18 Nilsson O, Handel P. Time synchronization and temporal ordering of asynchronous sensor measurements of a multi-sensor navigation system. In: Proceedings of IEEE/ION Position Location and Navigation Symposium, Indian Wells, 2010. 897–902

19 Wu Z X, Yu J Z, Su Z S, et al. Towards an Esox lucius inspired multimodal robotic fish. Sci China Inf Sci, 2015, 58: 052203

20 Tomasi C, Manduchi R. Bilateral filtering for gray and color images. In: Proceedings of IEEE International Conference on Computer Vision, Bombay, 1998. 839–846

21 Hartigan J A, Wong M A. A k-means clustering algorithm. J Roy Stat Soc C-App, 1979, 28: 100–108

22 Wu Z X, Yu J Z, Tan M, et al. Kinematic comparison of forward and backward swimming and maneuvering in a self-propelled sub-carangiform robotic fish. J Bionic Eng, 2014, 11: 199–212

23 Kennedy J. Particle swarm optimization. In: Sammut C, Webb G, eds. Encyclopedia of Machine Learning. Berlin: Springer, 2010. 760–766

24 Ebbesen S, Kiwitz P, Guzzella L. A generic particle swarm optimization Matlab function. In: Proceedings of American Control Conference, Montreal, 2012. 1519–1524

25 Reinsch C H. Smoothing by spline functions. Numer Math, 1967, 10: 177–183

26 Comaniciu D, Meer P. Mean shift: a robust approach toward feature space analysis. IEEE Trans Patt Anal and Mach Intell, 2002, 24: 603–619

27 Rother C, Kolmogorov V, Blake A. Grabcut: interactive foreground extraction using iterated graph cuts. ACM Trans Graph, 2004, 23: 309–314