Image and Vision Computing 60 (2017) 142–153

Contents lists available at ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

Cell tracking using deep neural networks with multi-task learning[☆],[☆☆]

Tao He, Hua Mao[*], Jixiang Guo, Zhang Yi

Machine Intelligence Laboratory, College of Computer Science, Sichuan University, Chengdu 610065, People's Republic of China

ARTICLE INFO

Article history:
Received 3 May 2016
Received in revised form 31 July 2016
Accepted 16 November 2016
Available online 24 November 2016

Keywords:
Cell tracking
Deep learning
Convolutional Neural Networks
Multi-task learning

ABSTRACT

Cell tracking plays crucial role in biomedical and computer vision areas. As cells generally have frequent deformation activities and small sizes in microscope image, tracking the non-rigid and non-significant cells is quite difficult in practice. Traditional visual tracking methods have good performances on tracking rigid and significant visual objects, however, they are not suitable for cell tracking problem. In this paper, a novel cell tracking method is proposed by using Convolutional Neural Networks (CNNs) as well as multi-task learning (MTL) techniques. The CNNs learn robust cell features and MTL improves the generalization performance of the tracking. The proposed cell tracking method consists of a particle filter motion model, a multi-task learning observation model, and an optimized model update strategy. In the training procedure, the cell tracking is divided into an online tracking task and an accompanying classification task using the MTL technique. The observation model is trained by building a CNN to learn robust cell features. The tracking procedure is started by assigning the cell position in the first frame of a microscope image sequence. Then, the particle filter model is applied to produce a set of candidate bounding boxes in the subsequent frames. The trained observation model provides the confidence probabilities corresponding to all of the candidates and selects the candidate with the highest probability as the final prediction. Finally, an optimized model update strategy is proposed to enable the multi-task observation model for the variation of the tracked cell over the entire tracking procedure. The performance and robustness of the proposed method are analyzed by comparing with other commonly-used methods. Experimental results demonstrate that the proposed method has good performance to the cell tracking problem.

1. Introduction

Visual tracking is an important research topic in the field of computer vision [1,2], which aims to track the trajectories of single or multiple objects and is widely applied in many practical visual tasks such as video surveillance, automatic driving systems, and biological living cell pedigree analysis. As a typical visual tracking application, cell tracking [3] aims at tracking cells directly from microscopic images. By the results of cell tracking, we can investigate cell behavior to further construct cell lineage and analyze cell morphology [4,5]. Cell tracking methods, deployed on a large number of cells, are helpful to facilitate feasible conclusions about cell populations [6]. The inspection of living cells allows researchers to obtain the correlation between many diseases and abnormal cell behavior [7]. Thus, cell tracking with an automatic method is essential.

Challenges of cell tracking are summarized into four categories. The first challenge is cell deformation, e.g., elongation, expansion, and shrinkage [5]. Traditional visual tracking methods handle rigid bodies without significant shapes changes [8]. However, cells are non-rigid bodies and tracking them is more challenging because they always change shapes with time, which are explained in Fig. 1. The second category of challenges is about cell behavior. For instance, cell migration entails complex motion with multiple modes. The complicated cell behavior increases the difficulty of cell tracking. The third challenge comes from the living environment of the cell. There are many particles in the cytochylema, which contains dead cells, germs, and other organic material. Cell tracking methods must distinguish cells from other particles [9] mentioned above. The final challenge is that the cell images are captured at a low resolution and the cell is non-significant in the image because of its small size.

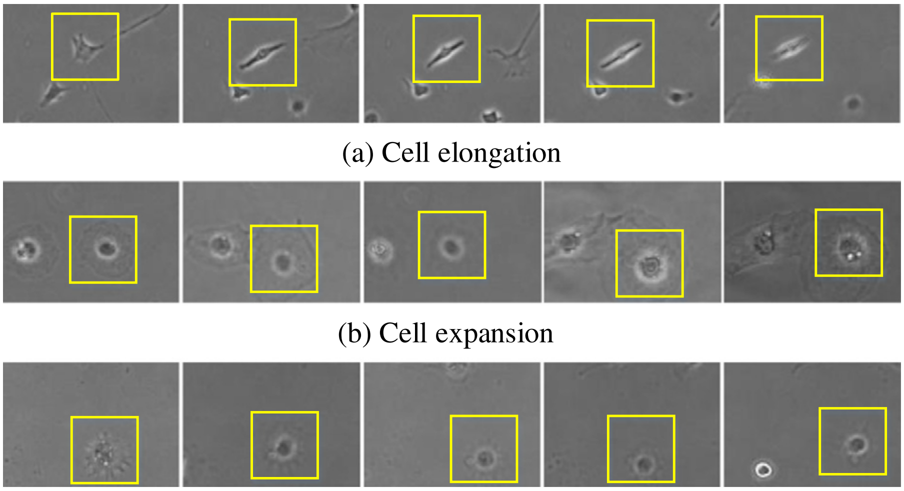
With the development of deep learning, feature learning methods [10] have been successfully applied to computer vision area [11]. In our work, Convolutional Neural Networks (CNNs) [8] are utilized to learn robust features of cells in the cell tracking because the

[☆] This paper has been recommended for acceptance by Xi Peng, Ph.D.

[☆☆] This work was supported by the National Science Foundation of China [grant numbers 61432012, 61402306]; Science & Technology Department of Science Province [grant number 2015GZ0119].

[*] Corresponding author.
E-mail addresses: taohe@stu.scu.edu.cn (T. He), huamao@scu.edu.cn (H. Mao), guojixiang@scu.edu.cn (J. Guo), zhangyi@scu.edu.cn (Z. Yi).
URL: http://www.machineilab.org (Z. Yi).

http://dx.doi.org/10.1016/j.imavis.2016.11.010
0262-8856/© 2016 Elsevier B.V. All rights reserved.



(a) Cell elongation

(b) Cell expansion

(c) Cell shrink

Fig. 1. Common types of cell deformation.

robust cell features can benefit the tracking of the non-rigid and non-significant cells. CNNs are widely used to visual tasks [12]. In traditional image classification problems, CNNs are used to achieve increased accuracy on large-scale datasets [13] and have become the state-of-the-art feature learning models among many machine learning methods. Recently, CNNs have also been applied to visual tracking problems [14]. However, to the best of our knowledge, it is the first time to develop the CNN method for the cell tracking problem.

In real-world visual problems, deep learning [12] methods sometimes encounter the overfitting problem [15]. Since the cell objects are non-rigid and non-significant, the cell tracking methods are required to have good generalization performance. Multi-task learning (MTL) [16] usually decomposes large problems into independent tasks that are learned separately and combined later. It is a machine learning method that learns a shared representation from all independent tasks. This learning procedure provides the MTL a good generalization performance [16]. It has been successfully applied to address challenges in many domains including machine translation [17], speech recognition [18], and object recognition [19].

Based on CNNs and MTL techniques, we further propose a cell tracking method to address the aforementioned challenges in cell tracking problem by learning robust features of non-rigid and non-significant cell objects. The proposed method contains a particle filter motion model, a MTL observation model, and an optimized model update strategy. We start the tracking procedure by being given with the cell position in the first frame at a microscope image sequence. Then, the particle filter motion model produces a set of candidate bounding boxes in the following frames. Finally, the MTL observation model chooses the best candidate as the final target. CNNs act as the part of MTL observation model while the optimized model update strategy updates the observation model to improve the tracking performance.

The contributions of this paper can be summarized as follows:

- We propose a multi-task observation model to solve the cell tracking problem. The model consists of an offline cell behavior recognition stage and an online cell tracking stage. The offline and online stages use conjunct CNNs to extract features and softmax classifiers to solve individual tasks. The training of CNNs in the offline stage acts as the pre-training of the online learning task.
- We design an optimized model update strategy to mitigate the impact of cell deformation and cell migration in the online tracking stage. This strategy selects the positive samples by managing a confidence queue and a queue update rule.
- We manually construct a cell tracking dataset for single-cell tracking. All of the cell image sequence samples are labeled with ground truth annotations, forming an important public resource for visual tracking researchers and biologists.

The remainder of this paper is structured as follows. The related cell tracking methods are introduced in Section 2. Section 3 provides a literature review of the techniques related to the proposed method. The details of our proposed method are outlined in Section 4. In Section 5, the proposed method is evaluated in our cell tracking benchmark dataset. Section 6 draws conclusions about our work.

2. Related works

Many cell tracking methods have been proposed in recent decades. These works typically utilize different cell tracking methods according to variable cell types and tracking requirements. Generally, cell tracking methods can be divided into three categories: *level sets based approaches* [20], *data association methods* [21], and *tracking-by-detection methods* [22].

Level sets based approaches track living cells by exploiting the intensity characteristics and shape of the cells. An energy function is defined to model the gradient magnitude along the cell boundary and the spatial overlap of the detected cells. The tracking procedure is performed by minimizing the energy function. Ref. [20] demonstrated a solution to the energy function to search the image-level lines of boundaries of connected components within the level sets by threshold decomposition. The cell tracking is implemented by using intensity coherency and spatial information based on the detected cells. Ref. [23] integrated the level sets method with the model evolution approach for cell tracking in time-lapse fluorescence microscopy. Each level-set function represents one object (cell or nucleus) and the evolution equation for each level-set function is derived by replacing the original weights from the energy function.

Data association methods define probabilistic objective functions to associate cells between two or more coherent frames for cell tracking. The most typical method is nearest neighbor association, which associates each cell from frame to frame within a threshold range. Ref. [24] proposed a graph theory-based minimum cost flow method to solve the cell tracking problem by detecting defined graph edges. Global spatio-temporal data association methods are another type of advanced data association tracking method [25]. In Ref. [26], a tree-structure global spatio-temporal data association method is proposed to obtain cell trajectories and lineage trees. This method addressed the cell tracking problem by solving the maximum-a-posteriori problem with a linear programming method. The more generalized data association approach in Ref. [21] was proposed for cell tracking by using the combination of information containing cell position, dynamics, and morphology.

Although level sets based and data association methods are widely used and have been verified in many cell tracking applications, some challenges remain. First, these methods are typically non-universal solutions applied to specific practical data and many parameters are manually tuned. Second, some methods design a set of skills to select features for later tracking and thereby require substantial computational time.

In recent years, tracking-by-detection methods have been widely used for visual tracking [1,22,27]. These methods aim to solve cell tracking problems by feature learning [28] and using a classifier to recognize the object from the background. The tracking-by-detection methods view tracking as a real-time processing procedure with some online tracking techniques. Given the location of the target object in the first frame of a video, the algorithm tracks the object from frame-to-frame and simultaneously automatically updates the tracking model and rules. Ref. [29] proposed a structured learning method using optimum parameters from a given dataset to learn a large number of features. This method intended to exploit the structure and dependency arising from the assumption that cells are associated. In Ref. [30], the spatio-temporal information was fully learned to automatically track cells. Ref. [22] proposed an optimized multiple-instance learning method to achieve greater robustness with fewer parameters. A discriminative classifier was trained by using a multiple instance learning algorithm to separate the object from other particles. In another tracking method published in Ref. [8], a stacked sparse autoencoder was pre-trained to learn more meaningful features and the online classification problem was performed by optimizing a squared-error cost function. A subtler online learning method was proposed in Ref. [27] to understand and diagnose the visual tracking problem. Our proposed cell tracking method is also a tracking-by-detection method.

3. Preliminaries

In this section, we review the details of multi-task learning (MTL) and Convolutional Neural Networks (CNNs).

3.1. Convolutional Neural Networks

CNNs typically involve two layers: the convolution and pooling layers. The function of the convolution layer is to apply many convolutional filters over the input volumes. The aim of the pooling layer, which is a type of down-sampling, is to reduce the spatial size of the representation and the number of parameters in the network.

Suppose that the CNN model has L layers. $\mathbf{k}^l$ denotes the parameters of the kernel and b^l denotes the bias parameters at l-th layer. The input and output of l-th convolution layer are defined by $\mathbf{a}^{l-1}$ and $\mathbf{a}^l$, respectively. The input of the network is defined as image $\mathbf{x}$. The calculation begins by initializing the activations with $\mathbf{a}^{\mathbf{0}} = \mathbf{x}$. We have the following formulation of convolution layers

$$a_{ij}^{(l)} = f\left(\sum_{p=0}^{P-1}\sum_{q=0}^{Q-1} k_{pq}^{(l)}\cdot a_{(i+p)(j+q)}^{(l-1)} + b^{(l)}\right), \qquad (1)$$

where P and Q denote the size of the kernel, $f(\cdot)$ denotes the activation function, and $l \in (1, L)$. a_{ij}^{l} denotes the feature of i-th row and j-th column in the l-th layer. The feature maps of layer $l-1$ are convolved with the learnable kernel k and the non-linear function $f(\cdot)$ is used to form the output feature map. The computation of the pooling layers is formulated as follows

$$\mathbf{a}^{(l)} = f\left(\beta^{(l)}\cdot down\left(\mathbf{a}^{(l-1)}\right) + b^{(l-1)}\right), \qquad (2)$$

where $down(\cdot)$ represents a sub-sampling function and β is the multiplicative bias. Typically, this function sums over each distinct n-by-n block in the input image such that the output image is n-times smaller along spatial dimensions. The outputs of the convolution layers are usually used as the input of the pooling layers.

3.2. Multitask learning

The basic idea of MTL is to share parameters between related tasks. A simple MTL model consists of a neural network with shared hidden units among tasks. Let T denote the number of tasks and $\mathcal{N}_T$ indicate the set of tasks. For each task $t \in \mathcal{N}_T$, the trainset is given by m examples $\left\{\left(x_t^{(1)}, y_t^{(1)}\right), \ldots, \left(x_t^{(m)}, y_t^{(m)}\right)\right\} \in \mathcal{R}^x \times \mathcal{R}^y$, where $\mathcal{R}$ is the set of real numbers. The aim of MTL is to learn a conjunct function $g_t : \mathcal{R}^x \rightarrow \mathcal{R}^y$. The computation of MTL is achieved by the formulations of the CNNs in this paper, such that the function g_t is represented by

$$g_t(x) = f\left(\theta_t \cdot f_{CNN}(x_t)\right), \qquad t \in \mathcal{N}_T, \qquad (3)$$

where, $f_{CNN}(\cdot)$ denotes the formulations of the CNN, θ denotes the parameters of the public activations of the CNN in the t-th task. Assume that the cost function is the squared error objective, the related tasks are learned by minimizing the following formulation

$$\arg\min_{\theta_t} \sum_{j}^{m}\sum_{t}^{\mathcal{N}_T}\left(\left\|y_t^{(j)} - g_t\left(x^{(j)}\right)\right\|^2 + \frac{\lambda}{2}\|\theta_t\|^2\right), \qquad (4)$$

where, λ is the weight decay parameter and $\|\cdot\|^2$ denotes the 2-norm. The MTL shares parameters in the public CNN structure between the related tasks and learns separate tasks with different parameters θ_t. The common information can be shared to learn joint features among the related tasks resulting in better generalization performance.

4. The proposed cell tracking method

In Ref. [27], visual tracking is separated into five individual models including the motion model, feature extractor, observation model, model updater, and ensemble post-processor. The motion model produces a number of candidate bounding boxes on the object appearing in the current frame. The feature extractor further describes each candidate generated during the motion model stage. The observation model determines the most likely target candidate. The model updater provides a series of strategies to update the observation model. Finally, the ensemble post-processor addresses the situation of multiple trackers.

Compared with Ref. [27], the proposed method entails three parts including the motion model, MTL model, and model update. The CNN is a component of the MTL model and learns features directly from the raw image frames, thus, we do not include a feature extractor in our method. An ensemble post-processor is also unnecessary because we use a single MTL-based tracker in our method.

Motion model. Widely used motion models include the particle filter, sliding window, and radius sliding window [27]. Among them, the particle filter is a common sequential Bayesian estimation method that predicts the target recursively. So the particle filter is adopted as the motion model in this paper. More details on the particle filter can be found in Ref. [31].

Multitask learning model. The MTL model breaks cell tracking into two separate tasks. The main task is online cell tracking and the assistant task is cell classification. The classification task classifies each candidate generated by the motion model into three categories: *no-cell*, *active cell*, and *general cell*. The *no-cell* category includes particles, dead cells, and other organic material in the cell living environment. The *active cell* category indicates cells that will fissure in the next frames. All types of dividing cells are included in this category. The *general cell* category includes cells in other situations. This is a straightforward classification task related to cell behavior and is designed to help the main tracking task to learn more meaningful features.

The main tracking task aims to find the position of the target in the current frame by choosing the highest confidence candidate from a set of candidates selected in the motion model part. The main task can be simplified as a binary classifier. The cell to be tracked is specified by its bounding box in the first frame. The positive samples are captured from the images close to the bounding box of the current cell within a one- or two-pixel bias. We add some Gaussian noise to positive samples to improve the robustness and denoising ability of the model [32]. We collect some negative examples from the background at a short distance from the object. Based on the CNN and the softmax classifier, the main task is combined with the assistant task as depicted in Fig. 2. The model contains three parts including the cell input, feature extractor (CNN), and softmax classifier.

Model update. As the MTL model is trained, the single cell is tracked from frame-to-frame. A challenge is posed by the loss of the target resulting from cell behavior. Thus, the MTL model must be updated according to some rules. The model update is an important part of the visual tracking method [27], moreover, it is a significant part of the cell tracking problem because cell deformation occurs frequently, but there have been relatively few studies of model update methods in online tracking methods. Ref. [33] uses a method based on entropy minimization to identify a reliable model update approach, forming a rather complicated model update strategy. The most commonly used method is to set a confidence threshold to judge whether to update the observation model. In this paper, we use a sample strategy to improve the performance of the model update based on confidence thresholds. We manage a queue of positive samples to update the MTL model. The notation used in this section is summarized in Table 1.

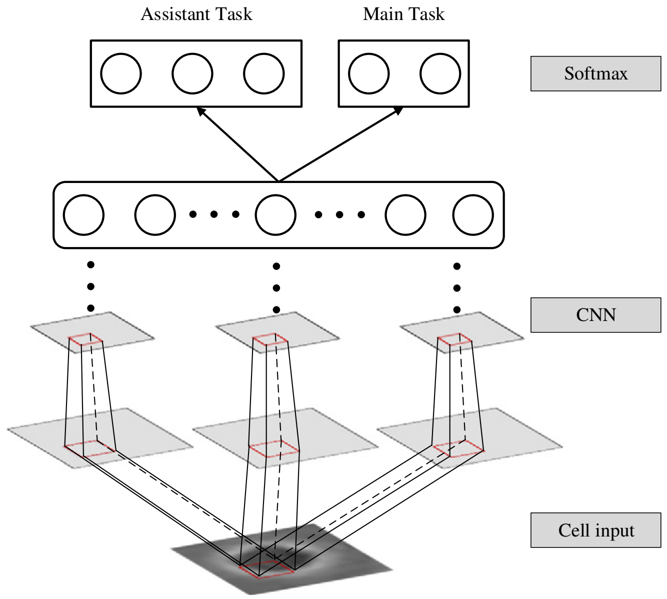


Fig. 2. The architecture of multitask observation model is divided into three parts including cell input, feature extractor, and softmax classifier.

4.1. Task definition

Given two datasets $S_{off} = \{(a_1, b_1), (a_2, b_2), \ldots, (a_i, b_i)\}$ and $S_{on} = \{(x_1, y_1),\ (x_2, y_2),\ \ldots,\ (x_j,\ y_j)\}$, where S_{off} denotes the dataset for the offline assistant task and S_{on} indicates the dataset for the online main task, a_i, b_i denote the input and output of i-th sample in S_{off}, and x_j, y_j denote the input and output of j-th sample in S_{on}. The learning task can be summarized as matching two functions:

$$f : \mathbf{a} \to \mathbf{b}, \qquad g : \mathbf{x} \to \mathbf{y}, \tag{5}$$

where f and g denote the functions of the assistant task and the main task, respectively.

A set of candidates $C = \{c_1, c_2, \ldots, c_k\}$ are selected by the particle filter motion model. The aim of tracking is to determine the most probable target t from C, which is achieved by:

$$t = \max\left\{g(c_1),\ g(c_2), \ldots, g(c_k)\right\}. \tag{6}$$

To incorporate the online learning techniques into the MTL system, we break the main task and assistant task into continued

Table 1
Notations in the proposed cell tracking method.

Symbol	Description
S_{off}, S_{on}	S_{off} denotes the dataset for the offline assistant task and S_{on} indicates the dataset given for the online main task.
(a_i, b_i)	(a_i, b_i) is the i-th sample in the dataset for the offline task and $(a_i, b_i) \in S_{off}$.
(x_j, y_j)	(x_j, y_j) is the j-th sample in the dataset for online task and $(x_j, y_j) \in S_{on}$.
f, g	f is the function that denotes the offline learning task and g is the function that indicates the online learning task.
C, c_k, t	C is the set of candidates produced in motion model part and c_k denotes k-th candidate, where $c_k \in C$. t denotes the target selected by the online softmax classifier with best probability.
$\mathbf{w}, h, \mathbf{z}$	$\mathbf{w}$ denotes the parameters of the CNNs, h is the abstract function that indicates the CNNs, and $\mathbf{z}$ denotes the output of the CNNs.
θ, η	θ and η are the parameters of the softmax layer at assistant and main task, respectively.
D, T, S, m	D and T are the categories of the assistant and main task, respectively. S is the number of neurons, indicating the dimensionality of $\mathbf{z}$. m is the number of samples.
p, λ	p denotes the probability distribution and λ denotes the parameters of weight decay.
$\mathbf{z}^o, \mathbf{w}^o$	$\mathbf{w}^o$ and $\mathbf{z}^o$ are the parameters and outputs, respectively, of the CNNs in the online main task.
Q, J	Q is the threshold to determine whether to update the observation model. J denotes the cost function.
α_i, β_i, N	α_i and β_i denote the i-th positive sample and confidence probability, respectively. N is the number of positive samples.
G, μ, σ	G denotes the normal distribution, μ is its mean, and σ is its standard deviation.
τ, ϕ	τ is the constant to tune the noise and ϕ is the constant used to adjust the frequency of the model update.

training procedures. The CNN is trained offline in the assistant task, and then the model is trained and updated in the main tracking task. The remainder of this section describes the details of the offline learning task, the online tracking task, and the optimized model update strategy.

4.2. Offline assistant learning task

The learning of the offline assistant task serves as the pre-training of the online tracking task. The network is trained using the S_{off} dataset. To formulate the network, we simplify the CNN parameter set as $\mathbf{w}$ to construct the CNN structure to compute the output $\mathbf{z} = h_{\mathbf{w}}(\mathbf{a})$. Here, $\mathbf{z}$ is the output of the CNN part and h denotes the function mapping from input $\mathbf{a}$ to output $\mathbf{z}$. The output $\mathbf{z}$ of the CNN is the input into the softmax classifier. The parameters of the softmax are defined as $\boldsymbol{\theta}$. The training target would be calculated by minimizing the following cost function:

$$J_{(\boldsymbol{\theta})} = -\left(\sum_{i=1}^{m}\sum_{d=1}^{D} 1\{b_i = d\}\cdot p(d|\boldsymbol{\theta};\mathbf{z}_i)\right) + \frac{\lambda}{2}\sum_{i=1}^{D}\sum_{j=1}^{S}\theta_{ij}^2, \tag{7}$$

where m denotes the data scale of S_{off}, λ denotes the weight decay parameters, D denotes the categories of the target, and S is the number of neurons indicating the dimensionality of $\mathbf{z}$. $1\{\cdot\}$ is a binary function that $1\{true\} = 1$, and $1\{false\} = 0$. The weight decay term $\frac{\lambda}{2}\sum_i^D\sum_j^S\theta_{ij}^2$ penalizes large values of the parameters. $p(d|\boldsymbol{\theta};\mathbf{z}_i)$ is the probability that the target corresponds to the d-th label when the input is $\mathbf{z}_i$ and is formulated as:

$$p(d|\boldsymbol{\theta};\mathbf{z}_i) = \log\frac{\exp^{\theta_d^{\mathrm{T}}\mathbf{z}_i}}{\sum_{j=1}^{D}\exp^{\theta_j^{\mathrm{T}}\mathbf{z}_i}}. \tag{8}$$

When the assistant task has been learned, the network is trained with a set of convergent parameters $\mathbf{w}$ and $\boldsymbol{\theta}$. The learning of the main task begins with initializing the parameters of the CNN by using $\mathbf{w}$, and then fine-tuning the network using data from S_{on}.

4.3. Online main learning task

The learning of the main task is similar to the learning of the assistant task. The parameters of the CNN are redefined as $\mathbf{w}^o$ and the outputs of the CNN are redefined as $\mathbf{z}^o$ which are calculated as $\mathbf{z}^o = h_{\mathbf{w}^o}(\mathbf{x})$. The parameters of the softmax in the main task are defined as $\boldsymbol{\eta}$. The learning of the main task is achieved by minimizing the function:

$$J_{(\boldsymbol{\eta})} = -\left(\sum_{i=1}^{n}\sum_{t=1}^{T} 1\{y_i = t\}\cdot p\left(t|\boldsymbol{\eta};\mathbf{z}_i^o\right)\right) + \frac{\lambda}{2}\sum_{i=1}^{T}\sum_{j=1}^{S}\eta_{ij}^2, \tag{9}$$

$$p(t|\boldsymbol{\eta};\mathbf{z}_i) = log\frac{\exp^{\eta_t^{\mathrm{T}}\mathbf{z}_i}}{\sum_{j=1}^{T}\exp^{\eta_j^{\mathrm{T}}\mathbf{z}_i}}, \tag{10}$$

where n denotes the number of samples in S_{on} and T denotes the categories of the target. The meanings of the remaining variables correspond to those in the assistant task. Given a threshold Q, when the maximum prediction is recorded and the statement $t > Q$ is computed as *false*, the network is updated by resampling data from the current image frames.

4.4. An optimized model update strategy

Because cell deformation occurs in the real time stream, the MTL model degrades over time. A simple method to address this is to manage a positive sample queue, which maintains a trade-off between optimizing the observation model and preventing the tracker from drifting to the background [8,27]. However, this can capture noisy examples when the prediction t is lower. Instead, we can set a threshold to determine whether to adopt the current prediction. Unfortunately, this would lead to another challenge whereby the observation model is not updated when the confidence of the predictions remains lower than the specific threshold.

Considering these challenges, an optimized strategy to manage the positive sample queue is designed to judge whether to adopt the current prediction. A positive sample set is defined by $\{\alpha_1,\ldots,\alpha_i,\ldots,\alpha_N\}$, where α_i is the i-th positive sample and N denotes the number of positive samples. For each positive sample, $\{\beta_1,\ldots,\beta_i,\ldots,\beta_N\}$ denote the corresponding confidences. In the first frame, the target, which is defined as α_0, is manually labeled. The positive samples are initialized by

$$\alpha_1 = \alpha_0, \tag{11}$$

$$\alpha_i = \alpha_0 + G\left(\mu,\sigma_i^2\right), \tag{12}$$

where $i \geq 2$, $\sigma_i = \tau\cdot(i-1)$, and τ is a constant. G denotes the normal distribution, where μ is its mean and σ is its standard deviation. These equations initialize the positive samples by adding Gaussian noise. Gaussian noise is typically added to improve the generalization ability of unsupervised learning [32]. In this scheme, the constants μ and τ are set to relatively low values to maintain the precision for the positive samples. In this paper, we set $\mu = 0$ and $\tau = 0.0001$. The simple linear correlation $\sigma_i = \tau\cdot(i-1)$ determines that the positive samples are noisier as the subscript i increases. The corresponding confidences are initialized to $\beta_i = 1, i \in (1, N)$ and are updated by the following rule from frame-to-frame.

$$\beta_i := \beta_i\cdot\frac{(N-i+1)}{N-i+1+\phi}, \tag{13}$$

where ϕ is a constant used to adjust the frequency of model updates. The rule decays the confidences of positive samples with time. The new predicted cell target is added to the queue when its confidence conforms to $t > \beta_{min}$, where β_{min} is the minimum confidence of positive samples. The suitable cell image is added to the end of the queue with the corresponding confidence setting $\beta_1 = t$. The proposed strategy balances the frequency of model updates and the selection of improved positive samples to optimize the proposed MTL model. The pipeline of the proposed strategy is illustrated in Fig. 3. According to the example and the formulaic explanation, the advantages of this model update strategy could be summarized as:

- Image sample with lower confidence does not enter the positive sample queue.
- The least suitable samples in the queue are the first to be removed by selecting the sample with the minimum confidence.
- With time, the earlier samples in the queue are removed more frequently because their confidences decay faster.

The pipeline of the overall method is illustrated in Fig. 4. The MTL algorithm is descripted in Algorithm 1.

Algorithm 1. The algorithm of proposed cell tracking method.

```
Input Training datasets $S_{on}, S_{off}$.
Output The candidate image with largest prediction $t_{max}$.
Initialization $\mathbf{w} \leftarrow w_0, \eta \leftarrow \eta_0, \theta \leftarrow \theta_0, Q \leftarrow Q_0$
Step one. Offline assistant task learning.
Maximize the cost function of Equation 7 to optimize $\mathbf{w}$ and $\theta$ .
Step two. Online main task learning.
Initialize the parameters $\eta$ and $w^o$ by maximizing the cost function in Equation 9.
Step three. Track the target online and update the observation model.
for $i$-th image frame in cell sequence, $i > 2$ do
    Particle Filter. Produce the candidate set $C$.
    Calculate the probability of each candidate $c_j$, $j \in (1, k)$, select the maximum probability $t$.
    Draw the bounding box in the image.
    Update the positive samples in $S_{on}$ by the proposed strategy.
    if $t \leq Q$ then
        Update the model.
    end if
end for
```

5. Experiments

5.1. Experiment setup

The source data consists of image sequences captured by a microscope at 5-minute intervals. The size of each image is 320 × 240 pixels. The image is in the standard RGB channel but has a lower resolution. The sizes of the cells range from about 200 to 1000 pixels.

We build a manually-annotated cell tracking benchmark. The dataset consists of 80 image sequences and the positions of cells in each image are labeled with a rectangular bounding box. Each sequence of the dataset has an average 29 images, the maximum number of images in a sequence is 66, and the minimum number is 7 images. Compared with existing benchmark datasets, our dataset entails shorter sequences as a result of cell division and cell apoptosis. Our dataset forms a more challenging visual tracking problem because the cells are non-rigid and non-significant and have an unstable living environment.

We further build a three-category cell image classification dataset by manually capturing patches from the source data image. The size of each cell is resized to 32 × 32 pixels in this dataset. There are 30,000 samples in the dataset. Techniques including adding Gaussian noise, rotating, and slightly shifting, are utilized to increase the size of the dataset, as used in Ref. [32]. These labeled images correspond to the samples in S_{off}.

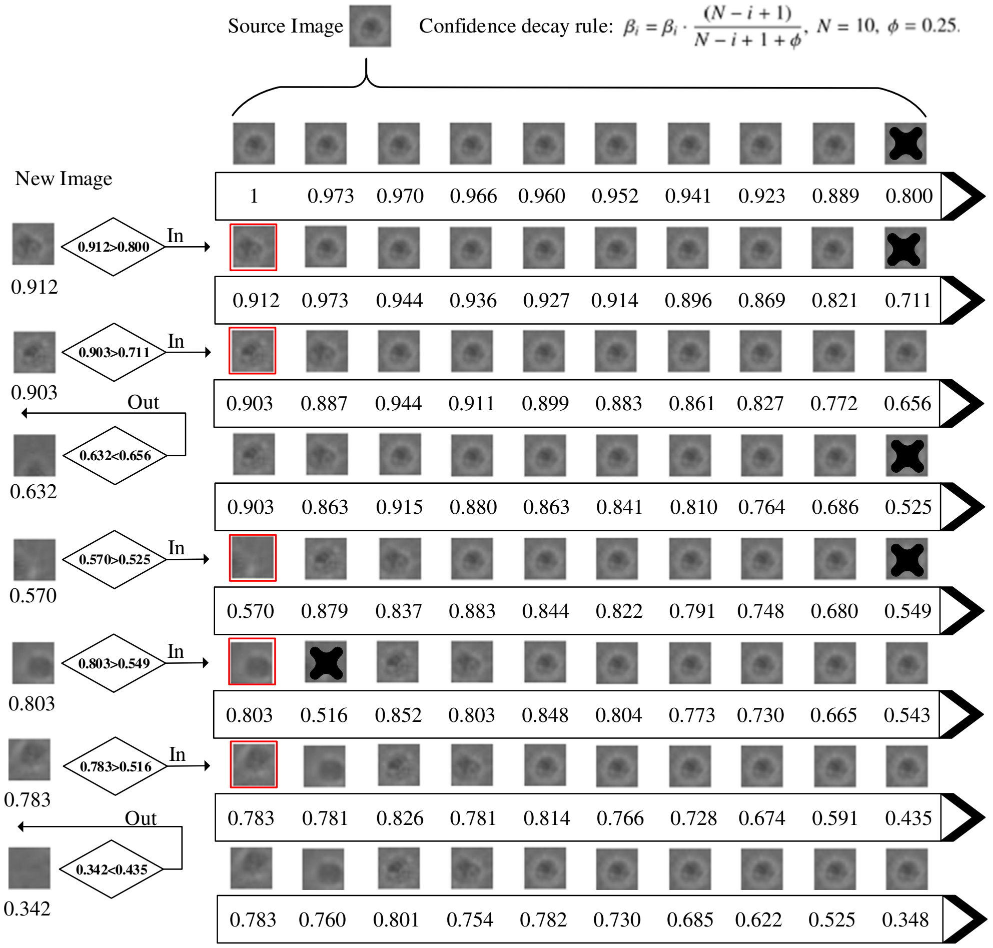


Fig. 3. An example illustrating the management of the positive sample queue. The queue has 10 samples with the corresponding confidence recorded below. The cell images labeled by "×" are removed from the queue and the cell images labeled by the red rectangle are the new samples into the queue. The rule that determines whether a current image should enter into the queue depends on the judgment statement in the diamond.

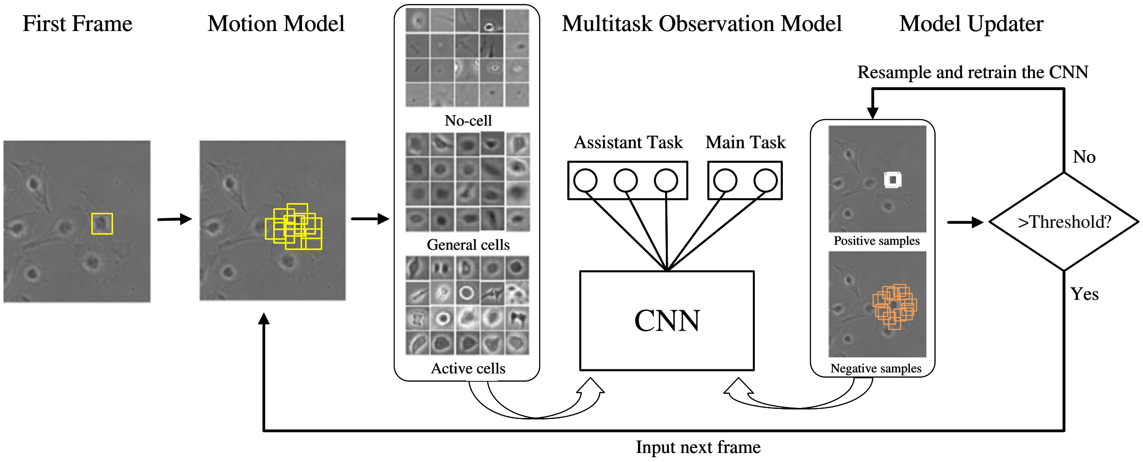


Fig. 4. Pipeline of the proposed cell tracking method including the motion model, multitask observation model, and model update.

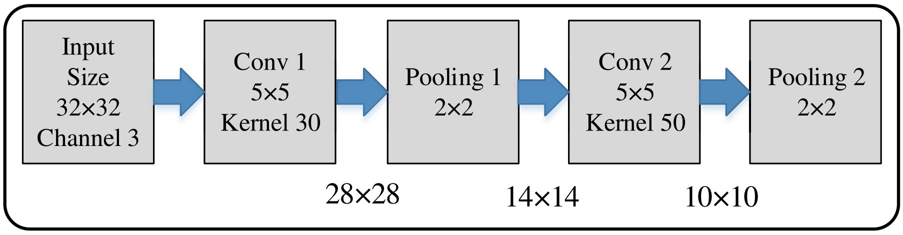


Fig. 5. The structure of CNN model.

As the image size is small (32 × 32), building more network layers cannot improve the recognition results but take more computation when designing the CNN model. For balancing the run time and recognition performance, the CNN model is designed to own four layers in this paper. It has two convolution layers and two pooling layers which is illustrated in Fig. 5. In the offline assistant learning stage, the learning rate is 0.01, the batch size is 1000 and the weight decay parameter is 0.0001. The model is trained with 20 epoches. In the online cell tracking stage, the training epoch size is reset to 5 for decreasing the run time of online tracking.

The proposed method is implemented in MATLAB and achieves an average frame rate of 0.9 fps (frames per second) on a i7 3.4 GHz dual core PC. In our cell tracking problem, the images are captured by the microscope every 5 min. Thus, the proposed method is sufficient for the cell tracking.

In recent years, visual tracking evaluation methods typically adopt the evaluation methodology in Ref. [1]. The main evaluation metrics include *precision plot* and *success plot*. The *precision plot* evaluates the performance of visual tracking by calculating the average Euclidean distance between the center locations of the

Table 2
The average center location errors of samples 1–20. The best results are highlighted in bold face and the second results are underlined.

Seq.	Ours	MTT	L1APG	ASLA	CSK	IVT	LOT	SCM	STRUCK	DLT
1	5.02	33.13	28.78	23.56	26.15	40.84	17.15	18.08	**4.26**	7.51
2	**0.47**	3.87	3.14	4.59	4.04	11.72	7.92	3.35	1.61	0.53
3	**1.75**	2.48	3.33	19.92	36.06	36.62	30.41	3.51	3.80	8.86
4	3.70	38.72	37.04	54.08	49.70	35.35	21.98	50.14	37.08	**1.99**
5	4.64	23.24	27.15	49.58	57.82	53.61	63.32	5.70	**2.40**	7.50
6	1.38	2.65	2.25	3.41	4.09	22.91	18.28	3.05	5.62	**0.89**
7	**1.40**	7.76	3.05	2.80	2.15	25.30	12.20	2.35	48.86	1.53
8	**2.23**	4.32	19.57	35.00	20.42	20.54	13.70	3.00	3.19	4.51
9	**6.40**	34.08	26.14	36.25	39.55	29.32	36.91	29.86	7.35	7.52
10	**1.82**	14.72	13.14	17.45	12.72	28.79	31.63	2.83	2.67	1.86
11	**1.54**	2.25	4.03	4.56	10.17	42.86	17.32	6.12	4.58	1.78
12	3.16	4.21	4.01	2.89	4.10	21.63	4.83	**2.26**	2.28	3.50
13	**1.62**	4.06	1.66	1.91	4.67	44.74	9.11	2.00	2.94	1.74
14	**1.08**	29.57	28.70	40.22	4.78	34.43	21.57	3.31	40.81	1.26
15	26.58	49.19	58.62	78.68	73.55	71.57	73.08	51.85	34.29	**24.47**
16	3.54	5.49	**2.84**	24.15	22.88	53.44	65.94	3.73	5.93	5.60
17	5.77	32.43	30.36	31.09	33.64	31.09	32.29	29.27	8.57	**4.20**
18	**4.05**	21.05	54.36	35.48	178.05	56.29	9.25	6.22	4.22	4.56
19	**1.34**	10.74	11.10	31.40	10.78	29.77	33.80	11.39	1.97	1.90
20	**0.84**	27.30	30.87	25.68	25.67	29.53	32.60	12.57	3.73	3.43
Average	**3.92**	17.56	19.51	26.14	31.05	36.02	27.66	12.53	11.31	4.76

Table 3
Baseline descriptions in model performance evaluation.

Symbol	Description
LR	A simple logistic regression with l_2 regularization.
RR	Least squares regression with l_2 regularization.
SVM	The standard SVM with hinge loss and l_2 regularization.
CNN + Soft	A general CNN and a softmax classifier are simply jointed.
init-MLM	MTL model without the proposed model update strategy.
MLM	MTL model with the proposed model update strategy.

tracked targets and the ground truth. The *success plot* uses the bounding box overlap between the tracked targets and ground truths. More details can be found in Ref. [1].

5.2. Performance evaluation

To evaluate the performance of the proposed cell tracking method, we set up a set of baselines as described in Table 3. All of the baseline tracking methods use the particle filter motion model, raw grayscale features, and a simple model update procedure. Among the methods, LR, RR, and SVM correspond to the basic observation models in tracking-by-detection methods [27]. MLM is the MTL observation model with the proposed model update strategy. The remaining baselines represent the online tracking method with some components of our model removed.

The listed methods are test in the 80 cell tracking dataset. The *precision plot* and *success plot* are calculated and the results are averaged at (1–20), (21–40), (41–60) and (61–80) samples. The experimental results are depicted in Figs. 6 and 7.

From the experimental results, it is obvious that the CNN based model is better than the traditional observation models, although the CNN + Soft is not always good according to the average precision plots in 41–60 samples and the average success plots in 21–40 samples. The MTL model outperforms other methods according to the results of init-MLM and MLM except the average success plots in 1–20 samples. Among the baseline methods, the proposed MTL model with the proposed model update strategy achieves the best results specially referring to the average precision plots in the 1–20 and 21–40 samples.

5.3. Comparison with other methods

Although the MTL model with proposed model update strategy has the best performance as an observation model compared with the baseline methods, it is also needed to be compared with other state-of-the-art visual tracking methods. Some general used visual tracking methods are tested in our provided dataset. These methods are summarized in Refs. [1,8]. In order to give the detail results of tracking performance, the location errors calculated by central-pixel distance between predicted target and true cell object are test. The results of 1–20 samples are list in Table 2. Our MTL model based method has shown best or second performance except the results of sequence 12. Our method has best performance according to the average results of 1–20 samples.

Among these methods, the Multi-Task Tracking (MTT) is another MTL based visual tracking method [34]. The MTT is based on the research about particle filter. The particles are modeled as linear combinations of dictionary templates and learning the joint sparse representation of each particle is considered a single task in MTT. The MTT would be viewed as the motion model level

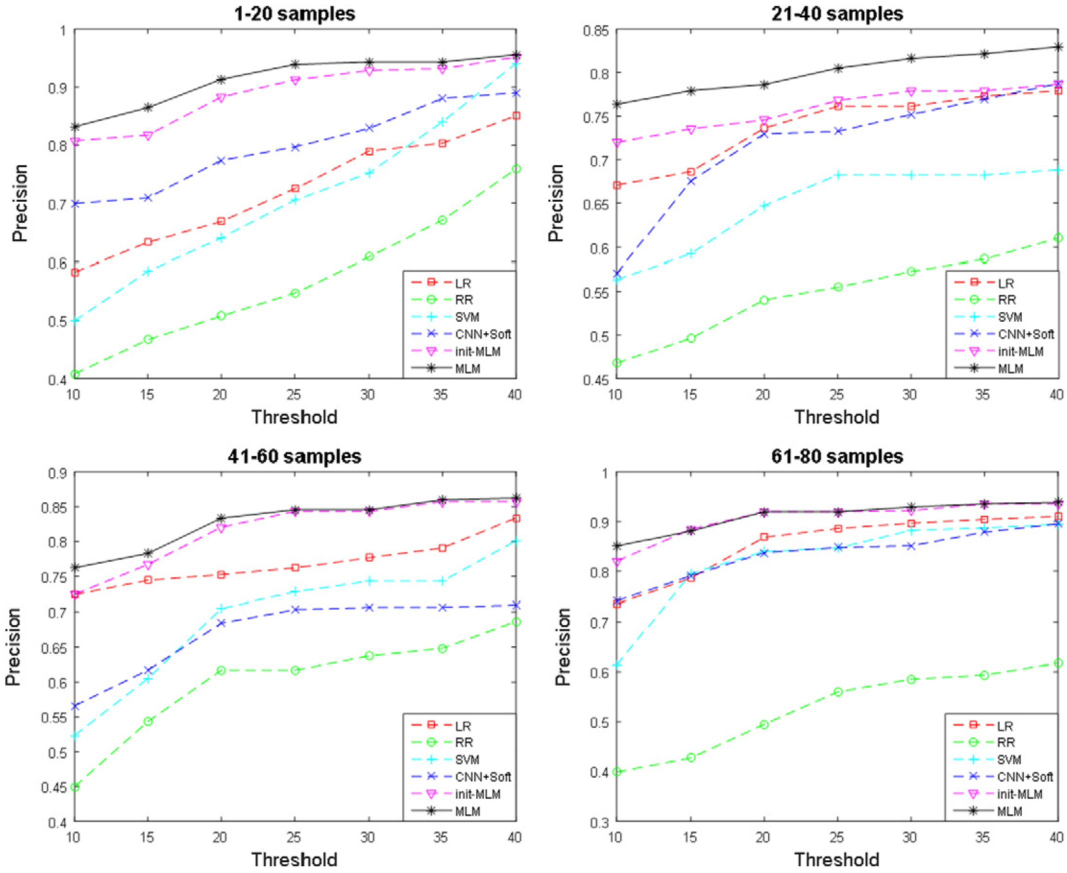


Fig. 6. The average precision plots of samples 1–20, 21–40, 41–60, and 61–80.

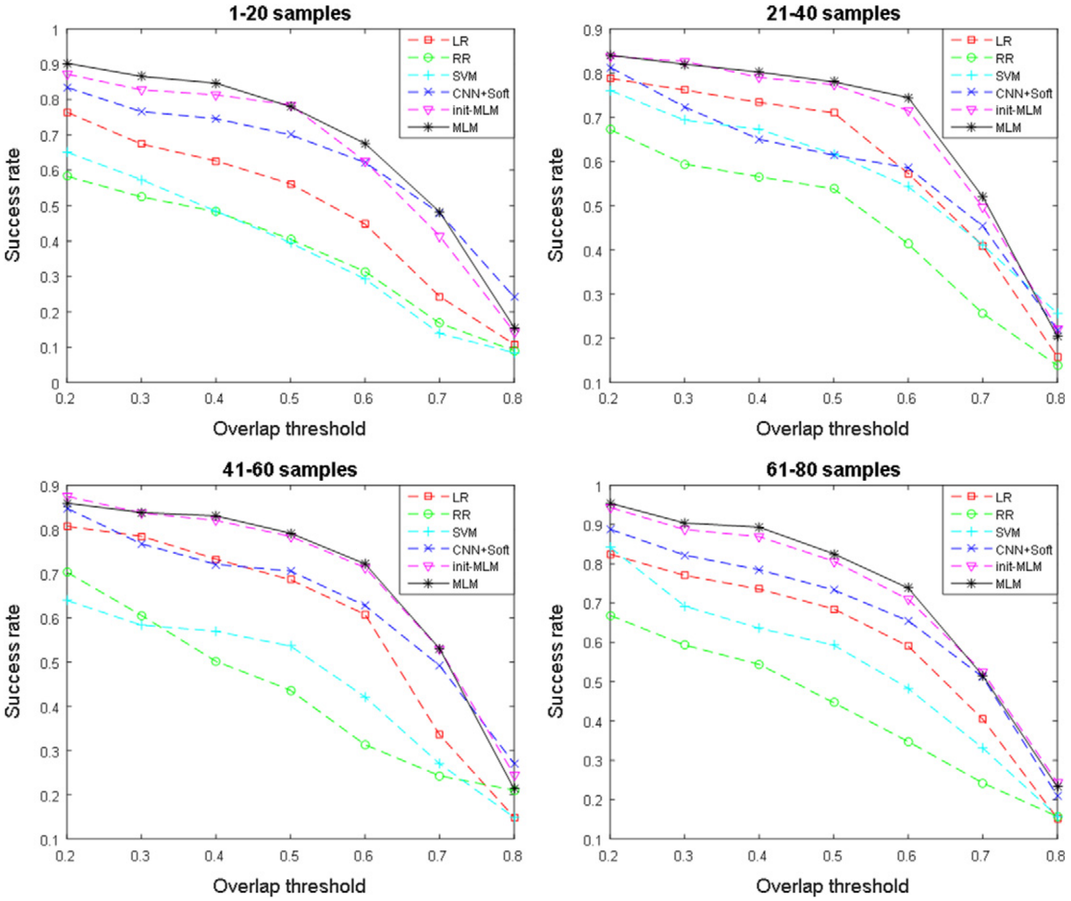


Fig. 7. The average success plots of 1–20, 21–40, 41–60, and 61–80 samples.

method while our MLM is observation model level method. It is obvious that our MLM method outperforms MTT in the cell tracking dataset.

The DLT method [8] is also neural networks based method in visual tracking. The Stacked Denoising Autoencoder (SDAE) [32] is used as the feature extractor in the literature. The DLT is verified to be a good cell tracking method especially in the 4, 6, 15 and 17 tracking sequences. Compared with DLT, Our MTL model uses CNN as the feature extractor. The assistant task is designed to deal with a classification problem in 30,000 samples while the SDAE of DLT is trained in about 80 million images, the DLT would spend more training time than our MTL model.

5.4. Robustness estimation

The robustness of cell tracking is considering significantly since the special cell living environment is full of plenty particles and organics. The popular evaluation methods contain temporal robustness evaluation (TRE) and spatial robustness evaluation (SRE). The TRE evaluates the temporal sensitivity by tracking target at first frame and initializing the trackers by bounding boxes with different sizes, while the SRE estimates the spatial performance by tracking target starting at different frames. The average precision plots at TRE and SRE metrics are drawn in Fig. 8. From the precision plots of SRE, it shows that the proposed MTL has robust

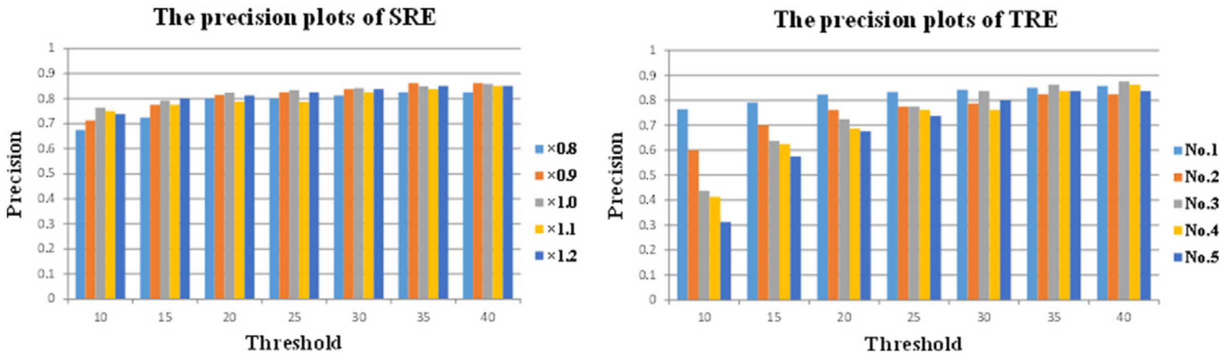


Fig. 8. The precision plots of SRE and TRE. The SRE test initializes the trackers by bounding boxes with different sizes, "× 0.8" denotes that the bounding box is 0.8 times than manual one. The TRE test initializes the trackers by tracking target starting at different frames. "No. 1" denotes that the initial frame of the tracker utilizes the 1-st image of the sequence.

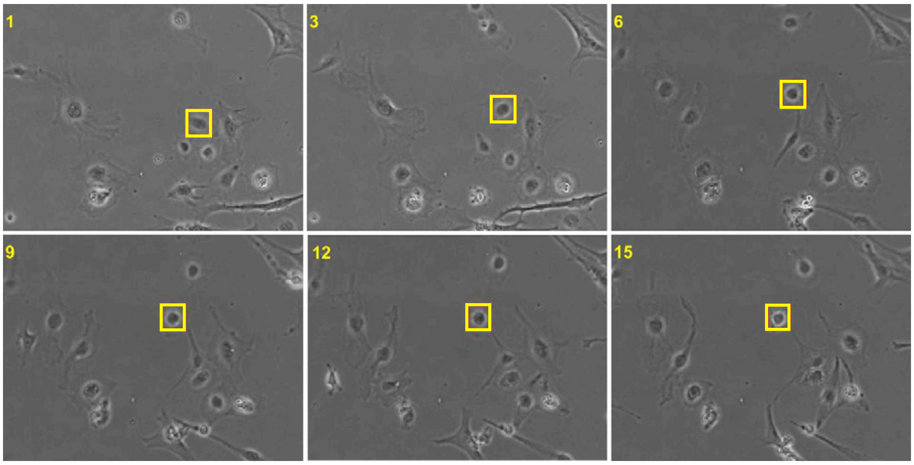


Fig. 9. The tracking example of stable activation.

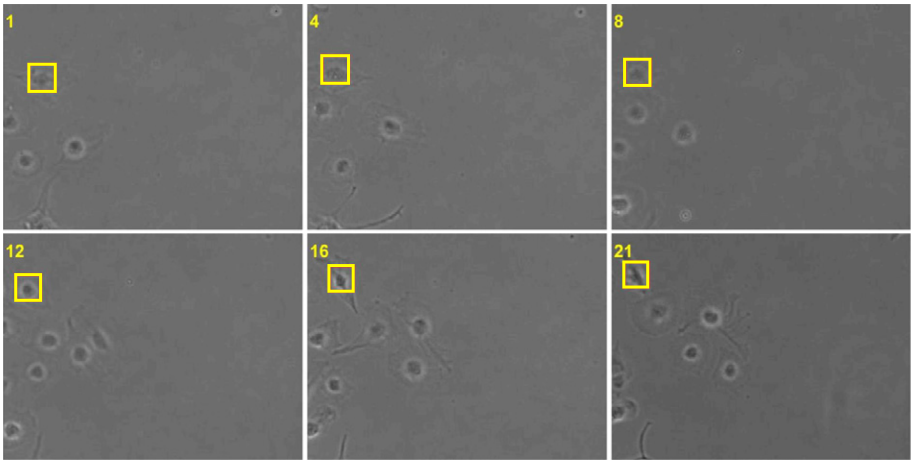


Fig. 10. The tracking example of drastically deformation.

space performance, but from the precision plots of TRE, it is sensitive to the temporal robustness especially when the threshold is lower. The reason of sensitive TRE could be caused by that the cell deformation happens frequently between the initial 5 cell image frames. It is more obvious when setting a lower central-pixel threshold, such as 10.

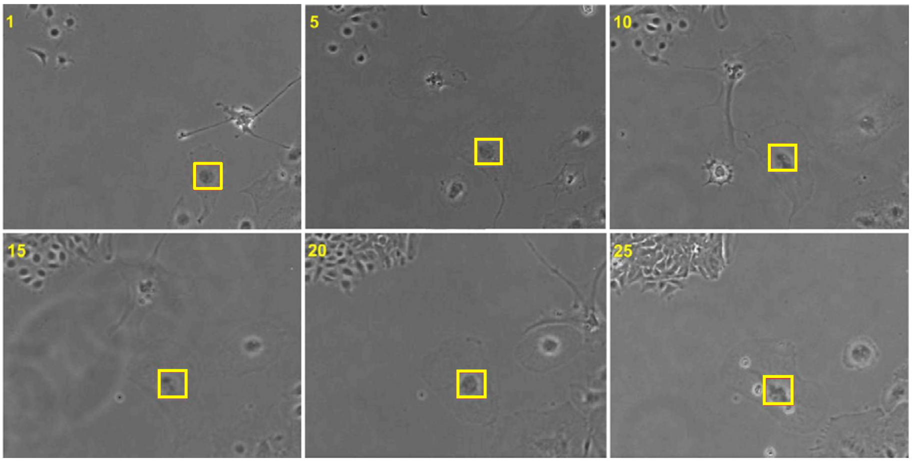


Fig. 11. The tracking example of variational environment.

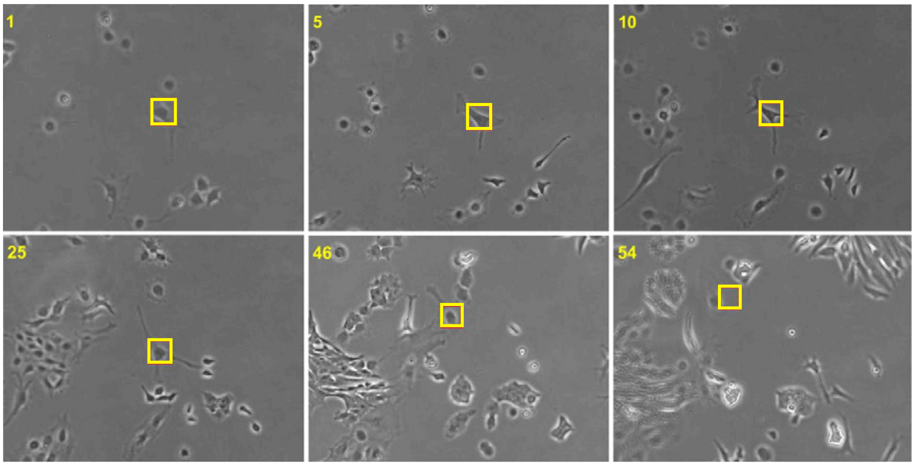


Fig. 12. The tracking example of movement together with deformation.

Table 4
The average success plots on the common dataset with threshold 50%.

Seq.	Ours	DLT	MTT	CT	VTD	ASLA	L1T	MIL	SCM
car4	**100**	**100**	**100**	24.7	35.2	89.1	30.8	24.7	89.0
car11	66.0	**100**	**100**	70.7	65.6	81.4	**100**	68.4	79.2
trellis	51.3	**93.6**	66.3	23.0	30.1	58.1	62.1	25.9	*84.3*
woman	**79.1**	67.1	19.8	16.0	17.1	65.2	21.1	12.2	68.6
shaking	**99.2**	88.4	12.3	92.3	**99.2**	39.8	0.5	26.0	67.2
singer1	**100**	**100**	35.6	10.3	99.4	52.0	**100**	10.3	84.8
surfer	**100**	86.5	83.8	13.5	90.5	9.3	75.7	44.6	61.1
football	58.6	54.4	71.1	78.5	**80.8**	57.2	65.7	55.1	69.4
box	76.1	72.5	25.6	33.3	34.1	59.7	4.5	65.1	**92.0**
average	81.1	**84.7**	57.2	40.3	61.3	56.9	51.2	37.0	77.3

From the high visual perspective, we divide the actions of cell targets into four situations listed as stable activation (Fig. 9), drastically deformation (Fig. 10), variational environment (Fig. 11), and movement together with deformation (Fig. 12). It is visible that the tracking is credible when the cell objects in stable activation, drastically deformation, and variational environment, but in more complicated environments, the cell objects would be lost after some frames, which is showed in 54-th frames of Fig. 12.

5.5. Experimental results on the common dataset

The proposed MTL method is tested on 9 video sequences of the TB-100 dataset.[1] Since the MTL model has an off-line learning stage, it is pre-trained on the CIFAR10 dataset. The CIFAR10 [35] is a widely used image classification dataset. It has 60,000 images and each image is 32 × 32 pixels that is the same as those of the initial bounding boxes in our experiment. The quantitative comparison results with other 8 methods are summarized in Tables 4 and 5. The best results are highlighted in bold face and the second results are underlined. The average success plot with overlap percentage threshold 50% and center location is reported as those in Refs. [1,8]. In Table 4, the method has best results in *car4*, *woman*, *shaking*, *singer1* and *surfer* and second results in *box* while the method has best results in *surfer* and second results in *woman*, *shaking* and *box* in Table 5. Totally, our method has second results on the average success plots and third results on the average center location errors.

Our method has a good performance in *car4* sequences but a bad result in *car11*. For *car11*, it has a dark environment, but there are no dark situations in the CIFAR10 dataset. Thus, our MTL observation model obtains a bad result in *car11* as maybe it haven't learned image features of the dark environment. In the *woman* sequences, the woman is severely occluded by the parked cars. Our method tracks the woman continuously when part of her body is occluded. In *shaking* and *singer1* sequences, our method outperforms some others in the illumination changes situation. In *surfer* sequences, our method has a good performance when the surfer changes the pose of his head frequently. In other sequences, our method doesn't have the best results but is near the average.

6. Conclusion

In this paper, we propose a cell tracking method to track the non-rigid and non-significant cells. Considering the four challenges in the cell tracking, the MTL method and CNNs are applied as the observation model to keep the performance of the cell tracking. The dataset used in this literature is taken from the real world cell image sequences captured from microscope. Extensive experiments show the superiority of the proposed method over other 9 methods in cell tracking problem.

Though the proposed method has been demonstrated to be a good tracker, it only considers single cell tracking in this paper. In the future work, we will extend the proposed method to multi-object cell tracking. Some challenges such as cell deformation, cell migration, and cell living environment are considered in our work, but there are many other challenges existing in multi-object cell tracking. First, recognizing the cell division event is significant in cell tracking. Second, it is challenging to detect cell apoptosis. Finally, the cell occlusion should be considered. As a common visual tracking

[1] http://cvlab.hanyang.ac.kr/tracker_benchmark/datasets.html.

Table 5
The average center location errors on the common dataset.

Seq.	Ours	DLT	MTT	CT	VTD	ASLA	L1T	MIL	SCM
car4	3.7	6.0	**3.4**	95.4	41.5	4.3	16.8	81.8	3.5
car11	23.5	**1.2**	1.3	6.0	23.9	2.0	1.3	19.3	1.8
trellis	22.9	**3.3**	33.7	80.4	81.3	35.7	37.6	71.7	11.5
woman	9.6	**9.4**	257.8	109.6	133.6	43.5	138.2	123.7	30.8
shaking	9.0	11.5	28.1	10.9	**5.2**	31.7	90.8	28.6	9.4
singer1	5.2	**3.3**	34.0	16.8	3.4	14.5	3.7	26.0	3.7
surfer	**1.2**	4.6	6.9	18.7	5.5	164.4	9.5	14.7	23.0
football	11.2	7.1	6.5	11.9	**4.1**	18.0	9.3	16.0	10.4
box	20.4	24.0	54.8	169.0	114.1	49.1	77.5	109.0	**8.4**
average	11.9	**7.8**	47.4	57.6	45.8	40.4	42.7	54.5	11.4

method, the run time will be reduced with the help of GPU parallel computing in our future work.

References

[1] Y. Wu, J. Lim, M.-H. Yang, Online object tracking: a benchmark, Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013. pp. 2411–2418.

[2] J. Hu, J. Lu, Y.-P. Tan, Deep metric learning for visual tracking, IEEE Trans. Circuits Syst. Video Technol. (2015) http://dx.doi.org/10.1109/TCSVT.2015.2477936.

[3] S. Huh, S. Eom, R. Bise, Z. Yin, T. Kanade, Mitosis detection for stem cell tracking in phase-contrast microscopy images, Proceedings of the 8th IEEE International Symposium on Biomedical Imaging (ISBI), 2011. pp. 2121–2127.

[4] X. Yang, H. Li, X. Zhou, Nuclei segmentation using marker-controlled watershed, tracking using mean-shift, and Kalman filter in time-lapse microscopy, IEEE Trans. Circuits Syst. 53 (11) (2006) 2405–2414.

[5] F. Yang, M.A. Mackey, F. Ianzini, G. Gallardo, M. Sonka, Cell segmentation, tracking, and mitosis detection using temporal context, Proceedings of the 2nd International Conference on Medical Image Computing and Computer-assisted Intervention (MICCAI), 2005. pp. 302–309.

[6] K.E. Magnusson, J. Jaldén, A batch algorithm using iterative application of the Viterbi algorithm to track cells and construct cell lineages, Proceedings of the 9th IEEE International Symposium on Biomedical Imaging (ISBI), 2012. pp. 382–385.

[7] X. Chen, X. Zhou, S.T. Wong, Automated segmentation, classification, and tracking of cancer cell nuclei in time-lapse microscopy, IEEE Trans. Biomed. Eng. 53 (4) (2006) 762–766.

[8] N. Wang, D. Yeung, Learning a deep compact image representation for visual tracking, Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS), 2013. pp. 809–817.

[9] E. Meijering, O. Dzyubachyk, I. Smal, Methods for cell and particle tracking, Methods Enzymol. 504 (9) (2012) 183–200.

[10] Z. Yi, K.K. Tan, Convergence Analysis of Recurrent Neural Networks, vol. 13. Kluwer, Boston, MA, 2004.

[11] J. Lu, G. Wang, W. Deng, P. Moulin, J. Zhou, Multi-manifold deep metric learning for image set classification, Proceedings of the 25th the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015. pp. 1137–1145.

[12] D.C. Ciresan, U. Meier, J. Masci, L. Maria Gambardella, J. Schmidhuber, Flexible, high performance convolutional neural networks for image classification, Proceedings of the 32nd International Joint Conference on Artificial Intelligence (IJCAI), 2011. pp. 1237–1242.

[13] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Proceedings of the 24th Annual Conference on Neural Information Processing Systems (NIPS), 2012. pp. 1106–1114.

[14] J. Fan, W. Xu, Y. Wu, Y. Gong, Human tracking using convolutional neural networks, IEEE Trans. Neural Netw. 21 (10) (2010) 1610–1623.

[15] Y.S. Abu-Mostafa, Learning from hints in neural networks, J. Complex. 6 (2) (1990) 192–198.

[16] P. Gong, J. Ye, C. Zhang, Robust multi-task feature learning, Proceedings of the 18th ACM International Conference on Knowledge Discovery and Data Mining (KDD), 2012. pp. 895–903.

[17] D. Dong, H. Wu, W. He, D. Yu, H. Wang, Multi-task learning for multiple language translation, Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL), 2015. pp. 1723–1732.

[18] G. Heigold, V. Vanhoucke, A.W. Senior, P. Nguyen, M. Ranzato, M. Devin, J. Dean, Multilingual acoustic models using distributed deep neural networks, Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing, (ICASSP), 2013. pp. 8619–8623.

[19] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, DeCAF: a deep convolutional activation feature for generic visual recognition, Proceedings of the 31st International Conference on Machine Learning (ICML), 2014. pp. 647–655.

[20] D.P. Mukherjee, N. Ray, S.T. Acton, Level set analysis for leukocyte detection and tracking, IEEE Trans. Image Process. 13 (4) (2004) 562–572.

[21] Y. Ren, B. Xu, J. Zhang, W. Zhang, L. Xu, A generalized data association approach for cell tracking in high-density population, Proceedings of the 4th IEEE International Conference on Control, Automation and Information Sciences (ICCAIS), 2015. pp. 502–507.

[22] B. Babenko, M.-H. Yang, S. Belongie, Robust object tracking with online multiple instance learning, IEEE Trans. Pattern Anal. Mach. Intell. 33 (8) (2011) 1619–1632.

[23] O. Dzyubachyk, W.A. van Cappellen, J. Essers, W.J. Niessen, E.H.W. Meijering, Advanced level-set-based cell tracking in time-lapse fluorescence microscopy, IEEE, Transactions on Medical Imaging 29 (3) (2010) 852–867.

[24] C. Huang, B. Wu, R. Nevatia, Robust object tracking by hierarchical association of detection responses, Proceedings of the 10th European Conference on Computer Vision (ECCV), 2008. pp. 788–801.

[25] L. Zhang, Y. Li, R. Nevatia, Global data association for multi-object tracking using network flows, Proceedings of the 18th IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2008. pp. 1–8.

[26] R. Bise, Z. Yin, T. Kanade, Reliable cell tracking by global data association, Proceedings of the 8th IEEE International Symposium on Biomedical Imaging (ISBI), 2011. pp. 1004–1010.

[27] N. Wang, J. Shi, D.-Y. Yeung, J. Jia, Understanding and diagnosing visual tracking systems, Proceedings of the 15Th IEEE International Conference on Computer Vision (ICCV), 2015. pp. 3101–3109.

[28] Z. Yi, Foundations of implementing the competitive layer model by Lotka—Volterra recurrent neural networks, IEEE Trans. Neural Netw. 21 (3) (2010) 494–507.

[29] X. Lou, F.A. Hamprecht, Structured learning for cell tracking, Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NIPS), 2011. pp. 1296–1304.

[30] K. Li, E.D. Miller, M. Chen, T. Kanade, L.E. Weiss, P.G. Campbell, Cell population tracking and lineage construction with spatiotemporal context, Med. Image Anal. 12 (5) (2008) 546–566.

[31] M.S. Arulampalam, S. Maskell, N.J. Gordon, T. Clapp, A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking, IEEE Trans. Signal Process. 50 (2) (2002) 174–188.

[32] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P. Manzagol, Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion, J. Mach. Learn. Res. 11 (2010) 3371–3408.

[33] J. Zhang, S. Ma, S. Sclaroff, MEEM: robust tracking via multiple experts using entropy minimization, Proceedings of the 13rd European Conference on Computer Vision (ECCV), 2014. pp. 188–203.

[34] T. Zhang, B. Ghanem, S. Liu, N. Ahuja, Robust visual tracking via multi-task sparse learning, Proceedings of the 22nd IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012. pp. 2042–2049.

[35] L. Wan, M.D. Zeiler, S. Zhang, Y. LeCun, R. Fergus, Regularization of neural networks using DropConnect, Proceedings of the 30th International Conference on Machine Learning (ICML), 2013. pp. 1058–1066.