Swarm-based identification of animation key points from 2D-medialness maps

Prashant Aparajeya[1]*, Frederic Fol Leymarie[2], and
Mohammad Majid al-Rifaie[2,3]

[1] Headers Ltd, London, United Kingdon
[2] Goldsmiths, University of London, Department of Computing
[3] University of Greenwich
School of Computing & Mathematical Sciences
* Correspondance: p.aparajeya@headers-dev.com

Abstract. In this article we present the use of dispersive flies optimisation (DFO) for swarms of particles active on a medialness map – a 2D field representation of shape informed by perception studies. Optimising swarms activity permits to efficiently identify shape-based keypoints to automatically annotate movement and is capable of producing meaningful qualitative descriptions for animation applications. When taken together as a set, these keypoints represent the full body pose of a character in each processed frame. In addition, such keypoints can be used to embody the notion of the Line of Action (LoA), a well known classic technique from the Disney studios used to capture the overall pose of a character to be fleshed out. Keypoints along a medialness ridge are local peaks which are efficiently localised using DFO driven swarms. DFO is optimised in a way so that it does not need to scan every image pixel and always tend to converge at these peaks. A series of experimental trials on different animation characters in movement sequences confirms the promising performance of the optimiser over a simpler, currently-in-use brute-force approach.

Keywords: Line of action, Medialness, Dispersive flies optimisation, Swarms, Dominant points, Animation

1 Introduction

We consider the problem that faces an animator when drawing a succession of frames to generate the illusion of movement of a character (e.g. Mickey Mouse). A now classic technique, which emerged in the Disney studios in the 1930's is to indicate the main pose via a single curve and redraw and deform this curve to indicate how the character to animate shall change its main allure and position [2]. This technique, often referred to as the "Line of Action" (or LoA) is also used to draw 3D characters, but we focus in this initial work on using the LoA in conjunction with 2D profiles only (examples in Fig.1). By simply changing the LoA, e.g. curving or bending it in a different direction, the

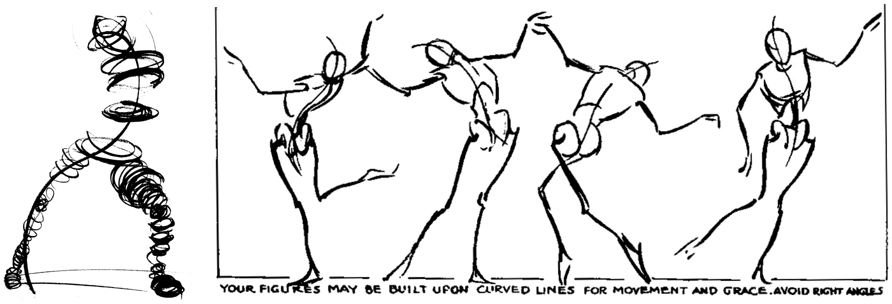
YOUR FIGURES MAY BE BUILT UPON CURVED LINES FOR MOVEMENT AND GRACE. AVOID RIGHT ANGLES

Fig. 1. Examples of the use of the Line of Action to specify the pose of a drawn character. On the left hand side is an example of a rapid sketching of 3D humanoid form using a single main 3D curve indicating the body pose (here from the bottom right foot towards the head, via the hips and torso). To the right is a series of 2D profiles of manikins drawn by Loomis [1] where the different main poses are specified via an LoA.

entire qualitative impression of the drawing can be controlled (Fig.1). Such a technique has been explored in recent years in computer graphics to provide novel tools in the context of a graphical user interface to map poses from one series of drawn characters to another [3] or to efficiently design from scratch series of poses for 2D or 3D characters [4,5]. Intuitively, there is a close relationship between such a tool used by an expert animator (on paper or via a GUI), and the human perception of shapes in a static pose or in movement. In particular, in the cognitive and vision science literature, a number of experimental results support the idea that medialness provides a powerful substrate to characterise the shape of a profile of an object (articulated or not) [6,7]. Medialness is a mapping of an image of intensity to an image of measures of medialness, i.e. how much a given locus of the 2D image is in relation with object boundaries, as a function of its minimal distance from one or more boundary segments [8]. The larger the amount of simultaneously proximal boundary fragments, the higher the measure of medialness (within a tolerance level). The highest possible medialness value being for the locus at the center of a circle, disk or oval object (refer to §2 for details).

Medialness tends to create a concentration of information along ridges centred more or less in the middle of a character as well as along the middle of its limbs. We propose in this paper a new technique to detect automatically a useful set of "hot spots" [9] where medialness is most prominent which can provide a good set of "knobs" or control points to specify an LoA, which the animator can then modify as they see fit (e.g. to change the pose, or to get inspired by an existing drawn set).

The traditional approach to identify loci of interest in medialness is to perform ridge following [10]. This represents a rather exhaustive search procedure usually followed by a thinning to identify a graph-like path in the direction of elongation of a ridge. In our work we replace this path-based approach by an efficient optimisation scheme based on swarms. We explore the application of the recently developed technique for that purpose (refer to §3). This has the advantage of "zooming-in" to loci of high medialness to get a good sampling of the main ridges without the need for an explicit (graph-like) tracing. A simple interpolation scheme through the identified hot spots provides support of an effective LoA ready for the animator to use.

2 Medialness Measurement

We define the computation of 2D medialness gauge at a point p, modifying Kovács et al. [6] original definition, by adding an orientation constraint such that only those boundary loci which are pointing inward (with respect to the figure) are considered for evaluation (Fig. 2):

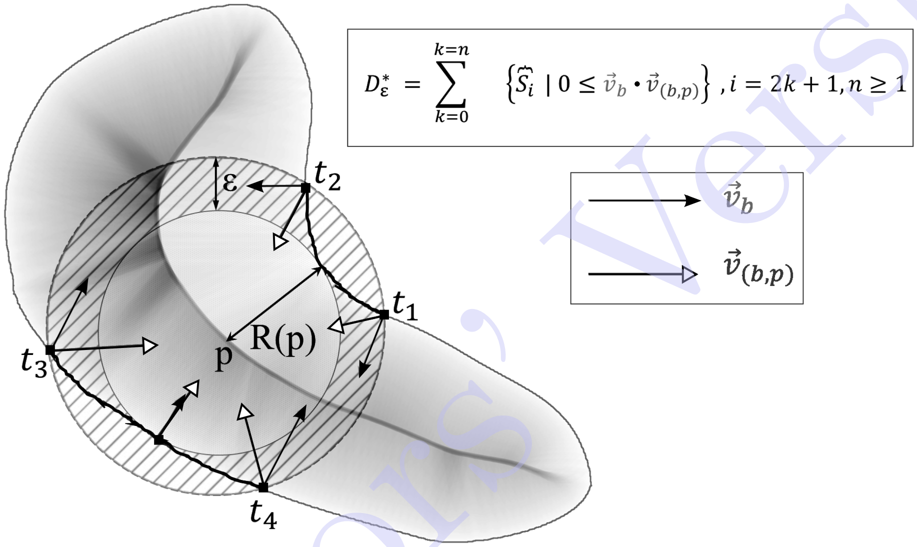
$D_{\varepsilon}^{*} = \sum_{k=0}^{k=n} \left\{ \widehat{S_i} \mid 0 \leq \vec{v}_b \bullet \vec{v}_{(b,p)} \right\}, i = 2k+1, n \geq 1$
$\vec{v}_b$
$\vec{v}_{(b,p)}$
ε
t_2
t_1
t_3
t_4
R(p)
p

Fig. 2. Illustration of the D^*_ϵ function for a simple shape, defined as an accumulation of boundary segments falling inside an annulus neighborhood of thickness ϵ (shown as darker boundary segments within the annulus' ring) centered around a position p, and such that the associated orientation vector $\overrightarrow{v}$ has a dot product with the unit radius vector (taken in the direction of the line from $b(t)$ to p) which is positive. $R(p)$ is taken as the minimum radial distance from p to the nearest contour point.

$$D^+_\epsilon = \sum_k \widehat{S_{k+1}} \cdot \delta_{v_b \cdot (p-b)} \tag{1}$$

$$D^-_\epsilon = \sum_k \widehat{S_{k+1}} \cdot \delta_{-v_b \cdot (p-b)} \tag{2}$$

where

$$\delta_x = \begin{cases} 1 \;\; \text{if x} \geq 0 \\ 0 \text{ otherwise} \end{cases} \tag{3}$$

for a point $p = (x_p,\, y_p)$, vector $b(t) = (x(t),\, y(t))$ describing (pixel) loci along a piece of 2D bounding contour (B) of the object, and such that v_b is the orientation of the boundary point $b(t)$, $\overrightarrow{v_{(b,p)}}$ is the orientation of the line joining $b(t)$ to p. The positiveness (case of internal medialness, i.e., D^+_ϵ) or negativeness (case of external medialness, i.e., D^-_ϵ) of the scalar product $v_b(p - b)$ is used to rule out boundary pixels which are oriented away from the given annulus center. We do not consider the geometry (differential continuity) of a contour other than provided by that gradient orientation.

If a boundary pixel has an orientation ϕ , its weight is the value of projection of the square-pixel in the direction of ϕ. Mathematically, this is calculated as:

$$l = \frac{1}{\max(|\sin(\phi)|, |\cos(\phi)|)} \tag{4}$$

The final contribution ∂b of a boundary point b to $\widehat{S}_i$, with orientation ϕ and angular distance θ , is calculated as:

$$\partial b = \begin{cases} l \cos\theta, \text{ if } \; -\pi/2 \leq \theta \leq \pi/2 \\ \;\; 0, \qquad\qquad \text{otherwise} \end{cases} \tag{5}$$

The metric $R(p)$, the minimum radial distance to the interior annular shell, is taken as the smallest available distance between p and a bounding contour element:

$$R(p) = \underset{t}{min} \left\{ |p - b(t)| \;\mid\; 0 \leq \overrightarrow{v_b} \cdot \overrightarrow{v_{(b,p)}} \;\right\} \tag{6}$$

The medialness measure of a point p varies with the two parameters: $R(p)$ and ϵ, where $R(p)$ is the minimum radial distance between p and the bounding contour, and ϵ is the width of the annulus region (capturing object trace or boundary information). Any boundary point b falling inside this annulus that satisfies the definition of equation 1 (for interior medialness) or equation 2 (for exterior medialness) is added in support for medialness at p. ϵ is selected as a logarithmic function of $R(p)$:

$$\epsilon_p = \kappa \log_b \left(\frac{R(p)}{\kappa} + 1 \right),\; b > 1 \tag{7}$$

Having selected ϵ to be adaptive as a logarithmic function of $R(p)$, the next step is to define a useful logarithmic *base*. We use the same logarithmic base of $e/2$, where e is the Euler's number (approximately 2.718281828), as we report in [11,12].

3 Dispersive flies optimisation

Dispersive flies optimisation (DFO) – first introduced in [13] – is an algorithm inspired by the swarming behaviour of flies hovering over food sources. The swarming behaviour of flies is determined by several factors including the presence of threat which disturbs their convergence on the marker (or the optimum value). Therefore, having considered the formation of the swarms over the marker, the breaking or weakening of the swarms is also noted in the proposed algorithm.

In other words, the swarming behaviour of the flies, in DFO consists of two tightly connected mechanisms, one is the formation of the swarms and the other is its breaking or weakening[4].

Being a population-based continuous optimiser, this algorithm bears similarities with other swarm intelligence algorithms. However, what distinguishes DFO from other population-based optimisers is its sole reliance on agents' position vectors at time t to generate the position vectors for time $t+1$. Furthermore, other than population size, the algorithm uses a single tunable parameter, Δ, which adjusts the diversity of the population. Therefore, the simplicity of the update equation – due to its lone reliance on the position vector – and having a single parameter to adjust (other than population size) as well as its competitiveness in contrast with other algorithms (as reported later in this section), provides the motivation to use the algorithm for this work.

The algorithm and the mathematical formulation of the update equations are introduced next.

The position vectors of the population are defined as:

$$\vec{x}_i^{\,t} = \left[x_{i1}^t, x_{i2}^t, ..., x_{iD}^t\right], \qquad i = 1, 2, ..., N \tag{8}$$

where t is the current time step, D is the dimension of the problem space and N is the number of flies (population size). For continuous problems, $x_{id} \in \mathbb{R}$, and in the discrete cases, $x_{id} \in \mathbb{Z}$ (or a subset of $\mathbb{Z}$).

In the first iteration, $t = 0$, the i^{th} vector's d^{th} component is initialised as:

$$x_{id}^0 = x_{\min,d} + \mathrm{U}(0,1)\left(x_{\max,d} - x_{\min,d}\right) \tag{9}$$

where $\mathrm{U}(0,1)$ is the uniform distribution between 0 and 1; $x_{\min}$ and $x_{\max}$ are the lower and upper initialisation bounds of the d^{th} dimension, respectively. Therefore, a population of flies are randomly initialised with a position for each flies in the search space.

On each iteration, the components of the position vectors are independently updated, taking into account the component's value, the corresponding value of

[4] Regarding weakening of the population formation, several elements play a role in *disturbing* the swarms of flies; for instance, the presence of a threat causes the swarms to disperse, leaving their current marker; they return to the marker immediately after the threat is over. However, during this period if they discover another marker which matches their criteria closer, they adopt the new marker. Another contributing factor to disturbance is the wind speed, which is suggested to influence the position of the swarm [14].

the best neighbouring fly with the best fitness (considering ring topology), and the value of the best fly in the whole swarm:

$$x_{id}^{t} = x_{i_n d}^{t-1} + \mathrm{U}\left(0,1\right) \times (x_{sd}^{t-1} - x_{id}^{t-1}) \tag{10}$$

where $x_{i_n d}^{t-1}$ is the position value of $\vec{x}_i^{\,t-1}$'s best *n*eighbouring fly in the d^{th} dimension at time step $t-1$, and x_{sd}^{t-1} is the value of the *s*warm's best fly in the d^{th} dimension at time step $t-1$. The update equation is illustrated in Fig. 3.

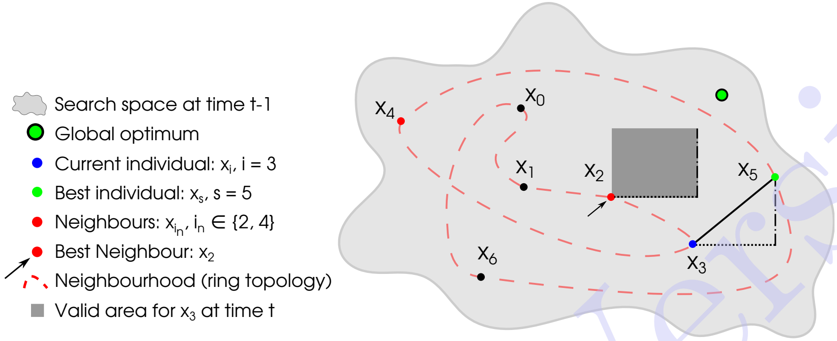
Search space at time t-1
Global optimum
Current individual: x_i, i = 3
Best individual: x_s, s = 5
Neighbours: x_{i_n}, $i_n \in \{2, 4\}$
Best Neighbour: x_2
Neighbourhood (ring topology)
Valid area for x_3 at time t
X_4
X_0
X_1
X_2
X_5
X_3
X_6

Fig. 3. Sample update of x_i, where $i = 3$ in a 2D space.

The algorithm is characterised by two main components: a dynamic rule for updating flies position (assisted by a social neighbouring network that informs this update), and communication of the results of the best found fly to other flies.

As stated earlier, the swarm is disturbed for various reasons; one of the impacts of such disturbances is the displacement of flies which may lead to discovering better positions. To consider this eventuality, an element of stochasticity is introduced to the update process. Based on this, individual components of flies' position vectors are reset if a random number, r, generated from a uniform distribution on the unit interval $U\left(0,1\right)$ is less than the *disturbance threshold* or Δ. This guarantees a disturbance to the otherwise permanent stagnation over a likely local minima. Algorithm 1 summarises the DFO algorithm. In this algorithm, each fly is assumed to have two neighbours (i.e. ring topology).

In summary, DFO is a numerical optimiser over continuous or discretised search spaces. DFO is a population based algorithm, originally proposed to search for an optimum value in the feasible solution space. The algorithm's simplicity – in addition to its update equation's bare-bones reliance on only position vectors – comes from having only *one* tunable parameter, Δ, controlling the component-wise dispersion in its population.

As reported in [13], despite the algorithm's simplicity, it is shown that DFO outperforms the standard versions of the well-known Particle Swarm Optimisa-

Algorithm 1 Dispersive flies optimisation

```
procedure DFO($N, D, X_{\min}, X_{\max}, f$)        ▷ INPUT: swarm size, dimensions, lower/upper bounds, fitness func
    for $i = 0 \to N-1$ do                          ▷ Initialisation
        for $d = 0 \to D-1$ do
            $x_{id}^{0} \leftarrow \mathrm{U}(x_{\min,d}, x_{\max,d})$
        end for
    end for
    while ! termination criteria do                 ▷ Main DFO loop
        for $i = 0 \to N-1$ do
            $\vec{x}_i$.fitness $\leftarrow f(\vec{x}_i)$
        end for
        $\vec{x}_s \ = \text{arg min}\ [f(\vec{x}_i)], \quad i \in \{0, \dots, N-1\}$
        for $i = 0 \to N-1$ and $i \neq s$ do
            $\vec{x}_{i_n} = \text{arg min}\ [f(\vec{x}_{(i-1)\%N}), f(\vec{x}_{(i+1)\%N})]$
            for $d = 0 \to D-1$ do
                if $\mathrm{U}(0,1) < \Delta$ then
                    $x_{id}^{t+1} \leftarrow \mathrm{U}(x_{\min,d}, x_{\max,d})$
                else
                    $x_{id}^{t+1} \leftarrow x_{i_n d}^{t} + u(x_{sd}^{t} - x_{id}^{t})$          ▷ Update equation
                    if $x_{id}^{t+1} < x_{\min,d}$ or $x_{id}^{t+1} > x_{\max,d}$ then      ▷ Out of bounds
                        $x_{id}^{t+1} \leftarrow \mathrm{U}(x_{\min,d}, x_{\max,d})$
                    end if
                end if
            end for
        end for
    end while
    return $\vec{x}_s$
end procedure
```

tion [15], Genetic Algorithm (GA) [16] as well as Differential Evolution (DE) [17] on an extended set of benchmarks over three performance measures of error, efficiency and reliability. It was shown that DFO is more efficient in 85% and more reliable in 90% of the 28 standard optimisation benchmarks used; furthermore, when there exists a statistically significant difference, DFO converges to better solutions in 71% of problem set. Further analysis was also conducted to explore the diversity of the algorithm throughout the optimisation process, a measure that potentially provide more understanding on algorithm's ability to escape local minima. In addition to studies on the algorithm's optimisation performance, DFO has recently been applied to medical imaging, image analysis, machine learning and data mining, deep learning, simulation and gaming, computational aesthetic measurements and (digital) arts.

4 Experiments and results

In our set of experiments, we have used different frames of animated characters – as static poses, poses along with the LoA drawn, and frames from an animated (in movement) sequence. Our target is to identify key-points providing a sampling of a LoA, in a numerically efficient way. The LoA is itself an abstract concept (existing in the mind of the professional and trained human animators), and at this stage in our research we only target a recovery of key-points which can be interpolated to provide useful if only approximate LoA traces. A future stage of our research will require a verification experiment of suggested LoA traces by a group of professionals familiar with the use of the LoA. However, we take advantage of the fact that, by definition a medialness map helps to localise features which are approximately (always) near the center lines of an object's boundary, ensuring that our approximate recovery is at least a potential LoA trace. In terms of numerical efficiency, a continuum pixel-wise scanning (brute-force approach) is time consuming when video frames are of high-resolution. Therefore, we aim to compare DFO with the brute-force search to depict how DFO is robust and time efficient in finding these key-points.

Fitness Function

For a given image frame, we first extract boundary points and apply the medialness function (defined in equation 1) to generate a medialness map. The medialness values at different positions in this map are taken as the fitness value for the DFO search process. In other words, medialness serves as the fitness function for DFO. A peak finding process runs in parallel and assigns as "best flies" those which are near or at a local medialness peak; other flies, not near local peaks, can also be initiated (randomly) but with higher fitness values. In following iterations, a best fly can only be replaced if another fly finds a nearby medialness peak with higher fitness value, while other flies roam around (refer to algorithm 1).

In our experimental set-up, we annotate each character pose with desired key-points location as (x,y) co-ordinates. This annotation is used to design the stopping condition for DFO. The DFO search stops if all desired key-points are found. We categorise our experiments into three parts: (a) standing human pose, (b) static character, and (c) characters in movement (sequence). For each of these, we run 100 DFO trails to find best, worst, and average swarming results. We use 30 flies for swarming where the maximum iteration of convergence is set to 100. Flies are randomly re-dispersed if either all flies have converged to a peak or they exceed the maximum allowed number of iterations. These outcomes are then compared and analysed with the brute-force approach.

Experiment 1: Standing Human Pose

The first test case is a human silhouette (300×520 pixels) in standing pose containing 18 dominant points, where the key-points resembles to markers found

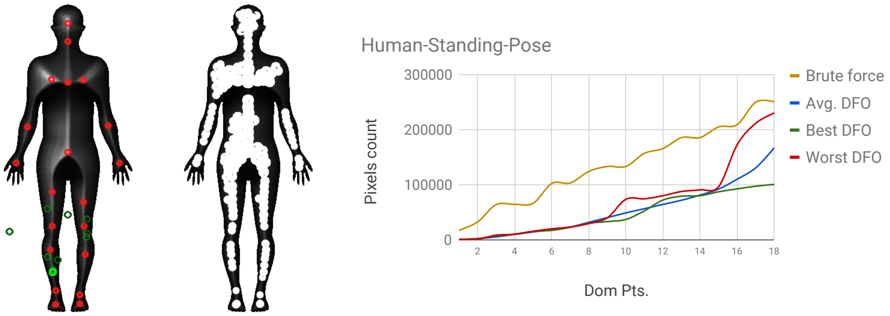

Fig. 4. *Left*: Medialness map of the standing human is represented via grey levels, where high values correspond to brighter shades. Red circles indicate peaks found by the DFO; bright green-circle represents the position of a best-fly, while remaining green circles represent other fly positions. *Middle*: Removal of medialness values in the circular area of radius ϵ (see equation 7) for any visited medialness locus. *Right*: graph indicating the number of dominant points found versus the total number of image pixels visited. The yellow curve indicates brute-force approach, while the Green, Red and Blue curves correspond to the results for the Best, Worst and Average DFO search respectively.

in MoCAP suits [18,19]. The medialness map (figure 4 (left)) is generated using equation 1 and a copy of this map is created to track the flies' converging positions (figure 4 (right)). Whenever flies converge at a local peak, a circular area of radius ϵ is removed from the tracking map, which then avoids flies to converge again at the same position. Graph in figure 4 (right) shows the performace of DFO w.r.to brute-force approach.

DFO converges in a few iterations when the local peaks have high values, but it takes respectively longer time for lower-valued or isolated peaks. From figure 4, we can observe that local medialness peaks near head, chest, and abdomen have substantially high medialness values and are surrounded by decreasing medialness field with larger spread area. This allows DFO flies to climb and converge quickly at the peak. We also observe that the DFO flies are less prone to go towards hands and legs area, since the local peaks have lower medialness values with smaller medialness spread. However, smaller peaks are found either after sufficient removal of high valued areas from the tracking map or if flies go in those area because of randomness (in re-dispersal of flies).

Experiment 2: Static Characters Posing

To create a believable (intention of) movement, deciding on the pose of a character is a critical task. The basic principles of to identify a (good) pose are based on the selection of the LoA, a corresponding silhouette, and a sense of weight and balance for the character. Among these overall features, the LoA plays a critical role. By simply changing the LoA making it more curved, sloped or arched in a different direction the entire essence of the drawing can be changed.

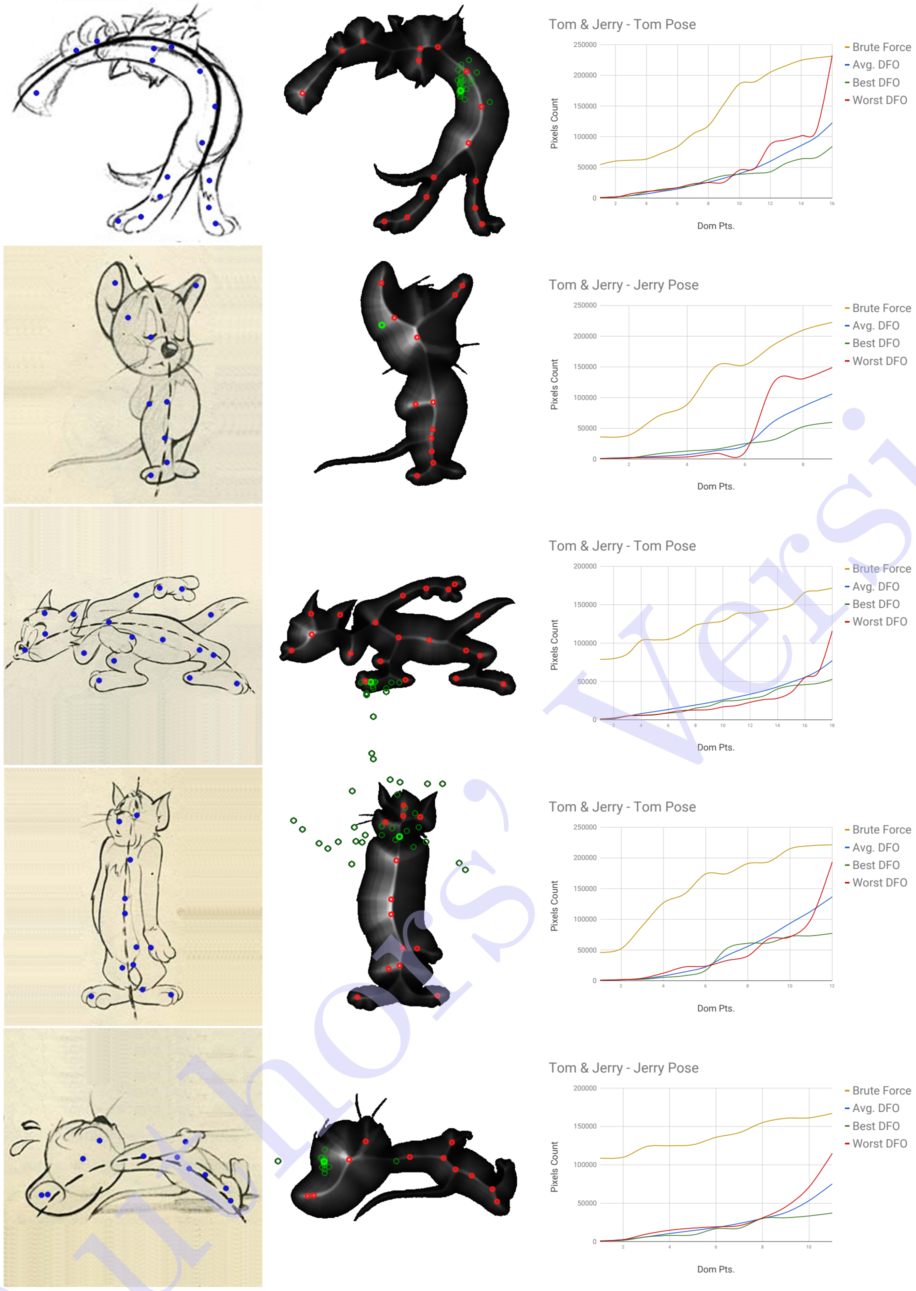

Fig. 5. Tom and Jerry cartoon characters. On the left hand side are different poses of Tom & Jerry drawn on LoA registered with annotated key-points as blue dots. In the middle, medialness maps are represented as gray regions, red circles are peaks found by DFO, bright green circle is the position of best fly, and rest of green circle represent other flies' positions. To the right are time comparison graphs – total number of peaks found versus total number of pixels visited, where Yellow curve indicates brute-force approach, while Green, Red and Blue curves indicate best, worst and average DFO search respectively.

Our proposed shape representation via medial nodes can be seen as a potential psychologically motivated support for LoA [11,7].

In this experiment set, we aim to identify such medial nodes on the famous "Tom and Jerry" cartoon characters by using DFO. The left part of figure 5 is the image frames with original LoAs and registered annotated key-points. It can be seen more clearly how some of these key-points are will fitted along the hand-drawn LoA. In the middle of this figure, a snapshot of DFO processing is shown, where red dots indicated peaks found by DFO and green ones are the swarming flies. Alike the experiment 1, flies converges quickly near the high medilaness peak with larger surroundings while lower peaks take slightly longer time. Another interesting find in the graph shown in figure 5 is that more than 60% of dominant points takes almost similar number of pixel scanning in 100 trials, i.e., the best, worst and average DFO processing is similar in first 60% cases. The graph also indicates that average DFO processing outperforms the brute-force approach.

Experiment 3: Characters in Movement

In this experiment, we have created and annotated set (classes) of characters' frames in movements. Due to the nature of moment cycle, inter class variability exists in both number of frames and desired number of dominant points. A moment cycle consists of the sequence of several body poses where each body pose is different from each other and can have a different number of visible key-points. Example of this can be seen in figures 6 (first two rows – "an angry cat" in running poses) & 8 (first two rows – "a normal cat" in running poses). We keep the image resolution the same (500×500) for each class. Snapshots of DFO behaviour are shown in figures 6 (bottom two rows) and 8 (bottom two rows).

Similar to our other two experiments, the graph in figure 7 shows that more than 75% of keypoints are found by DFO by visiting less than 20-25% of pixels. DFO flies are mostly attracted towards higher and larger medialness areas and the peaks in those areas such as head and abdomen are found more quickly. The tail part of "angry-cat" is much thicker than for a "normal-cat", hence flies converges more quickly.

Our experiments also show that image padding affects the execution time for brute-force approach as it's behaviour is linear. While DFO gets unaffected by such change and produces same search result in similar execution time.

5 Conclusion

Our initial premise was to consider the Line of Action (LoA) for character pose drawing and how the character's pose gets changed by changing the LoA. Our goal is then to design an efficient algorithmic chain to systematically retrieve a useful sampling of "hot spots" in medialness, a generalisation of classic distance maps, Voronoi diagrams and other symmetry-based graph representation. Once

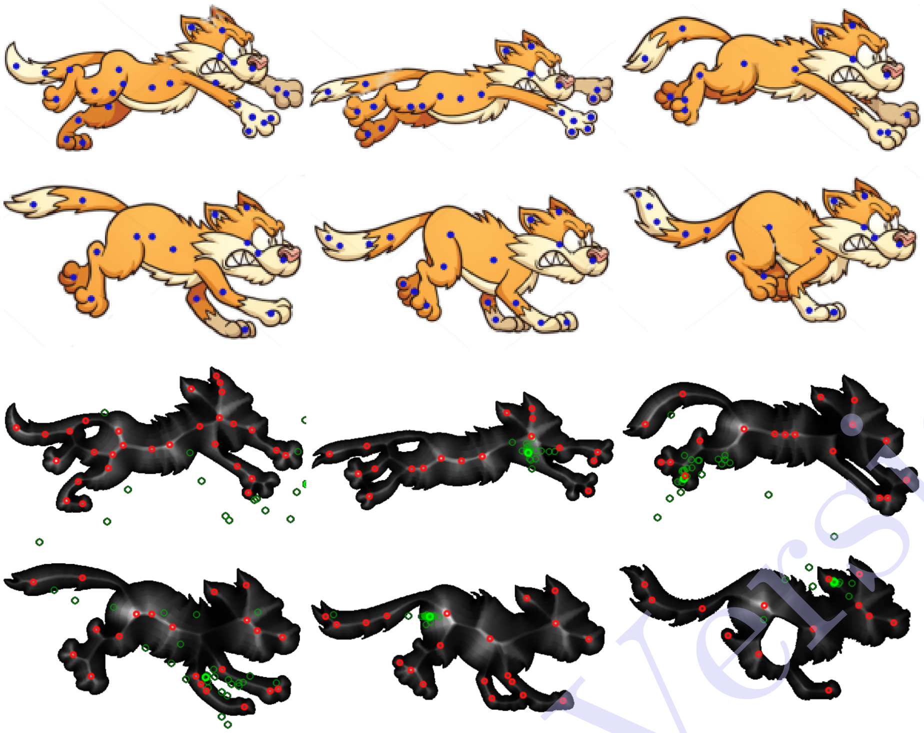

Fig. 6. First two rows: An animated angry cat running poses. The desired and annotated dominant points are indicated as blue dots. For proper visibility they these single point dots are enlarged into circles. Bottom two rows: Medilaness maps are indicated as gray regions, red circles are peaks found by DFO, bright green circle is the position of best fly, and rest of green circle represent other flies' positions.

a medialness map is produced for existing character boundaries, a brute force approach can be used to seek peaks in magnitude of medialness. For high resolution images or when processing hundreds of frames, a more efficient scheme proves valuable in practice. We have proposed here an efficient search based on Dispersive Flies Optimisation (DFO) which performs well in our targeted scenarios and automatically finds key-points along the LoA quickly and robustly.

Our experimental results indicate that DFO converges much earlier than the currently-in-use brute-force approach and further identifies all dominant points (in labelled test images). Given DFO's GPU-friendly nature, this approach could be further improved for larger, high-resolution input images. For frames where annotation is not available — the usual practical case — a good experimental stopping condition is to limit the swarm search space to up to 70% of the image size. However, in order to find the first 10 best keypoints, our empirical investigation (according to graphs in figure 9) suggests that less than 20% of the image size is largely sufficient.

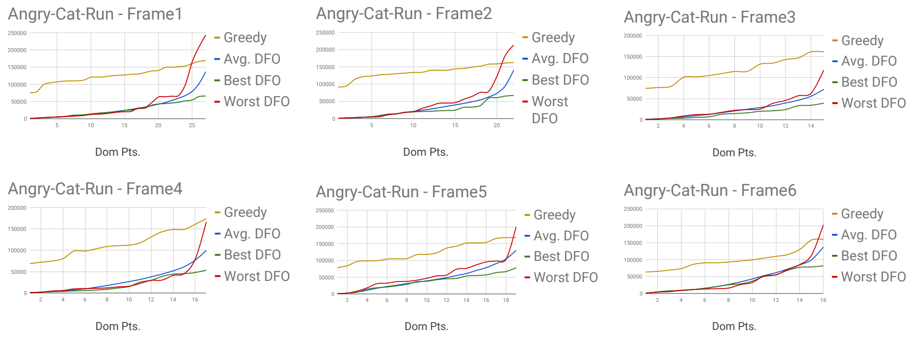

Fig. 7. Time comparison graph, number of peaks found versus total number of image pixel traversed, for frames indicated in Figure 6 (top rows). Yellow curve indicates brute-force approach, while Green, Red and Blue curves indicate best, worst and average DFO search respectively.

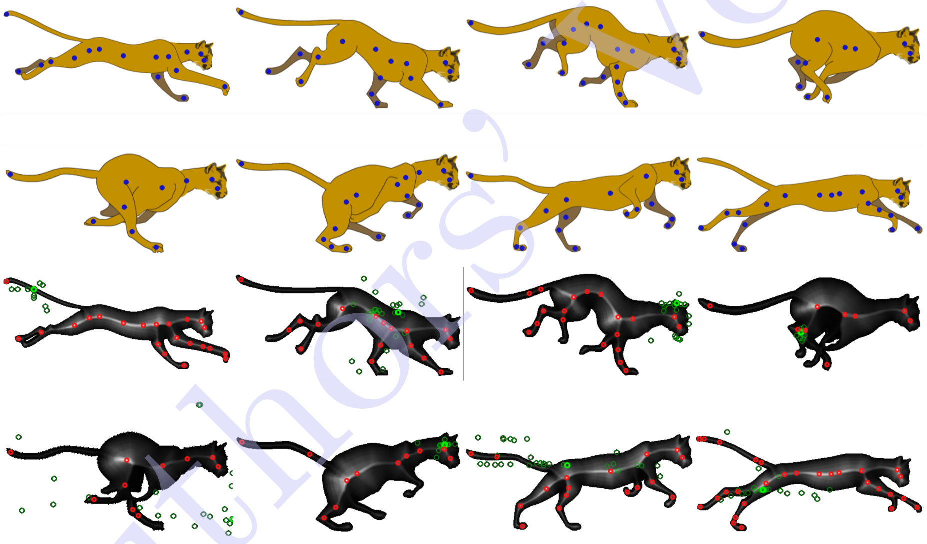

Fig. 8. Animated frames of a running cat. The a priori annotated dominant points are indicated as blue dots. For proper visibility they these single points are enlarged into circles. Bottom two rows: Medialness maps are indicated as gray regions, red circles are peaks found by DFO, while a bright green circle is the position of a current best fly, and other of green circles represent other current fly positions.

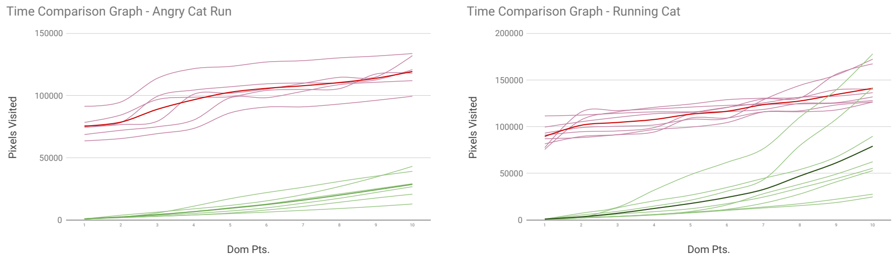

Fig. 9. Time comparison graph to find first 10 dominant points. Light red curves are for brute-force approach for different character frames, while the dark-thick red one indicates the average time execution for the whole movement sequence. Light green curves are for average DFO processing for different frames, while the dark-thick green one is the overall average DFO curve for the full movement sequence. Left graph is for the angry-cat in running pose (see figure 6), and right one is for the normal cat in running pose (see figure 8)

References

1. Loomis, A.: Successful Drawing. Viking Books (1951)
2. Leymarie, F.F., Aparajeya, P.: Medialness and the perception of visual art. Art & Perception 5(2), 169–232 (2017)
3. Bregler, C., Loeb, L., Chuang, E., Deshpande, H.: Turning to the masters: Motion capturing cartoons. ACM Transactions on Graphics 21(3), 399–407 (2002)
4. Guay, M., Cani, M.P., Ronfard, R.: The Line of Action: An intuitive interface for expressive character posing. ACM Transactions on Graphics 32(6), Article no. 205 (2013)
5. Guay, M., Ronfard, R., Gleicher, M., Cani, M.P.: Adding dynamics to sketch-based character animations. In: Proceedings of the Eurographics/ACM Symposium: Expressive Graphics — Sketch-Based Interfaces and Modeling. Istanbul, Turkey (June 2015)
6. Kovács, I., Fehér, Á., Julesz, B.: Medial-point description of shape: a representation for action coding and its psychophysical correlates. Vision research 38(15), 2323–2333 (1998)
7. Leymarie, F.F., Aparajeya, P., MacGillivray, C.: Point-based medialness for movement computing. In: Workshop on Movement and Computing (MOCO). pp. 31–36. ACM, IRCAM, Paris (2014)
8. Burbeck, C.A., Pizer, S.M.: Object representation by cores: Identifying and representing primitive spatial regions. Vision Research 35(13), 1917 – 1930 (1995)
9. Kovács, I.: "Hot spots" and dynamic coordination in Gestalt perception. In: Von der Malsburg, C.V., Phillips, W.A., Singer, W. (eds.) Dynamic Coordination in the Brain: From Neurons to Mind, chap. 14, pp. 215–28. Strüngmann Forum Reports, MIT Press (2010)
10. Eberly, D., Gardner, R., Morse, B., Pizer, S., Scharlach, C.: Ridges for image analysis. Journal of Mathematical Imaging and Vision 4(4), 353–373 (1994)
11. Aparajeya, P., Leymarie, F.F.: Point-based medialness for 2d shape description and identification. Multimedia Tools and Applications 75(3), 1667–1699 (2016)

12. Aparajeya, P.: Medialness-based Shape Invariant Feature Transformation. Ph.D. thesis, Goldsmiths, University of London (2016)
13. al-Rifaie, M.M.: Dispersive flies optimisation. In: M. Ganzha, L. Maciaszek, M.P. (ed.) Proceedings of the 2014 Federated Conference on Computer Science and Information Systems. Annals of Computer Science and Information Systems, vol. 2, pp. pages 529–538. IEEE (2014), `http://dx.doi.org/10.15439/2014F142`
14. Downes, J.: The swarming and mating flight of diptera. Annual review of entomology 14(1), 271–298 (1969)
15. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks. vol. IV, pp. 1942–1948. IEEE Service Center, Piscataway, NJ (1995)
16. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA (1989)
17. Storn, R., Price, K.: Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces (1995), tR-95-012, [online]. Available: http://www.icsi.berkeley.edu/ storn/litera.html
18. Ceseracciu, E., Sawacha, Z., Cobelli, C.: Comparison of markerless and marker-based motion capture technologies through simultaneous data collection during gait: proof of concept. PloS one 9(3) (2014)
19. Aparajeya, P., Petresin, V., Leymarie, F.F., Rueger, S.: Movement description and gesture recognition for live media arts. In: 12th European Conference on Visual Media Production. pp. 19–20. ACM (2015)