IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 6, 2019, accepted February 22, 2019, date of publication March 5, 2019, date of current version March 25, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2903148

Efficient PatchMatch-Based Synthesis for Cartoon Animation

CHUANYAN HAO[1], YADANG CHEN[2], AND ENHUA WU[3,4], (Member, IEEE)

[1]School of Education Science and Technology, Nanjing University of Posts and Telecommunications, Nanjing 210023, China
[2]School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China
[3]Department of Computer and Information Science, Faculty of Science and Technology, University of Macau, Macao 999078, China
[4]State Key Laboratory of Computer Science, Institute of Software, University of Chinese Academy of Sciences, Beijing 100190, China

Corresponding author: Chuanyan Hao (hcy@njupt.edu.cn)

This work was supported in part by the NSFC under Grant 61702278, Grant 61802197, and Grant 61602252, in part by the Natural Science Foundation of Jiangsu Province of China under Grant BK20160902, Grant BK20160964, and Grant BK20160967, and in part by the Startup Foundation for Introducing Talent of Nanjing University of Posts and Communications under Grant NY217015.

ABSTRACT Automating the production of 2D hand-drawn animations is a significant and interesting component in computer graphics and vision. However, traditional methods in animation production pipeline always use physically or geometrically based models which are consuming due to complicated and massive computations, reducing their practicability. In this paper, we propose an efficient data-driven approach to create hand-drawn animations in an automatic manner. The key idea is to employ a correspondence match-based random search process to extract the geometry samples and the global motion pattern in an input animation sequence and then to generate a new output sequence through a coarse-to-fine sample-based synthesis algorithm. Our experiments demonstrate that our method achieves good results with high quality and performance, producing a range of artistic effects that previously required disparate and professional techniques.

INDEX TERMS Animation, data-driven approach, exemplar-based approach, motion tracking and synthesis, PatchMatch, sample based synthesis.

I. INTRODUCTION

Synthesis of hand-drawn animation is an active research area and has many applications in computer graphics, vision and entertainment. Recent advances in motion capture techniques and other motion editing approaches facilitate the generating of animation with ease and realism. Motion capture and retargeting is pioneered by [1] where a specific motion in a sequence of images is tracked and then transferred to a new animation having a different visual appearance. This technique bridges the gap between traditional animation and computer animation. Traditional animation with the merits of being stylized and expressive is seen as time consuming and labor-intensive while computer created animations can be easily reused and transferred to different domains and characters. Therefore, automation of this drawing process is greatly in demand and many investigations have been conducted recently on this topic, including image morphing and deformation [2], image registration and retargeting [3], [4], and recently exemplar-based synthesis techniques [5]–[8].

The associate editor coordinating the review of this manuscript and approving it for publication was Lei Wei.

In this paper, we are also interested in how to capture the motion pattern in a given animation and produce a new one which has the similar appearance and motion with the input sequence. Shortly speaking, given a source cartoon animation video, our goal is to automatically generate such repetitions through a combination of random search process (seeking the motion pattern) and data driven computation (synthesizing details). Fig. 1 gives an example of our work, in which the left column is the source animation video represented by picked frames and the right one is the output animation sequence after synthesizing.

Different from most previous approaches which always capture motions by user inputs or data capture devices, we focus on an automatic treatment to track the motion. Considering that the visual similarity between drawing images or frames is an important cue to find the correspondences between two image regions, our method attempts to track the motion through an exemplar-based neighborhood search procedure in terms of visual diversity. Such an exemplar-based framework has been applicable for many hot issues, including texture synthesis, image inpainting and completion, image registration, motion field simulation, dense correspondences

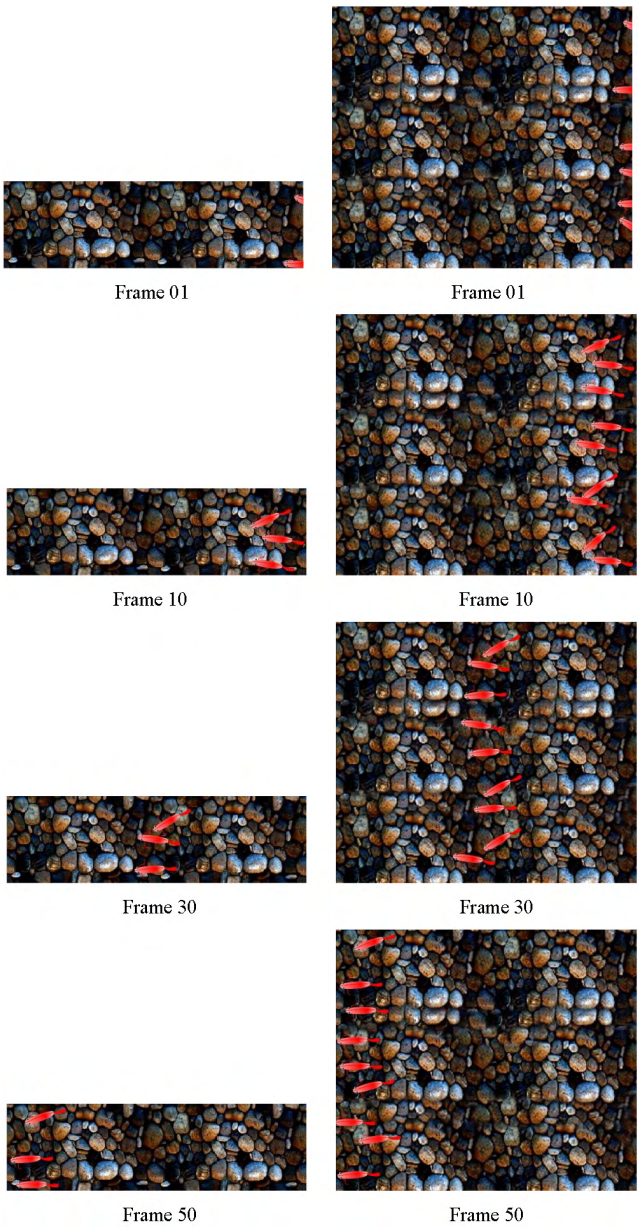

FIGURE 1. An example of our work.

matching, optical flow calculation and lots of other applications in graphics, vision and image processing. In contrast, as a fundamental part in image editing and scene correspondence matching, the nearest neighbor field (NNF) is closely related to solve our motion estimation problem. The aim of NNF computation is to find one or more nearest neighbors (visually similar) for each patch between a pair of images. The main difficulty in this technique is the computational complexity caused by the exhaustive search. To improve the efficiency of computing NNF, a seminal work called PatchMatch [9], [10] was proposed, the core idea behind which is a random search procedure and a coherence propagation methodology among neighbors. Enlightened by this boost, we propose a grid based PatchMatch method to efficiently and automatically obtain spatial-temporal samples [6], [7] and the global motion pattern for the next stage of synthesis task.

In the aspect of reusing a motion pattern, the mainstream methods prefer physically or geometrically based computations to find the points movements on which the action can be created through the inbetween or interpolation technique. This type of approaches are often too complicated to solve and limited to some kind of physical phenomena or geometrical topology. On the other hand, motion sequences can be regarded as stochastic processes in many cases, as well as texture images are. Meanwhile, cartoon images are usually composed of elements with nature of repetitions on both geometry and dynamics. Based on such observation, we can apply analogous methodology of texture synthesis to regenerate repetitive motion sequences. The difference lies on that texture images assume a spatial distribution while motion pattern in cartoon images display a temporal-spatial distribution. In hence, through extending the outstanding work in [6] and [7], we propose a coarse-to-fine sample-based local neighborhood similarity matching algorithm to synthesize the motion sequence similar to the input.

II. RELATED WORK

This section simply reviews several closely related methods to our work, including exemplar-based image and cartoon matching and data-driven animation synthesis.

A. EXEMPLAR-BASED IMAGE AND CARTOON MATCHING

Matching correspondences between images, such as photograph images, hand-drawn images, frame images, heterogeneous images [11] and so on, is always a challenging task attacking the attentions of many researchers within the last two decades. Related methods to matching or registering objects can be learned in surveys [12] and [13]. Recently, the high coherence [14] and rich color information [15] are employed to match the correspondence between frame images, however a hand-drawn cartoon animation always lacks this kind of information. Qiu *et al.* [16] proposed an algorithm to segment cartoon images into different parts, like head region or arm region, and match these closed regions, but it cannot work well on dense correspondences because fails to generate correspondence on pixel level. Although de Juan and Bodenheimer [17] employ dense correspondences in their work, its initialization is fully by hand. A fully automatic method is demonstrated in [4] which uses region dense correspondence matching based on the algorithm in [18]. Yet it is very slow due to exhaustive searches to find the matches of the patches in the source image between regions. By contrast, Barnes *et al.* [9], [10] demonstrated an example-based matching algorithm, called PatchMatch, to obtain dense correspondences between images and have been successfully applied to a broad range of vision problems [19], [20]. The key idea contains two steps, coherence propagation and picking up patches randomly, which are then optimized and converge in iterations. This algorithm dramatically reduces the

number of patch comparisons and achieves greater speed. In this paper, we also focus on fully automatic and efficient method to match cartoon frames, therefore, we propose a grid-based PatchMatch technique to accelerate the matching process while also obtaining the geometry description and the global motion pattern, which are difficulty to be extracted automatically and sufficiently with [6].

B. DATA-DRIVEN ANIMATION SYNTHESIS

Creating artistic effects of cartoon animations has attracted growing attentions in computer graphics and computer vision. There are lots of related topics included, such as deformation or warping techniques to generate animation sequences through radial basis functions, thin plate splines, energy minimization, multilevel free-form deformations (FFD) and so on [2], the use of as-rigid-as-possible interpolation for cartoon-like images by sampling the space of possible deformations [3], [21], and an improved deformation algorithm based on Moving Least Squares (MLS) [18] which solves a smaller linear system and achieves better quality and performance. Since there are so many techniques presented, this section only focuses on synthesizing cartoon animation using data-driven approaches. Early representatives are texture synthesis algorithms through non-parametric sampling procedures, as surveyed by [8]. This kind of approaches reproduce the output texture visually similar to the input exemplar by color based spatial neighborhood similarity matching. Then several approaches extend the spatial neighborhood searches to the temporal domain to synthesize animated sequences, like the works in [5], [22], and [23]. Another set of methods define the stochastic and repetitive behaviors in video sequences as textural motions [24], [25] and propose to synthesize the infinitely looping sequences by modeling the low cost transitions between frames. All the above schemes assume continuous distribution in texture regions while Ma *et al.* [7] note that many natural appearances are composed of similar discrete elements and hence use the irregular neighborhoods matching algorithm [7], [26]. After applying the irregular neighborhoods in temporal domain, work in [6] enables a more general framework for dynamic element textures, from which we draw inspiration to pursue benefits of data-driven approaches for synthesizing dynamic contents in 2D animation. Besides, there are other interesting data driven synthesis methods offering the state-of-the-art results, but most of them aim at specific applications, such as cloth wrinkles [27], crowds [28], faces cartooning [29], [30] and learning basketball dribbling skills [31]. Therefore, we draw our inspiration from all the above investigations to pursue similar benefits of data-driven approaches for synthesizing dynamic contents in 2D animation, which tends to be more general and user friendly.

III. OUR APPROACH

The core idea of our method is to obtain samples automatically by an efficient grid-based random search procedure. Then these samples can be used to synthesize the 2D dynamic details in a hierarchical architecture which conducts in a coarse-to-fine (top-to-down) scheme. Our matching and synthesis framework works in a fully automatic manner, and achieves good visual effects and performances in the output animated sequences. We discuss these two parts separately in the rest of this section.

A. GRID-BASED PATCHMATCH FOR AUTOMATIC MOTION SAMPLING

The aim of our motion sampling is to collect the samples which can describe the cartoon elements in an hand-drawn animation. Most traditional methods use physical or geometrical based models which are complicated and expensive to solve or require carefully parameter tuning. Comparatively, exemplar-based approaches offer better alternatives through dense correspondence matching techniques. A potential limitation of them is the huge amount of patch comparisons in the exhaustive search process, making them slow and even nonapplicable for high resolution images such as [4], but the randomized search strategy in PatchMatch [9] brings significant benefits to the solution of this problem, catering for automating the capture of motion information in an animation. On the other hand, considering that cartoon scenes are often drawn with clearly defined lines and a finite number of colorful regions which are easy to recognize compared with the NNF problem in [9], our matching approach is defined as to find the best correspondence on grid vertices rather than each pixel of the image.

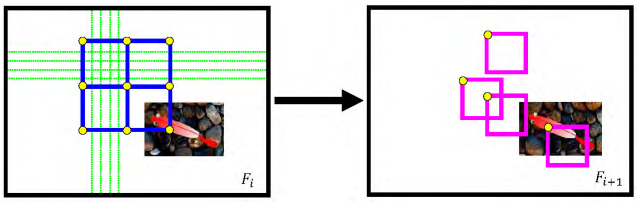

FIGURE 2. Our grid-based randomized search.

Firstly, we embed the source frames in a uniform partitioned grid with a span of d pixels to obtain these points, as shown in Fig. 2 where the dashed green lines, one the left image, are used to indicate the original pixels, the blue lattice is for the uniform grid and the yellow points represent the vertices on the crosses of the grid in the frame of F_i, while on the right, the magenta blocks are the matched patches corresponding to the blue grids in the neighbor frame of F_{i+1}, also starting at the left corner of a block as the same yellow points shown. Formally, given a pair of neighbor frames $F_i, F_{i+1} \subset R^2$ and a collection of vertices $V = \{v_j\}$ on their meshes with the location of $P = \{p(v_j)\}$, our goal is to determine the motion of each vertex. Let $p(v_j^i)$ be the original position of the current vertex v_j in frame F_i and $p(v_j^{i+1})$ be its corresponding location in frame F_{i+1} after the animation playing from frame F_i to F_{i+1}, the motion could be derived as

$M = p(v_j^{i+1}) - p(v_j^i)$. In order to find the new location $p^{i+1}(v_j)$ for the vertex v_j in its neighbor frame F_{i+1}, our approach carries out the neighborhood propagation and random search steps iteratively in scanline order on odd iterations and reverse scanline order on even iterations alternatively, similarly to the PatchMatch algorithm [9] dose.

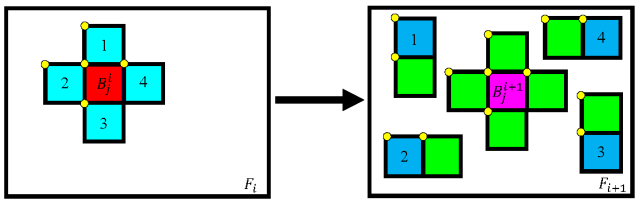

FIGURE 3. The candidate set in propagation step.

After the initialization on the vertices of the grid, each point is examined in the loop by neighborhood matching. Concretely speaking, for the current vertex v_j, we denote that its neighborhood range yields to its spatially adjacency according to the uniform grid. That is, as the propagation step, for the current block B_j^i starting at the left corner of the grid point $v_j^i(x, y)$ in F_i, our method sets up a candidate set $S(v_j^i)$ in F_{i+1} for B_j^i with a shift of d pixels or one grid as its initial neighbors. For the sake of coherence, we also add its relative neighbors to the candidate set in terms of the k-coherence algorithm [32]. So here, the candidate set is defined as $S(v_j^i) = \{(B_j^{i+1} + \triangle) \cup (B_{j+\triangle}^{i+1} + \triangle)\}$ where $\triangle$ takes the values of $(0, 0), (d, 0), (0, d), (-d, 0), (0, -d)$ as the displacements. As shown in Fig. 3, the current block B_j^i (red) and its neighbors $B_{1,2,3,4}$ (cyan) with one grid width in frame F_i are on the left image, and respectively, the magenta and blue blocks on the right image are for their best matches examined during the previous iteration, then the green blocks around the magenta one and those one grid cell shifting to the left, or right, or up, or down of the blues are all sent to the candidate set. Now the new location $p(v_j^{i+1})$ in the propagation step can be determined by the following minimization,

$$\arg\min \sum_{B_k \in S_j^i} |B(p(v_j^i)) - B_k(p(v_j^{i+1}))|^2 \qquad (1)$$

where k takes values from 1 to s, given s is the number of the candidates in S_j^i, i from 1 to f, the number of the frames and j from 1 to n, the number of the vertices in the grid, and $|\cdot|^2$ is the visual measurement to compare the blocks.

Next, $p(v_j^{i+1})$ is going on to be updated in the randomized search step. Also like PatchMatch [9], a sequence of random sampled blocks around their vertices are evaluated in a set of exponentially decreasing windows centering at the current best $p(v_j^{i+1})$ and starting from a maximum search radius such as the boundary of the frame. The decrement ratio among these windows is generally set to $0.5 \times n$, the number of vertices. The visual criteria follows (1) as well only with the B_k being picked up from search windows rather than the candidate set. Thanks to the good identifiability of cartoon images, we also limit the maximum search radius to cover the most relative neighborhoods so to avoid some ambiguous choices and to reach higher efficiency. At last, this algorithm converges within a fixed number of iterations although convergence criteria changes depending on different drawings and the motion data of the rest pixels is filled by using bilinear interpolation. Algorithm 2 gives an illustration.

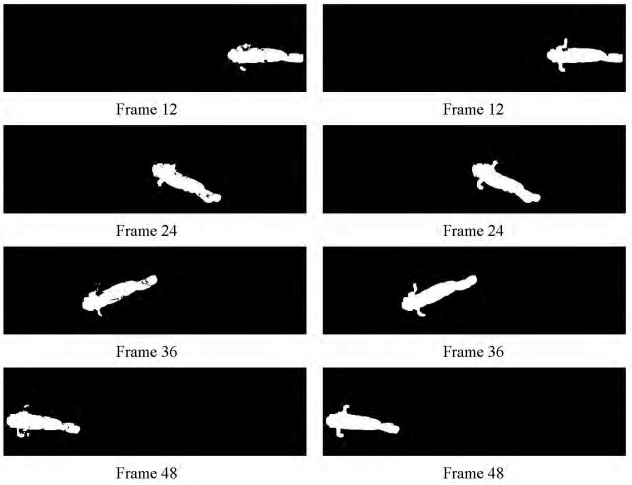

FIGURE 4. Our automatic motion capture.

It is worth to notice that our framework directly works on vertices not pixels, which may break the continuity in the propagation step and encourage the discontinuity in the randomized search step, causing that some visually similar but spatially distant blocks will be selected as the best matches. However, in practice we have found that such an expectation is an unusual case probably because a cartoon image has relatively simple lines and colors, iterations raise the odd of finding a good match and the restriction on search radius is helpful for maintaining the coherence. As in Fig. 4, the motion of the red fish in Fig. 1 is obtained through our grid-based randomized search algorithm listed on the left column. We also compared our method to the PatchMatch algorithm whose results are listed on the right column. It is easy to say that the results from PatchMatch are denser (the body) and more continuous (the whisker) than ours, however, our approach reaches eight times acceleration and more significantly, they have been able to provide sufficient samples for geometry details and the global motion pattern, as elaborated in Section III-B1.

B. COARSE-TO-FINE SAMPLE-BASED SYNTHESIS WITH GLOBAL MOTION CONSTRAINT

In this section, we present our synthesis framework and discuss its main features. This novel framework improves the sample based synthesis methods [7], [26] upon a hierarchical architecture and works in a coarse-to-fine manner. We first detail how to collect the geometry samples and extract the global motion pattern automatically based on our matching system as described in Section III-A, then

elaborate the synthesis procedure on one level of the pyramid in Section III-B2, and finally describe the hierarchical structures of our approach as well as the search step among the levels in Section III-B3.

1) AUTOMATIC SAMPLING OF GEOMETRY DETAILS AND GLOBAL MOTION PATTERN

At present, many animation synthesis approaches require the motion pattern of the input sequence containing only details without global structures, such as [6], [24], and [25] where these repetitive details can be regarded as textural motions and be modeled as the stochastic distribution similar to the image textures. However, such feature also limits the use of this kind of methods to just handle textural repetition patterns and geometric topologies with sufficiently small and local scales. In hence, Ma *et al.* [6] propose an analysis tool in order to decompose the general input into local and global motions and geometries. But their analysis merely provides a coarse geometry samples through a low pass filtering of the original sample positions which are picked up by users, and a coarse global motion description.

FIGURE 5. Sampling the geometry topology through the bounding rectangle on the control map.

Since we have already obtained the motion flow that is dense enough in Section III-A, our approach caters for automating the acquisition of the geometry samples $C(q)$ and gives a fine-scale global motion tracking scheme. First, we compute the geometry center on the mask of each corresponding frame by solving its bounding rectangle as shown in Fig.5 and samples its boundary pixels at the interval of 45 degree, which represent the shortest distances, longest distances and the middle ones to the geometry center. Then, we can define the motion pattern of the animation character as the difference computation of the shifts of these samples between adjacent time frames. As illustrated in (2), $v(q,t)$ is the velocity of the sample q at frame $F(t)$ and it describes the global advection from frame to frame following the sample velocities. Later, this motion item will be added to the subsequent synthesis stage as a constraint term.

$$v(q,t) = (F(q,t) - F(q,t-1)) \cdot 1/\Delta t \tag{2}$$

2) BASIC SYNTHESIS WITH MOTION CONSTRAINT

In this section we detail the synthesis procedure on one level of the pyramid. Our basic methodology is enlightened by the work of [6] which is inspired by texture optimization [33]. In hence, given I be the input exemplar, then the corresponding synthesis output, denoted as O, can be achieved by minimizing the general energy function as in (3). The goal of it is to find the best matched input sample $q_i \in I$ for each output sample $q_o \in O$ with the most similar neighborhood.

$$E(O_t|I) = \sum_{q_o \in O_t} \min_{(q_i,t_i)\in I} |D(W(q_o,t)) - D(W(q_i,t_i))|^2 + \lambda e(O_t|I) \tag{3}$$

Specifically, the first term in (3) measures the difference between the spatial-temporal local neighborhoods of the input sample q_i and output one q_o via the sum of squared distances, in which $W(\cdot)$ indicates the neighborhoods and $D(\cdot)$ for the measurement criteria. The second term retains the potential of our method to deal with application specific problems. In our work, $e(O_t|I)$ is used to introduce the global motion pattern to the energy function, that is, the initialization of q_o in the current frame t can be computed through an Euler integration based on the velocity of q_i and time interval Δt, as seen in (4), where $v(\cdot)$ is determined from (2).

$$e(O|I) : F(q_o,t) = F(q_o,t-1) + v(q_i,t-1)\cdot\Delta t \tag{4}$$

Since cartoon images are generally composed of discrete elements, the local neighborhood $W(\cdot)$ and its metric $D(\cdot)$ are different from those defined in texture images. In order to keep the geometric details and the dynamic features, $W\cdot)$ considers the adjacent relationship between samples across both spatial and temporal regions. Concretely speaking, let Ω be the size of the spatial neighborhood and Δt be that of the temporal stride, then at time t, the spatial-temporal neighborhood of sample q can be defined as the collection of samples which have the relative spatial distance to q within Ω and temporal distance within Δt, formulated as in (5). In our approach, Ω is particularly set in terms of animated elements, like the fish in Fig. 1. Such convention can obtain element based samples easily from our automatic sampling algorithm and also offers benefits for boundary handling and appearance reconstruction.

$$W(q,t) = \{(q',t')\}, \quad |q-q'| < \Omega,\ |t-t'| < \Delta t \tag{5}$$

Then we can express the distance metric $D(\cdot)$ based on the relative differences of each q' with respect to the center sample q, allowing for neighborhoods in different spaces and times to match, as shown in (6). Here, $f(q,t)$ means the features to describe the sample q, including $\{position, color, texture, etc.\}$ and so the first term of (3) can be rewritten as in (7).

$$D(W(q,t))_{(q',t')\in w(q,t)} = \{f(q,t) - f(q',t')\} \tag{6}$$

In summary, we can schematically demonstrate the above synthesis method as in Fig. 6, where the energy of neighborhood $W(q_o,t)$ centered around sample q_o is given by its distance to the closest input neighborhood $W(q_i,t)$ and this distance enables the energy $E(O_t|I)$ to be minimum. The red and magenta circles represent the samples with green and blue diamonds around, being their element based spatial neighborhood, and when going cross the time segment Δt, the spatial-temporal local neighborhood is constructed. Similar to exemplar-based texture synthesis, the nearest input

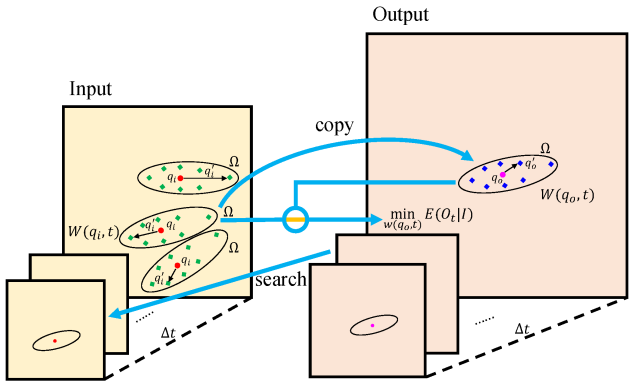

FIGURE 6. Schematic demonstrating our synthesis procedure.

sample is found by exhaustively searching every input sample and at last, the best matching one with the most similar neighborhood is assigned to the being synthesized output sample.

$$\min_{(q'_o,t'_o)\in W(q_o,t),(q'_i,t'_i)\in W(q_i,t)} |\cdot|^2 = |\{f(q_o,t)-f(q'_o,t'_o)\}-\{f(q_i,t)-f(q'_i,t'_i)\}|^2 \quad (7)$$

3) COARSE-TO-FINE STRATEGY

Our basic synthesis is similar to the work of [6] which achieves the-state-of-art results but needs complex computation with some application specific adjustments. Alike, our basic synthesis contains the time-consuming exhaustive search steps. A common way to handle it, as mentioned in [6], is to carry out the exhaustive search for small input exemplars and k-coherence acceleration for big ones, which may lead to less accurate matches. Thus, we introduce a simple but powerful hierarchical architecture with propagation from top to bottom to handle this problem.

First, a pyramid is set up with k levels for both input I and output O with a down-sampling factor $\alpha = 0.5$. Let the lth level of pyramid of frames in I and O as F_i^l and F_o^l, $l \in \{0, 1, \ldots, k-1\}$, the bottom level of the pyramids F_i^0 and F_o^0 are the raw images. Now our goal is to find the matches of every samples in F_o^0 against F_i^0. Then we can construct samples on each level. Given $\{C(q_{o,i}^l)\}$ the positions of samples on the lth level, the downscaled version from the raw samples in $F_{o,i}^0$ can be defined as in (8). Note that, the samples on each level preserve the same neighboring relation as the finest level.

$$\{C(q_{o,i}^l)\} = \alpha \cdot \{C(q_{o,i}^{l-1})\}, \quad l > 1 \quad (8)$$

After the construction of the pyramid and the generation of the samples in each level, we perform the basic synthesis algorithm in Section III-B2 on each level and propagate matched samples also from top to bottom on the pyramid. Fig. 7 gives an illustration. As seen, it works on the pyramid in coarse-to-fine shceme. The sampling domain is spatial-temporal neighborhood with red points representing

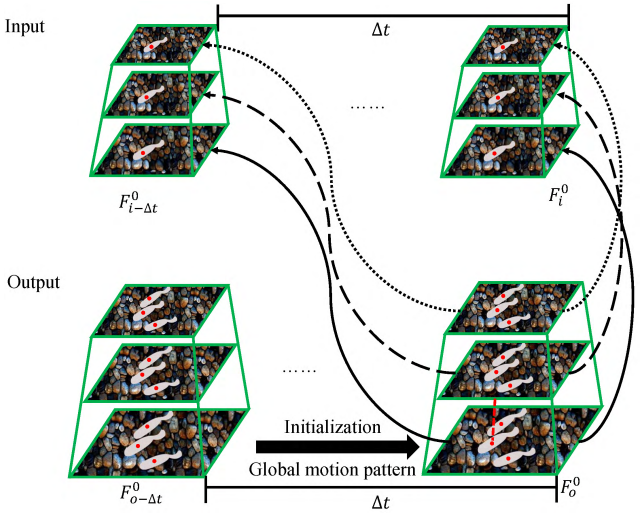

FIGURE 7. Overview of our synthesis procedure.

spatial samples and Δt for temporal stride (details referred to Section III-B2). We firstly initialize the first frame on the top level according to the input exemplars via the patch-based approach only in the spatial domain, like in [7]. Our method then optimizes the output through iterative search and match steps as inspired by Kwatra *et al.* [33]. After the transfer of the motion constraint based on (4), this optimization becomes the initialization of subsequent frames. So far the search and match steps happen between levels in the pyramid and among input frames. The whole process loops until convergence as summarized in Algorithm 1.

IV. RESULTS AND DISCUSSIONS

This section demonstrates our experiments, makes some discussions on the performances and compares with other methods on the matching scheme [4] and synthesis step [6] separately. We have tested our approach on several cartoon animations. All the synthesis results presented in this section are generated on the laptop of Intel Core i5 CUP double @2.30GHz with 8GB memory. Fig. 8 and Fig. 9 show some results of our synthesis framework. As seen in Fig. 8, our synthesis approach achieves good results both for textural cartoon elements with small and local motions and geometries, the top groups, and for characters with large displacements, the rest groups where the first row is a sequence of frames of the source animation, the second row is the corresponding motion pattern automatically captured by our grid-based match method, and the last rows are sequences of frames after synthesis. We also apply our approach to transfer the motion pattern extracted from the input animation to different targets, as shown in Fig. 9, in which the first row includes the input frames followed by their corresponding motion courses in the second row and the synthesized output frames are in the third row. More details about the experiments can refer to Table 1 in which the time unit is second (s) and the neighbor size is measured relative to the bounding rectangle

Algorithm 1 Coarse-to-Fine Synthesis With Motion Constraint

Input: $\{F_{i_t}\}$ frames of an animated sequence, a sample set $\{q_{i_t}\} \in F_{i_t} \leftarrow GRIDPM(F_{i_t}, F_{o_t})$
Output: $\{F_{o_t}\}$ frames of the synthesized animation sequence, $t \in \{0, 1, \ldots, N-1\}$
  Construct the input frame pyramid of $\{F^l_{i_t}\}$ and samples $\{q^i_{i_t}\}$
  for each $F_{i_t}, F_{o_t}, t \in \{0, 1, \ldots, N-1\}$ **do**
    if $t = 0$ **then**
      F_{o_t} is initialized by random noise
      Construct the initial output pyramid $\{F^l_{o_t}\}$ as in (8)
    else
      F_{o_t} is initialized by motion propagation as in (4)
      Construct the initial output pyramid $\{F^l_{o_t}\}$ and samples $\{q^l_{o_t}\}$ as in (8)
    end if
    for each $q^l_{o_t}$ from $\{q^{l-1}_{o_t}\}$ to $\{q^0_{o_t}\}$ in $F^{l-1}_{o_t}$ to $F^0_{o_t}$, $l \in \{0, 1, \ldots, k-1\}$ **do**
      if $l = k-1$ *andt* $= 0$ **then**
        $F^l_{o_t}$, $q^l_{o_t}$ is initialized by spatial patch-based synthesis algorithm [7]
      else
        repeat Optimizing (3)
          search $\rightarrow \{W(q_i, t)\}$ //(5)
          match $\rightarrow D(W(q_i, t)) - D(W(q_o, t))$ //(6)
          assign $\rightarrow \{W(q_o, t)\}$ //(7)
        until Convergence or enough iterations reached
      end if
    end for
  end for

Algorithm 2 Grid-Based PatchMatch

Input: pairs of frame images $F_{i_t}, F_{i_{t+1}}, t \in \{0, 1, \ldots, N-1\}$
Output: sample sets $\{q_{i_t}\}$
  function GRIDPM(F_{i_t}, F_{o_t})
    Construct grid vertices $V = \{v_j\}$ according to the spacing of d pixels
    Random initialization for correspondence map M
    for each $v_j, j \in \{0, 1, \ldots, n-1\}$ **do**
      if $j = 0$ **then**
        picking the initial patch randomly
      else
        repeat Minimizing (1)
          propagation based on k-coherence strategy as in Fig. 3
          for $r = maxserachradius, r \geq 1, r = r \times 0.5$ **do**
            random search
          end for
        until Convergence or enough iterations reached
      end if
    end for
    for all j between pairs of $F_{i_t}, F_{i_{t+1}}$ **do**
      $M = bilinearinterpolation(p(v^{t+1}_j) - p(v^t_j))$
    end forreturn $\{q_{i_t}\} = boundingbox(M)$ //Fig. 5
  end function

as shown in Fig. 5 of the input exemplar with normalized size of 1.

Then we show a simple analysis of the performance of our algorithm. We optimize the parameters of our method on a set of different input animations using the qualitative criterion: root-mean-square-error (RMSE) between the corresponding frames. It turns out that a set of constant parameters works well for most input exemplars: $d, r, n, w = 3, 8, 70, 16$, here d is the grid spacing, r is the search radius, n is the iterations (Depending on the input animations, slight differences available for iterations, average 70 for convergence.) and w is the patch size. Fig. 10 demonstrates the robustness of our method.

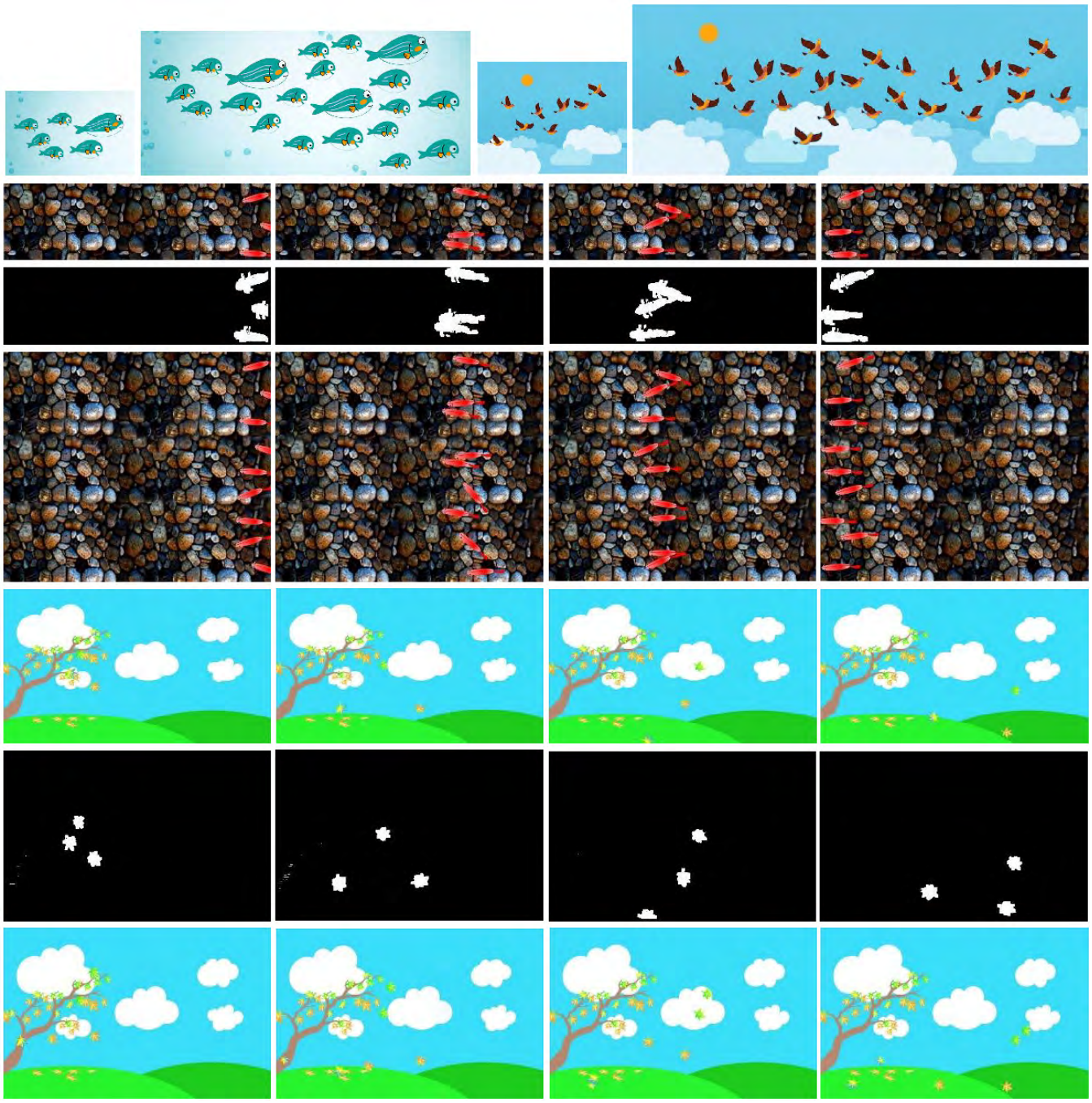

FIGURE 8. Synthesis results by our method. From left to right and top to down, test exemplars are named as green fish, bird, red fish and leaf respectively.

TABLE 1. Parameters and performance for our experiments.

Source	Frames	Size	Spatial Size	Temporal Size	Levels	Matching	Total Time
Green Fish	60	400 × 250	0.25	5	5	13.3s	310s
Bird	70	400 × 300	0.27	5	5	15.5s	480s
Red Fish	60	700 × 200	0.15	1	3	11.2s	370s
Leaf	100	700 × 430	0.3	1	3	12.7s	420s
Pencil	60	700 × 200	0.25	3	3	11.2s	200s
Butterfly	100	700 × 400	0.3	3	3	13.5s	300s
Smile Face	100	700 × 700	0.3	3	3	13s	150s

We also make a comparison between our implementation and the work in [4] for the matching procedure to find the correspondence among frames as shown in Fig. 11, the left one, where it seems our approach many demand more iterations, but since each iteration is much faster in the random search model, our algorithm is still faster. Compared with

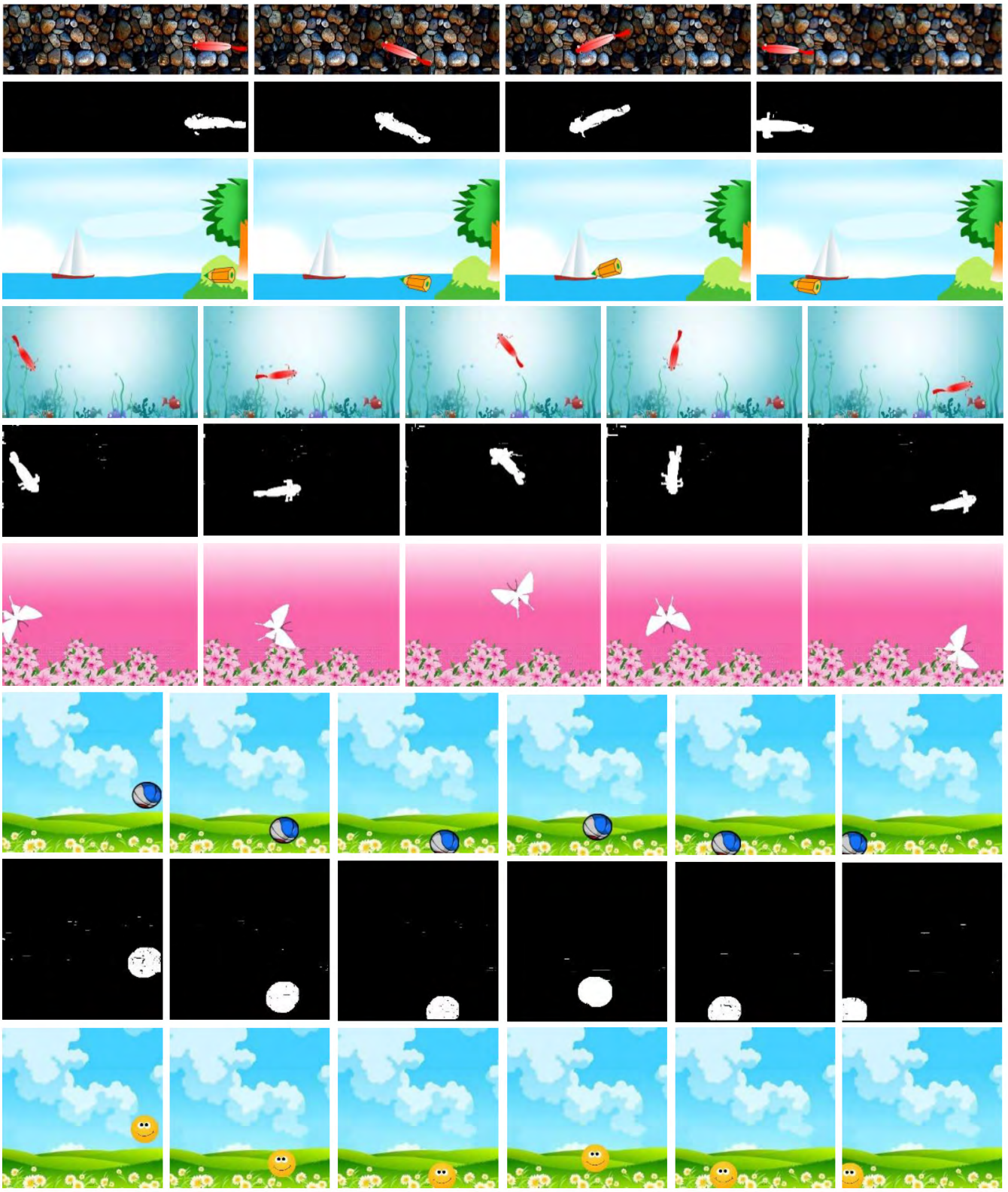

FIGURE 9. Transfer results by our method. From top to down, test exemplars are named as pencil, butterfly and smile face respectively.

the work of [6], our hierarchical structure works more easily and efficiently for textural geometries and motions with small and local scales, and meanwhile it can generate smooth outputs without obvious temporal jitters for elements with large relative motions between each other, which is difficulty to solve in [6]. The right figure in Fig. 11 presents that

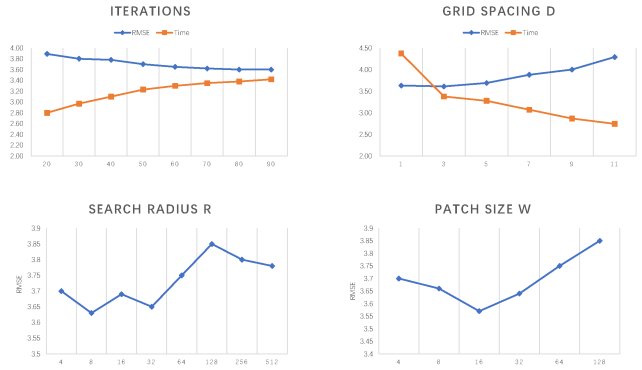

FIGURE 10. Parameter analysis for efficiency. RMSE is for root-mean-square-error and time is rewritten by log(2).

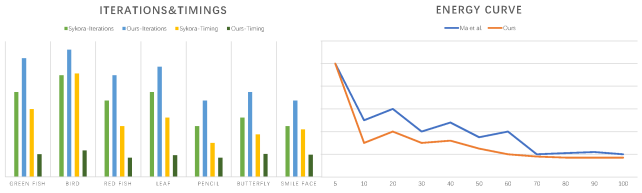

FIGURE 11. Comparison with other methods.

our hierarchical framework obtains better convergence when gradually decreasing the energy of (3).

V. CONCLUSION AND FUTURE WORK

In this paper, we demonstrates a simple and effective data-driven synthesis approach to create hand-drawn animations in an automatic manner. This task is achieved by two stages, a grid-based PatchMatch random search algorithm to automatically extract the global motion pattern in the input animated sequences and the coarse-to-fine sample-based synthesis architecture to generate new output animations which have similar geometries and motions to the input exemplars. The experiments show that our method can work successfully with plausible results and good performance. However, there are still limitations in our system. First, our current matching algorithm uses simple input animations with static backgrounds. As demonstrated in other methods such as [20] and [34], a potential direction is to incorporate more general query and match computations into our synthesis framework to produce outputs that more faithfully handle complex input behaviors in backgrounds. In addition, our method now just investigates on relatively elementary characters which do not have shape changes of themselves since a complex cartoon includes too much challenging problems, such as how to make sure each part of it, like body, head, leg even toes, and how to capture the complicated actions of the each part. Therefore, some ideas deserve the future studies on this topic, including the parallelization of our matching algorithm in the GPU for real-time processing, deep investigations on synthesis of complex cartoon motions, extending to other domains such as flows motions, facial changes, synthesis on captured input videos and so on.

REFERENCES

[1] C. Bregler, L. Loeb, E. Chuang, and H. Deshpande, "Turning to the masters: Motion capturing cartoons," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 399–407, Jul. 2002. [Online]. Available: http://doi.acm.org/10.1145/566654.566595

[2] G. Wolberg, "Image morphing: A survey," *The Vis. Comput.*, vol. 14, no. 8, pp. 360–372, 1998. doi:10.1007/s003710050148.

[3] T. Igarashi, T. Moscovich, and J. F. Hughes, "As-rigid-as-possible shape manipulation," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1134–1141, Jul. 2005. [Online]. Available: http://doi.acm.org/10.1145/1073204.1073323

[4] D. Sýkora, J. Dingliana, and S. Collins, "As-rigid-as-possible image registration for hand-drawn cartoon animations," in *Proc. 7th Int. Symp. Non-Photorealistic Animation Rendering (NPAR)*, New York, NY, USA, 2009, pp. 25–33. [Online]. Available: http://doi.acm.org/10.1145/1572614.1572619

[5] C. Ma, L.-Y. Wei, B. Guo, and K. Zhou, "Motion field texture synthesis," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 110:1–110:8, Dec. 2009.

[6] C. Ma, L.-Y. Wei, S. Lefebvre, and X. Tong, "Dynamic element textures," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 90:1–90:10, Jul. 2013. [Online]. Available: http://doi.acm.org/10.1145/2461912.2461921

[7] C. Ma, L.-Y. Wei, and X. Tong, "Discrete element textures," *ACM Trans. Graph.*, vol. 30, no. 4, pp. 62:1–62:10, Aug. 2011.

[8] L.-Y. Wei, S. Lefebvre, V. Kwatra, and G. Turk, "State of the art in example-based texture synthesis," in *Proc. Annu. Conf. Comput. Graph. Eur.* Munich, Germany: Eurograph. Assoc. Comput. Graph., 2009. [Online]. Available: http://www.eurographics2009.de/program/stars/

[9] C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman, "PatchMatch: A randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 24:1–24:11, Jul. 2009.

[10] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein, "The generalized patchmatch correspondence algorithm," in *Proc. 11th Eur. Conf. Comput. Vis. Conf. Comput. Vis., III (ECCV)*, Berlin, Germany: Springer-Verlag, 2010, pp. 29–43.

[11] J. Li, C. Li, T. Yang, and Z. Lu, "Cross-domain co-occurring feature for visible-infrared image matching," *IEEE Access*, vol. 6, pp. 17681–17698, 2018.

[12] B. Zitová and J. Flusser, "Image registration methods: A survey," *Image Vis. Comput.*, vol. 21, pp. 977–1000, Oct. 2003.

[13] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Comput. Surv.*, vol. 38, no. 4, p. 13, 2006.

[14] S. Vijayanarasimhan and K. Grauman, "Active frame selection for label propagation in videos," in in *Proc. ECCV*, 2012, pp. 496–509.

[15] B. Alexe, T. Deselaers, and V. Ferrari, "What is an object?" in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2010, pp. 73–80.

[16] J. Qiu, H. S. Seah, F. Tian, Q. Chen, Z. Wu, and K. Melikhov, "Auto coloring with enhanced character registration," *Int. J. Comput. Games Technol.*, vol. 2008, pp. 2:1–2:7, Jan. 2008. doi: 10.1155/2008/135398.

[17] C. N. de Juan and B. Bodenheimer, "Re-using traditional animation: Methods for semi-automatic segmentation and inbetweening," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation (SCA)*, Aire-la-Ville, Switzerland: Eurographics Association, 2006, pp. 223–232. [Online]. Available: http://dl.acm.org/citation.cfm?id=1218064.1218095

[18] S. Schaefer, T. McPhail, and J. Warren, "Image deformation using moving least squares," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 533–540, Jul. 2006. [Online]. Available: http://doi.acm.org/10.1145/1141911.1141920

[19] C. Hao, Y. Chen, W. Wu, and E. Wu, "An iterated randomized search algorithm for large-scale texture synthesis and manipulations," *Vis. Comput.*, vol. 31, no. 11, pp. 1447–1458, 2015.

[20] C. Barnes, F.-L. Zhang, L. Lou, X. Wu, and S.-M. Hu, "PatchTable: efficient patch queries for large datasets and applications," *ACM Trans. Graph. (TOG)*, vol. 34, no. 4, p. 97, Jul. 2015. [Online]. Available: http://doi.acm.org/10.1145/2766934

[21] M. Alexa, D. Cohen-Or, and D. Levin, "As-rigid-as-possible shape interpolation," in *Proc. 27th Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH)*. New York, NY, USA: ACM Press/Addison-Wesley Publishing, 2000, pp. 157–164. doi: 10.1145/344779.344859.

[22] V. Kwatra and A. Schödl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: Image and video synthesis using graph cuts," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 277–286, Jul. 2003. [Online]. Available: http://doi.acm.org/10.1145/882262.882264

[23] V. Kwatra, D. Adalsteinsson, T. Kim, N. Kwatra, M. Carlson, and M. Lin, "Texturing fluids," *IEEE Trans. Visualizat. Comput. Graph.*, vol. 13, no. 5, pp. 939–952, Sep./Oct. 2006.

[24] A. Schödl, R. Szeliski, D. H. Salesin, and I. Essa, "Video textures," in *Proc. 27th Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH)*, New York, NY, USA: ACM Press/Addison-Wesley Publishing, 2000, pp. 489–498.
[25] Y. Li, T. Wang, and H.-Y. Shum, "Motion texture: A two-level statistical model for character motion synthesis," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 465–472, Jul. 2002. [Online]. Available: http://doi.acm.org/10.1145/566654.566604
[26] T. Ijiri, R. Mêch, T. Igarashi, and G. Miller, "An example-based procedural system for element arrangement," *Comput. Graph. Forum*, vol. 27, no. 2, pp. 429–436, 2008.
[27] L. Kavan, D. Gerszewski, A. W. Bargteil, and P.-P. Sloan, "Physics-inspired upsampling for cloth simulation in games," *ACM Trans. Graph.*, vol. 30, no. 4, pp. 93:1–93:10, Jul. 2011. [Online]. Available: http://doi.acm.org/10.1145/2010324.1964988
[28] S. J. Guy, J. van den Berg, W. Liu, R. Lau, M. C. Lin, and D. Manocha, "A statistical similarity measure for aggregate crowd dynamics," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 190:1–190:11, Nov. 2012. [Online]. Available: http://doi.acm.org/10.1145/2366145.2366209
[29] Y. Zhang *et al.*, "Data-driven synthesis of cartoon faces using different styles," *IEEE Trans. Image Process.*, vol. 26, no. 1, pp. 464–478, 2017.
[30] A. Akram, N. Wang, J. Li, and X. Gao, "A comparative study on face sketch synthesis," *IEEE Access*, vol. 6, pp. 37084–37093, 2018.
[31] L. Liu and J. Hodgins, "Learning basketball dribbling skills using trajectory optimization and deep reinforcement learning," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 142:1–142:14, Jul. 2018. [Online]. Available: http://doi.acm.org/10.1145/3197517.3201315
[32] M. Ashikhmin, "Synthesizing natural textures," in *Proc. Symp. Interact. 3D Graph. (I3D)*, 2001, pp. 217–226. [Online]. Available: http://doi.acm.org/10.1145/364338.364405
[33] V. Kwatra, I. Essa, A. Bobick, and N. Kwatra, "Texture optimization for example-based synthesis," *ACM Trans. Graph., SIGGRAPH*, vol. 24, no. 3, pp. 795–802, Aug. 2005.
[34] J. Li, C. Li, T. Yang, and Z. Lu, "A novel visual-vocabulary-translator-based cross-domain image matching," *IEEE Access*, vol. 5, pp. 23190–23203, 2017.

CHUANYAN HAO received the Ph.D. degree in soft engineering from the University of Macau, in 2015. She is currently an Assistant Professor with the Nanjing University of Posts and Telecommunications. Her main research interests include texture synthesis and analysis, image processing, and image-based animation.

YADANG CHEN received the Ph.D. degree in soft engineering from the University of Macau, in 2015. He is currently an Assistant Professor with the Nanjing University of Information Science and Technology and holds a postdoctoral position at Michigan State University. His main research interests include video segmentation, video enhancement, video editing, and augment reality.

ENHUA WU (M'87) received the B.Sc. degree from Tsinghua University, and the Ph.D. degree from the Department of Computer Science, The University of Manchester, U.K., in 1984. He has been with the State Key Laboratory of Computer Science, Institute of Software (IOS), Chinese Academy of Sciences, since 1985. He was the Director of the Research Department of Fundamental Theory and Advanced Technology, IOS, until 2001. Since 1997, he has been a Full Professor with the University of Macau, where he is currently the Associate Dean of the Faculty of Science and Technology. In recent years, he has been invited as a Chair/Co-Chair or a Keynote Speaker at various conferences, such as CGI 2010, ACM VRST 2010, CASA 2011, ACM VRCAI 2008–2011, and IWAIT 2012. He is an Editorial Board Member of TVC, CAVW, IJIG, IJVR, and IJSI and the Associate Editor-in-Chief of the *Journal of Computer Science and Technology*.