Sub-Markov Random Walk for Image Segmentation

Xingping Dong, Jianbing Shen, Senior Member, IEEE, Ling Shao, Senior Member, IEEE, and Luc Van Gool

Abstract—A novel sub-Markov random walk (subRW) algorithm with label prior is proposed for seeded image segmentation, which can be interpreted as a traditional random walker on a graph with added auxiliary nodes. Under this explanation, we unify the proposed subRW and other popular random walk (RW) algorithms. This unifying view will make it possible for transferring intrinsic findings between different RW algorithms, and offer new ideas for designing novel RW algorithms by adding or changing auxiliary nodes. To verify the second benefit, we design a new subRW algorithm with label prior to solve the segmentation problem of objects with thin and elongated parts. The experimental results on both synthetic and natural images with twigs demonstrate that the proposed subRW method outperforms previous RW algorithms for seeded image segmentation.

Index Terms—Seeded image segmentation, subMarkov, random walk, optimization, label prior, complex texture.

I. INTRODUCTION

R ANDOM walk (RW) has been widely used for many different tasks in computer vision and machine learning, such as segmentation [20], [23], [28], [35], clustering [19], [25], ranking [13], [31], classification [10], [41] and the other applications [1], [13], [18], [22], [26], [33]. Grady and Funka-Lea [15] first proposed the RW for medical image segmentation and extended it in [20] for general image segmentation. In their work, the user should give labels to a small number of pixels. Assuming a random walker starts at each unlabeled pixel, they calculate the probabilities that these random walkers first reach the pre-labeled pixels. These probabilities will determine the probability of each unlabeled pixel belonging to a label. By assigning each pixel to a label with the greatest probability, the interactive image segmentation result can be obtained. After [15], many related and important methods based on RW [17], [23], [34], [35]. In [17] have been proposed, the RW has been extended to segment out disconnected objects by using prior models without labeling each object. In other words, a user only needs to indicate labels on some objects and the other similar objects will be segmented out. Sinop and Grady [23] proposed a common framework to unify the RW, graph cuts, and shortest path algorithms for interactive segmentation. Furthermore, they added the popular watershed segmentation algorithm to this framework [35] and made the theoretical analysis for the connections between these algorithms. This unified framework brings some benefits, including opening new possibilities for using unary terms in traditional watershed algorithms to optimize more general models.

Recently, some researchers [27], [40] have focused on segmenting natural images with complex textures. They extend the RW algorithm and obtain better performance for these challenging images. In general, these algorithms are graph-based so we can use a graph to describe an image for introducing them. Kim et al. [27] proposed a random walker with a restarting probability (RWR) for segmentation. It means that this random walker will return to the starting node with a probability c at each step, and walk to other adjacent nodes with probability 1 − c. Shen et al. [40] have developed the lazy random walk (LRW) for superpixel segmentation. A LRW will stay at the current node with a probability 1 − α and walk out along the edges connected with the current node with probability α. Wu et al. [37] proposed another similar RW algorithm called partially absorbing random walk (PARW) for applications based on cluster, such as ranking and classification. In PARW, a random walker is absorbed at current node i with a probability αi and follows a random edge out of it with probability 1 − αi. And they analyze the relations between PARW and other popular ranking and classification models, such as PageRank [7], hitting and commute times [32], and semi-supervised learning [11], [16]. Comparing the above three RW-based algorithms, we can conclude that they all satisfy the subMarkov property [30], i.e., the sum of transition probabilities q(i, j) that a random walker starts from a node to other adjacent nodes is less than or equal to 1. Is there a common framework to unify these algorithms? This is the problem we want to first solve in this work.

The other problem is how to segment objects with thin and elongated parts (twig problem) in natural images, which is difficult for most RW-based algorithms. For example, Figs. 1 (c)-(e) show the failed segmentation results of popular RW-based algorithms, including RW [20], RWR [27], and LRW [40]. One reason may be that the distance between
Fig. 1. Comparison results between our subRW method and three well-known RW-based algorithms. (a) Is an input image; (b) Is the scribbled image, where the red/blue scribbles specify the foreground/background regions respectively. (c)-(f) Are the corresponding results by RW [20], RWR [27], LRW [40], and our subRW algorithm with label prior. The red lines denote the boundaries of twig segmentation.

the twig parts and the corresponding seeds is so far that the random walker with a small probability starts from these seeds to twig parts (or from twig to seeds).

In this paper, we propose a novel subMarkov random walk (subRW) framework to unify four RW-based algorithms: RW, RWR, LRW and PARW, and extend it by adding label prior to solve the twig problem. First, according to the subMarkov property, we build a subRW framework for image segmentation. In subRW, a random walker will leave a graph $G$ from a node $i$ with probability $c_i$ and walk to the other adjacent nodes in $G$ with probability $1 - c_i$. This random walker can be transformed to a random walker with Markov transition probability $(q(i, j) = 1)$ that walks in an expanded graph $G'$. This graph is constructed by adding auxiliary staying nodes connected with seeds and auxiliary killing nodes connected with unseeded nodes into graph $G$. In order to further understand the subRW, we give a detailed optimization explanation. Then we unify the subRW and the aforementioned four RW-based algorithms in the expanded graph. After analyzing the connections between them, we design a new RW-based algorithm by changing edges or adding auxiliary nodes. According to this idea, we introduce a novel subRW with label prior to solve the twig problem. This label prior can be viewed as global ‘seeds’ connected with all nodes. Each global ‘seed’ corresponds to a label. So we can add some prior nodes connected with all nodes into graph $G'$ to build a new expanded graph $G_p$. Then we compute the probability that a random walker starting from each node reaches the staying nodes or the prior nodes in graph $G_p$, as the likelihoods probability of corresponding labels. In other words, we want to compute the probability of reaching the user specified seeds plus the probability of reaching the global ‘seeds’. These global ‘seeds’ will help to segment out the twig parts. As shown in Fig. 1 (f), our approach has successfully segmented the twigs of the tree using the proposed subRW algorithm with label prior information.

A preliminary version of this work was presented in [43], and the present is significantly different. The popular random walk with Markov transition probability is first added into our unified optimization framework, which makes this framework more complete. Considerable new theoretical analysis and proofs are added into the initial subRW algorithm, such as the uniqueness and a new optimization explanation for subRW with label prior, which make it applicable for more vision applications based on optimization. The optimization framework and explanation improves the initial algorithm to be more suitable for multi-label segmentation. We also extend the original experiments from 2-label segmentation to multi-label segmentation. Our source code will be available at.\footnote{http://github.com/shenjianbing/subrw}

The main contributions are summarized as follows:

1) A novel random walk (subRW) with label prior is proposed for unifying well-known RW-based algorithms, such as RW [20], RWR [27], LRW [40], and PARW [37], which all satisfy the SubMarkov property, making it easier to convert the intrinsic findings between them.

2) The subRW is interpreted as a general optimization problem, which makes it easier to find the latent problem of the subRW for different vision applications. For example, from the optimization explanation of subRW with label prior, we find the consistence between label prior and reaching probability may be violated in multi-label segmentation, and we successfully solve this problem.

3) We further introduce a novel subRW algorithm by adding auxiliary nodes into the original graph. According to this idea, a novel subRW method with label prior is proposed to solve the twig segmentation problem with thin and elongated objects. The optimization explanation is also given.

II. A Unifying View of subRW

In this section, we propose a novel random walk algorithm with a subMarkov transition probability (subRW) for interactive multi-labeled image segmentation, provide the optimization explanation of subRW, and analyze the relations between this proposed algorithm and other popular RW algorithms, such as RW [20], RWR [27], LRW [40], and PARW [37].

We first give some important notations and their corresponding descriptions. An image is formulated as a weighted undirected graph $G = (V, E)$ with nodes $v \in V$, and edges $e \in E \subseteq V \times V$. Each node $v_i$ represents an image pixel $x_i$. An edge $e_{ij}$ connects two nodes $v_i$ and $v_j$ in 8-connected neighborhood system. A weight $w_{ij} \in \mathbb{W}$ of edge $e_{ij}$ measures the likelihood that a random walker will cross this edge.
As many previous graph-based segmentation algorithms [20], [21], [27], [40], a weight $w_{ij}$ is defined as the following weighting function:

$$w_{ij} = \exp(-\frac{I_i - I_j}{\sigma}) + ,$$  \hspace{1cm} (1)

where $I_i$ and $I_j$ are the pixel colors at two nodes $v_i$ and $v_j$ in Lab color space, $\sigma$ is a controlling parameter which is set as 1/60 in this paper, and is a small constant as $10^{-6}$. The degree matrix $D$ is a diagonal matrix where $D_{ii} = d_i$, $d_i = \sum_{j \sim i} w_{ij}$ is the degree of a node $v_i$, $j \sim i$ represents a node $v_j$ in the neighborhood (not including itself) of $v_i$. $N$ is the number of nodes (pixels).

In our approach, a user needs to indicate some scribbles on foreground objects and backgrounds as multi-labeled seeds. Here, we define these seeds as a set of labeled nodes $V_M = \{V^{h}, V^{l}, \cdots, V^{k}\}$. Then a set of labels $LS = \{l_i, l_2, \cdots, l_K\}$ is also defined, where $K$ is the number of labels $V^{k} = \{v^{k}, v^{k}, \cdots, v^{k}\}$ and $M_k$ is the number of seeds with label $l_k$.

A. The SubMarkov Random Walk

Given a weighted graph $G$, a set of labeled nodes $V_{M}$, and a set of unlabeled nodes $V_{U}$, where $V_{U} \cup V_{M} = V$, the multi-labeled image segmentation can be formulated as a labeling problem, where each node $v_i \in V$ should be assigned with a label from set $LS$. This problem can be solved by comparing the probability $r_{jk}^i$ of each node belonging to a label $l_k$ in our algorithm. Before computing this probability, we define the subMarkov transition probability $q$ on $V$ as follows:

**Definition 1:** $q$ denotes a subMarkov transition probability if for each node $v_i$

$$q(i, j) = \begin{cases} 1 & j \sim i \\
\frac{1}{d_i} \sum_{j \sim i} w_{ij} & \text{otherwise}. \end{cases}$$  \hspace{1cm} (2)

According to [30], a subMarkov transition probability has the following property:

**Property 1:** Through adding an auxiliary node , a subMarkov transition probability $q$ on $G$ can be made into a (Markov) transition probability on $V \cup \{\emptyset\}$ by setting $q(\emptyset) = 1$ and $q(i, \emptyset) = 1 - q(i, j)$. The probability $q(i, \emptyset)$ can be viewed as a probability that a random walker leaves graph $G$.

According to the above property, we can design different subMarkov random walk algorithms by adding different auxiliary nodes. In fact, popular random walk algorithms, such as RW [20], RWR [27], LRW [40], and PARW [37], can be interpreted in this view (more details will be given in the next subsection). The other advantage of a subMarkov transition probability is that it will help to improve segmentation performance connected with all unseeded nodes (e.g., the green circle node in Fig. 2). When a random walker from a node $v_i$ reaches this node, it will be killed at this node and the corresponding probability will be omitted. In other words, an effective random walker will not reach this node. The other one is the staying node $s_{hk}$ connected with the $m$-seeded node with label $l_k$ (e.g., the blue, orange or purple circle nodes in Fig. 2). When a random walker reaches this node which can be viewed as a target node, it will stay at this node. We denote $S_M$ as a set of staying nodes corresponding to seeds set $V_M$, and $c_i$ as the leaving probability for each node in $V$. Then, the transition probability on $V \cup \{\emptyset\} \cup S_M$ is formulated as follows:

$$q(i, j) = \begin{cases} c_i, & \text{if } i \in V, j \in \{\emptyset\} \cup S_M \\
\frac{(1 - c_i) w_{ij}}{d_i}, & \text{if } j \sim i \in V \\
1, & \text{if } i = j \in \{\emptyset\} \cup S_M \\
0, & \text{otherwise}. \end{cases}$$  \hspace{1cm} (3)

Suppose a random walker starts from a node $v_i \in V$ and walks on $V \cup \{\emptyset\} \cup S_M$ with the transition probability $q(i, j)$ in (3). By setting $r_{ih}^j$ as the reaching probability that this random walker reaches the auxiliary staying node $s_{hk}$, then we have:

$$r_{ih}^j = \begin{cases} \frac{(1 - c_i) w_{ij}}{d} r_{jm}^k + c_i \cdot 1 & \text{if } v_i = v_{hm}^j, \\
\frac{(1 - c_i) w_{ij}}{d} r_{jm}^k + c_i \cdot 0 & \text{otherwise}. \end{cases}$$  \hspace{1cm} (4)

The vector notation $r_{im}^k = [r_{im}]_{N \times 1}$, which is formulated as follows:

$$r_{im}^k = [r_{im}]_{N \times 1},$$

where $N$ is the number of nodes and $k$ is the number of labels.
formance in images with complex texture. More experiments and analysis for solving the complex texture problem given in Section IV-B.

We first consider a general subRW algorithm for interactive seeded image segmentation. Two kinds of auxiliary nodes are added into graph $G$ to get an expanded graph $G_e$. As shown in Fig. 2, one kind of auxiliary node is a killing

\[ m = (I - D_c)P^k m + D_c b^l m, \]

where $b^k N = [b^l N] \times$ is the $N^\text{th}$-dimensional indicating vector with $b^k = 1$ if $v_i = v^k$ and $b^k = 0$ otherwise. $D_c$ is a diagonal matrix whose diagonal element is $c_i$ i.e. $D_c = diag(c_1, c_2, \cdots, c_N)$, and $I$ is a $N \times N$ identity matrix. The transition matrix $P = [p_{ij}]_{N \times N}$ is a row-normalized
matrix of the adjacency matrix $W$ (defined in (1)):

$$p_{ij} = w_{ij}/d_i.$$  \hspace{1cm} (6)

The number of a set of seeded nodes with a same label is often larger than one. A good RW approach should consider all seeded nodes. We use an average reaching probability $r_{ik}^l$, that a random walker from a node $v_i$ reaches a set of staying nodes with label $l_k$, as the likelihood of assigning this node to label $l_k$. (5) can be rewritten as follows:

$$r_{ik}^l = (I - (I - D_c)P)^{-1} D_k b_i^l = E^{-1} D_k b_i^l,$$  \hspace{1cm} (7)

where $E = I - (I - D_c)P$.

Thus, a vector formulation of this average steady-state probability $r_{ik}^l$ can be given as follows:

$$r_{ik}^l = \frac{1}{Z_k M_k} \sum_{m=1}^{M_k} r_{ik}^l = \frac{1}{Z_k M_k} E^{-1} D_k b^l,$$  \hspace{1cm} (8)

where $b^l = [b_{ik}^l]_{N \times 1}$ is a vector with $b_{ik}^l = 1$ if $v_i \in V^l$ and $b_{ik}^l = 0$ otherwise, and $Z_k$ is a normalized constant. The final labeling result (i.e., the segmentation result) for each node $v_i \in V$ is obtained as follows:

$$R_i = \arg \max_{l_k} r_{ik}^l.$$  \hspace{1cm} (9)

Note that, in (8), directly computing the inverse of matrix $E$ may be very expensive, especially when the number of nodes is very large. Fortunately, (8) can be converted to solve the linear equations and the corresponding coefficient matrix $E$ is a sparse matrix. We can use the existing sparse algorithms [3]-[5] to solve these linear equations. More importantly, if $0 < c_i < 1$, $\forall i \in V$, matrix $E$ is non-singular (the proof is given in Appendix A). The solution of these linear equations is unique in theory, when $0 < c_i < 1$, $\forall i \in V$. Thus, the segmentation results are unique.

B. The Optimization Explanation of SubRW

Although the subRW algorithm is based on random walks, it can also be interpreted as a general energy optimization problem, which will make the subRW easier to be employed for general vision applications using optimization [29], [34]. Now we will give the optimization explanation in detail.

Firstly, suppose $\forall i$, $0 < c_i < 1$, then let us consider the following objective function:

$$O^k = \sum_{i,l} O_{ik}^l = \sum_{i,l} (d_{ik}^{l} - b_{ik}^{l})^2,$$  \hspace{1cm} (10)

By taking the partial derivative of $r_{ik}^l$, we have:

$$\frac{\partial O^l_k}{\partial r_{ik}^l} = (I - D_c)^{-1} \frac{\partial}{\partial r_{ik}^l} [D_c r_{ik}^l - b_{ik}^l].$$

where $E = I - (I - D_c)P = I - (I - D_c)D^{-1} W$. By setting

$$\frac{\partial O^l_k}{\partial r_{ik}^l} = 0,$$  \hspace{1cm} (11)

minimizing the objective function $O^k$ is equivalent to the solution of subRW.

Let us analyze the objective function (10). This objective function consists of two components. The first component is a common smoothing term. Minimizing this term will keep the consistence of the local probabilities, i.e., when the nodes are more similar, their probabilities are more likely to be the same. The second component can be viewed as a unary term corresponding to seeds. By minimizing this term, probability $r_{ik}^l$ keeps close to the predefined value $b_{ik}^l$ which represents the information of seeds. $c_i$ is the regularization parameter, and it decides which component is more important.

C. Relations With Other Well-Known RW Algorithms

We will further analyze the relations between the proposed subMarkov random walk and other popular algorithms: RW [20], RWR [27], LRW [40], and PARW [37]. Then we find these algorithms can be unified and related with the subRW.

1) Relations With RW: In [20], Grady proposed a random walk algorithm with Markov transition probability for segmentation, which can be viewed as a special case of subRW. This algorithm places a random walker at each unlabeled node and then computes which labeled nodes they first arrive at. This computation would be completely impractical. Fortunately, the combinatorial Dirichlet algorithm provides a simple and convenient method for computing this problem [20]. Here, we directly give formulation of this solution. More details can be found in [20]. Denote $x^l$ as the probabilities of random walkers first arriving at the labeled nodes with label $l_k$ and $L$ as the Laplacian matrix, where $L = D - W$ [6]. Without loss of generality, we may assume that the nodes in $L$ and $x^l$ are ordered such that the labeled nodes are in first and the unlabeled nodes are in second. Then we have the block matrix of $L$ and $x$:

$$L = \begin{bmatrix} L_M & B_{l_k} \\ B_{l_k}^T & L_{U} \end{bmatrix}, \quad x = \begin{bmatrix} x^l_M \\ x^l_U \end{bmatrix}.$$  \hspace{1cm} (13)
\[ O^{lk} = \sum_{i=1}^{\frac{\tau}{2}} \sum_{j=1}^{\frac{\tau}{2}} w_{ij} (r_{im}^l - r_{jm}^l)^2 \]

\[ + \sum_{i=1}^{\frac{\tau}{2}} \frac{d_i c_i}{1 - c_i} (r_{im}^l - b_{im}^l)^2. \]  

(10)

The vector formulation of above equations is defined as:

\[ O^{lk} = \frac{1}{2} r_m^T (D - W) r_m^l \]

\[ + \frac{1}{2} (r_m - b_m) (I - D_c)^{-1} D_c (r_m - b_m). \]  

(11)

Different from the subRW, this method only needs to calculate the probabilities of unlabeled nodes and the final solution is as follows:

\[ L_U x_U^{lk} = -B^T x_U^{lk}, \]  

(14)

where \( x_M^{lk} \) is given by the labeled nodes.

By setting the indicating vector \( b^{lk} = [b_M^{lk}, b_U^{lk}]^T \), then we obtain:

\[ x_M^{lk} = b_M^{lk}. \]  

(15)
For the comparison between RW and subRW, we should rewrite (14). Denote the block matrix of $W, D$ as:

$$
W = \begin{bmatrix} W_M & W_B \\ W_B^T & W_U \end{bmatrix}, \quad D = \begin{bmatrix} D_M & 0 \\ 0 & D_U \end{bmatrix}.
$$

Note that $L = D - W$, so $L_U = D_U - W_U$ and $B = -W_B$. Substitute these above equations into (14), we then have:

$$
(D_U - W_U)x_i^k = W_B^T x_i^k + B_M x_i^k,
$$

Rewrite (18) as follows:

$$
d_i x_j^k = w_{ij} x_i^k + w_{ij} x_i^k, \quad i \in V_U
$$

where $x_j^k = b_j^k$, for $j \in V_M$.

Recall (8) for computing the desired probabilities of subRW, we can rewrite it as follows:

$$
r_{ij}^k = \frac{w_{ij} x_i^k c_i}{d_i} l_i.
$$

Comparing (15), (20) and (21), it indicates that if $1, i \in V_M, Z_k = 1/M_k, c_i = 0, i \in V_U$, then the subRW is equivalent to the RW.

2) Relations With RWR: Kim et al. [27] suppose a random walker starts from an $m$-th seed node $v_m^k$ of label $l_k$ in a graph $G$. Different from the traditional random walker, it has a restarting probability $c$ to return to seed $v_m^k$ at each step.

Then each node is assigned a steady-state probability $r_{ij}^k$ that this random walker will finally stay at this node, which is formulated as:

$$
r_{ij}^k = (1 - c) w_{ij} r_{ij}^k + c \cdot b_j^k.
$$
(4) of subRW is rewritten as follows:

\[
    r_{im}^h = (1 - c_i) \frac{w_{ij}}{d_i} r_{jm}^h + c_i \cdot b_{im}^h. \tag{23}
\]

Combining (22) and (23), we find that RWR is a special case of subRW with leaving probability \( c_i = c, i = 1, 2, \ldots, N \).
In other words, the subRW can be viewed as a set of RWR with a different restarting probability at each node.

3) Relations With LRW: In [40], Shen et al. proposed a lazy random walker for superpixel segmentation, which is viewed as a multi-labeled segmentation problem. Under their framework, a random walker will stay at the current position with probability \( 1 - a \) and walk out along an arbitrary edge with probability \( a \). They use the commute time \( CT_{ij} \) [14] to measure the probability that these two nodes belong to the same label, where \( CT_{ij} \) is the expected number of steps for a lazy random walker starting at \( v_i \) to \( v_j \) and then returning. After normalizing the commute time, probability \( f^h_m \), that

\[
    f^h_m \in \begin{cases} 
        1, & \text{if } i = j \in \{ \} \cup S_M \\
        0, & \text{otherwise.}
    \end{cases}
\]

4) Relations With PARW: In [37], partially absorbing random walks (PARWs) model were proposed for ranking, clustering, and classification. Wu et al. [37] suppose that a PARW is absorbed at current node \( i \) with probability \( \lambda_i \), and walks out of it following a random edge with probability \( 1 - \lambda_i \).
They set \( a_{ij} \) as the absorption probability, where a PARW starting from node \( i \) will be absorbed at node \( j \) with probability \( a_{ij} \). Then matrix \( A = [a_{ij}]_{N \times N} \) of absorption probabilities is formulated as:

\[
    A = (I + D - W)^{-1}, \tag{29}
\]

where \( D = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_N), \lambda_i \neq 0, i = 1, \ldots, N \).

To solve the segmentation or labeling problem, we can use the absorption probability \( r_{im}^h \) that a PARW starting from a seeded node \( v^h_m \) is absorbed at node \( v_i \), as the probabilities belonging to label \( l_k \). The vector notation \( r^h \) is:

\[
    r^h = Ab^h. \tag{30}
\]

\[
    \sum_{m} r^h_m = 1,
\]

\[
    r^h_m = 0, \quad \text{if } i = j \notin S_M
\]

\[
    r^h_m = 0, \quad \text{otherwise.}
\]
By combining (29), (30) can be rewritten as:

$$\mathbf{r}p^k_{im} = (\mathbf{C} + \mathbf{D})^{-1} (\mathbf{W} \mathbf{r}p^k_{im} + \mathbf{b}^k_m)$$

(31)

The above equation is equivalent to:

$$r_{pim} = \sum_{j-i} w_{ij} r_{pim}^j + \frac{\lambda_i}{d_i + \lambda_i} b_{im}^k.$$  

(32)

By comparing the above equation and (23), we find that (32) is equivalent to (23) when $c_i = \frac{\lambda_i}{d_i + \lambda_i}$, i.e. the PARW is equivalent to the subRW. In fact, after substituting $c_i = \frac{\lambda_i}{d_i + \lambda_i}$ into $1-c_i$, we have $\frac{1-c_i}{d_i + \lambda_i} = 1$.

5) Merits of a Unifying View: We have shown that subRW can unify or relate the popular models based on RW. There are at least two merits of the unifying view. First, it builds the connections between different RW-based algorithms, so that it is easier to transfer findings between them. For example, in PARW [37], the authors have proved that the choice of $\lambda_i = \alpha (Vi)$ can fulfill the cluster assumption. In other words, by fixing all $\lambda_i$ as a proper parameter, a desirable clustering performance with PARWs will be achieved. If we want to apply subRW to solve a clustering problem, the leaving probability $c_i$ can be fixed as $\frac{\alpha}{d_i + \alpha}$ to fulfill the cluster assumption - when $c_i = \frac{\lambda_i}{d_i + \lambda_i}$, the subRW is equivalent to the PARW. Another example is for segmentation. In RWR [27], the authors have discussed the influence of parameter $c$. In this paper, we have chosen parameter $c_i$ according to their suggestion since the subRW is the generalized version of RWR. Second, a unifying view offers a new way to design novel RW-based algorithms by adding some new auxiliary nodes or changing the edges between auxiliary nodes and original nodes in $V$. For example, the LRW can be viewed as an expansion of the subRW. Inspired by the second merit, we design a new subRW with label prior to segment out objects with twigs.

III. SUBRW WITH LABEL PRIOR FOR SEEDED IMAGE SEGMENTATION

An object with twigs can be separated into two parts: main branch object and twig part. Usually, the twig part is similar to the main object, so appropriate user-specified scribbles on the main object have included enough information for segmenting out the twig part. But most RW-based algorithms do not make full use of this information and often omit the twig part. In this section, we want to add a label prior constructed by these scribbles into the subRW to help segment out the twig part.

A. Adding Label Prior

In general, user-specified scribbles are considered as exact label prior. Unfortunately, this prior only works at the seeded nodes and all unseeded nodes do not have this prior. Therefore, we want to give all nodes in $V$ a new label prior, where $u^i$ denotes the probability density belonging to $H_k$ at node $v_i$. This distribution can be learned via many techniques, such as kernel estimation and Gaussian Mixture Model (GMM). Here, the GMM is used as the prior model.

Each prior distribution $H_k$ is viewed as a GMM learned by the seeded nodes with label $l_k$ (more details about GMM learning can be seen in [24], [36], and [38]). Given these prior distributions, we can add a set of prior auxiliary nodes $H_M = \{h_1, h_2, \ldots, h_k\}$ into expanded graph $G_s$ and get a graph with prior $G$. An example is shown in Fig. 3. Each prior node is connected with all nodes in $V$ and weight $w_{ihk}$ of an edge between a prior node $h_k$ and a node $v_i \in V$ is proportional to probability density $u^i_k$, i.e., $w_{ihk} \propto u^i_k$. In this paper, we set weight $w_{ihk}$ as:

$$w_{ihk} = (1 - c_i) \lambda u^i_k,$$

(33)

where $\lambda$ is a regularization parameter, which measures the importance of the prior distribution.

Then the transition probability on $V \cup \mathcal{S}_M \cup H_M$ is formulated as follows:

$$c_{i,j} \quad \text{if } i \in V, j \in \mathcal{S}_M$$

$$(1 - c_i) \lambda u^k_i \quad \text{if } i \in V, j = h_k$$

$$\tilde{q}(i, j) = \frac{(1 - c_i) d_i + \lambda g_i}{d_i + \lambda g_i} \quad \text{if } j \sim i \in V$$

which maybe less exact than user defined scribbles, but can be

FIG. 3. The nodes graph with prior nodes of a subRW. The red circle nodes represent the prior nodes, which should be connected with all the original ellipse nodes. We only show two edges for each prior node for simplification. One edge connects a seeded node, and the other connects an unseeded node. Except for the red circle nodes, the other nodes are defined as the same as the nodes in Fig. 2.
used for unseeded nodes. This label prior is constructed by the user scribbles, i.e., the seeded nodes. We can use probability distributions to build the prior model. Assume a label $l_k$ has an intensity distribution $H_k$ for each node, $k = 1, \ldots, K$.

Given a transition probability $\tilde{q}$ on a graph with prior $\tilde{G}$, the probability $r_{im}^k$, that a random walker from a node $v_i \in V$ reaches the $m$-th staying node $s^k_m$ with label $l_k$ or prior node $h_k$,

$$g_i = \prod_{k=1}^K u_i^k.$$
is formulated as follows:
\[
\bar{r}_{im} = \left(1 - c_i \right) \frac{j m}{d_i + \lambda g_i} + \left(1 - c_i \right) \frac{i m}{d_i + \lambda g_i} + c_i b_{im}^{b_k},
\]
(35)
where \(b_{im}^{b_k} = 1\) if \(v_i = v_m^{b_k}\) and \(b_{im}^{b_k} = 0\) otherwise.

In fact, this prior node \(h_k\) can be viewed as a new staying node with label \(l_k\), so this reaching probability of \(h_k\) should be considered. By setting a vector \(\bar{r}_k = [\bar{r}_m^{l_k}]_{N \times 1}\), we can get the vector formulation of (35):
\[
\bar{r}_k = (I - D_c) \bar{r}_k + (I - D_c) \bar{u}_k + D_c b_{m}^{b_k},
\]
(36)
where \(\bar{E} = I - (I - D_c) \bar{u}_k + D_c b_{m}^{b_k}\).

\[\bar{P} = [\bar{p}_{ij}]_{N \times N} \text{ in } V \text{ is defined as:}
\]
\[
\bar{p}_{ij} = \frac{w_{ij}}{d_i + \lambda g_i},
\]
(37)
\[\bar{u}_k = [\bar{u}_i^k]_{N \times 1} \text{ is a vector with}
\]
\[
\bar{u}_i^k = \frac{\lambda u_i^k}{d_i + \lambda g_i}.
\]
(38)
As mentioned before, we use the average reaching probability \(\bar{r}_k\) for each node \(v_i \in V\) as the probabilities belonging to label \(l_k\). The vector notation \(\bar{r}_k\) is formulated as:
\[
\bar{r}_k = \frac{1}{Z_k} \bar{E}^{-1}((I - D_c) \bar{u}_k + D_c b_{m}^{b_k}),
\]
(39)
The final labeling (segmentation) result with a label prior is obtained as follows:
\[
\bar{R}_i = \arg \max_{l_k} \bar{r}_k,
\]
(40)
where \(\bar{R}_i\) represents the final label for each node, i.e., each pixel in an image.

In fact, solving (39) can be converted to solving the equivalent sparse linear equations. We will scale the probability density to better utilize the GMMs, since the GMMs need the log operation to reduce the cost of computation. These algorithm steps are shown in Algorithm 1. And the solution of these equations is also unique, if \(0 < c_i \leq 1, \forall i \in V\) (the proof is given in Appendix A).

### Algorithm 1 Segmentation by SubRW With Label Prior

**Input:** An image \(V(v_i)\) and \(K\) kinds of user scribbles \(V_M = \{V_1, V_2, \ldots, V_K\}\), the parameters \(D_c = \text{diag}(c_1, c_2, \ldots, c_N), \lambda, \gamma\);

**Output:** The segmentation result \(\bar{R}_i\) for each pixel:
1. Obtain the indicating vectors \(b_{m}^{b_k}, k = 1 : K\) of scribbles;
2. Define an adjacency matrix \(W = [w_{ij}]_{N \times N}\) with 8-neighbors graph structure by (1);
3. Generate the GMMs with five components from all scribbles and then get the probability density \(u_k\);
4. Scale probability: \(u_k = \max(100 + \log(u_k), 10^{-10})\);
5. If \(\gamma = 1\) then
6. Obtain coarse segmentation result \(C\bar{R}_i = \arg \max_k u_k^c\);
7. Set \(g_k = g_k^c\), where \(g_k^c = u_k^c + \max_{k \neq k} u_k^c\);
8. Compute the transition probability matrix \(\bar{P}\) and the vector \(\bar{u}_k\) by (37) and (38);
9. Solve linear equations: \(\bar{E}\bar{r}_k = (I - D_c) \bar{u}_k + D_c b_{m}^{b_k}\);
10. Normalize the reaching probabilities: \(\bar{r}_k = \frac{\bar{r}_k}{\bar{r}_k^T}\);
11. Obtain segmentation result \(\bar{R}_i = \arg \max_k \bar{r}_k\);

The vector formulation is as follows:
\[
\bar{\Omega}_k = \frac{1}{2} \bar{r}_m^T (D - W) \bar{r}_k + \sum_k T_k \frac{1}{2} \left( (\bar{r}_m - b_{m})^T (I - D_c) \bar{r}_m - \bar{r}_m \right) + \frac{1}{2} \left( (\bar{r}_m - \bar{e})^T \bar{D}_u (\bar{r}_m - \bar{e}) + \bar{D}_g \bar{r}_m \right),
\]
(42)
where \(\bar{D}_g = \text{diag}(g_1, g_2, \ldots, g_N)\) is a diagonal matrix, \(\bar{D}_u^k = \text{diag}(u_1^k, u_2^k, \ldots, u_N^k)\) is also a diagonal matrix, and \(\bar{e} = [1, 1, \ldots, 1]^T\) is an \(N \times 1\) vector. We set \(\bar{D}_a = D + \lambda \bar{D}_g\) and \(\bar{D}_u = I - D_c\).

By taking the partial derivative of \(\bar{\Omega}_k\), we have:
\[
\frac{\partial \bar{\Omega}_k}{\partial \bar{r}_k^T} = (D - W \bar{r}_k + D \bar{D}_a (\bar{r}_k - b_k^k))
\]
(43)

\[
\begin{align*}
&= m \begin{bmatrix}
\lambda & -1 & \cdots & -1 \\
\vdots & \ddots & \ddots & \vdots \\
-1 & \cdots & \lambda & -1 \\
-1 & \cdots & -1 & \lambda \\
& \vdots & \ddots & \ddots & \vdots \\
\lambda & -1 & \cdots & -1
\end{bmatrix}
\end{align*}
\]
(44)

**B. The Optimization Explanation**

Similar to the subRW, we can also give the optimization
\[\eta \sum_k \eta_A \eta_C (\eta_I - 1) + \sum_{i=1}^2 \lambda u_i (\eta^I - 1) + \sum_{t=1,t=k}^2 \lambda u_i \eta^I + \sum_{i=1}^2 \lambda u_i \eta^I.\]
From (41), we can find that it consists of three components. The first two components are the smooth term and the unary term, which are similar to the components in (10). The last component corresponds to the label prior. By minimizing this component, the reaching probability $\tilde{r}_{lm}^{k}$ will be consistent with the label prior. Note that, if we only consider the label prior, the reaching probability $\tilde{r}_{lm}^{k}$ should satisfy the following proposition.

**Proposition 1:** If the probability density $u_{l}^{k}$ of label $l$ is larger than the probability densities of any other labels, i.e., $u_{l}^{k} > u_{l}^{i}$, then the corresponding reaching probability $\tilde{r}_{lm}^{i}$ should be close to 1.

However, from the last component of (41), we can find that only when $u_{l}^{k} > u_{l}^{i}$, $\tilde{r}_{lm}^{k}$ will be close to 1. When $K > 2$, this situation is not expected. Thus, a proper adjustment is made to correct this problem. Denote $g_{l}^{k} = u_{l}^{k} + \max_{i \neq k} u_{l}^{i}$ and set $g_{l} = g_{l}^{k}$ for each label, then the last component of (41) will become:

$$\frac{1}{\lambda} \sum_{l=1}^{N} \sum_{i} \frac{(\tilde{r}_{lm}^{k} - 1)^{2}}{g_{l}^{k}} + \frac{1}{\lambda} \sum_{l=1}^{N} \sum_{i \neq k} \frac{(\max_{i \neq k} u_{l}^{i} \tilde{r}_{lm}^{i})^{2}}{g_{l}^{k}}.$$  \hspace{1cm} (44)

By minimizing this component, $\tilde{r}_{lm}^{k}$ will satisfy **Proposition 1.** In practice, the above adjustment is made by setting $g_{l} = g_{l}^{k}$ in (39). More details are shown in Algorithm 1.

**C. Noise Reduction**

Adding the label prior may produce some noise. One solution is decreasing parameter $\lambda$. However, when $\lambda$ is too small, the twig parts may be lost. Then we need to use the other strategy to reduce noise. Combining the label prior value for each node, we can get a coarse segmentation result:

$$C R_{l} = \arg \max_{k} u_{l}^{k}.$$ \hspace{1cm} (45)

Although there is much noise in this coarse segmentation, it mostly does not connect with the main part. So we can choose the connected regions with seeds from the coarse segmentation as the candidate regions. Then we only add the label prior into these candidate regions, which will keep the prior information of twigs and decrease the noise. Furthermore, the candidate regions are dilated to add more prior information into the boundary regions. Since the contrast of the prior information near boundaries is high, it will help to find correct boundaries. We denote the indicating vector of the dilated candidate regions as $c r_{l}^{k} = [c r_{l}^{k}]_{N \times 1}$, i.e., $c r_{l}^{k} = 1$, if node $i$ belongs to these regions; $c r_{l}^{k}_{i} = 0$, otherwise. Then the prior vector to the object. Both RWR and LRW reduce the probability that a random walker walks on the original image graph, which thus reduces the influence of these barriers. Then the random walker in RWR or LRW will be more likely to find the correct boundaries. Fig. 4 (c) and (d) show the similar results by RWR and LRW, since they can be viewed as similar subRWRs without label prior. In the top row, they perform well and only lose some twigs and produce some error boundaries. However, in the bottom row, they only detect very few right boundaries. This may be due to that the twigs are too long and complicated. Our algorithm still performs very well for this complicated situation. Our algorithm also reduces the influence of barriers by decreasing the walking probability on.
\[ u^k = [u^k_i]_{N \times 1} \] is reset as: \[ u^k \leftarrow u^k \cdot c r^k \], where \( c \) means the corresponding elements of two vectors are multiplied respectively and a vector is returned with the same scale. For some images with a high contrast between the object and the background, it is not necessary to make this noise removal process. So we offer a selecting parameter \( \gamma \) for users to choose whether to perform this process.

Finally, we summarize the main steps of the proposed subMarkov random walk with label prior in Algorithm 1.

We have also compared our algorithm with the other three well-known RW algorithms for natural images shown in Fig. 5 and Fig. 6. These images are taken from three datasets: the Berkeley segmentation dataset (BSD) [9], the
Microsoft Research Cambridge object recognition image database (MSRC), and Visual Object Classes Challenge 2012 (VOC2012). The manual labeled ground-truth masks are provided in these three datasets, and the VOC2012 dataset only offers ground-truth masks of a part of images. However, the masks in MSRC and VOC2012 are not accurate enough or complete for our experiments, and the VOC2012 does not offer the masks of background. Thus, we only used the ground truth masks from BSD and labeled the new ground truth masks for a part of images from MSRC and VOC2012 ourselves.

Two metrics are used for the quantitative comparisons. The first one is a normalized overlap score $a_0$ [23] which measures the accuracy of the object segmentation results. The overlap region is formulated as: $a_0 = \frac{\hat{R} \cap \hat{G}}{\hat{R} \cup \hat{G}}$, where $\hat{R}$ is the set of pixels assigned as the object from the segmentation result and $\hat{G}$ is that from the ground-truth.

A higher score indicates better segmentation performance. This metric is effective for objective evaluation. However, it cannot fully reflect the evaluation mechanism of human visual system. It may ignore the connectivity or the coherence along the segmented boundaries. This limitation has been shown
in [39]. To further verify the performance of our method, we adopt the second metric: the percentage of incorrectly labeled pixels, which is also called error rate. Let us denote

\[ L_A = [l_A^1, \cdots, l_A^N]^T \] and \[ L_B = [l_B^1, \cdots, l_B^N]^T \] as the sets of labels from the segmentation results and from the ground-truth. Then, the is defined as follows:

\[ \text{error rate} = \frac{\sum_{i=1}^{N} I_A^i \oplus I_B^i}{N} \times 100, \]

where \( I_A^i \oplus I_B^i = 1 \) if \( l_A^i = l_B^i \); \( I_A^i \oplus I_B^i = 0 \), otherwise.

We evaluated our algorithm and other three algorithms on two segmentation problems. The first one is foreground/background segmentation, i.e., 2-label segmentation, which is the most common segmentation problem. And the second is multi-label segmentation. Both metrics are applied for measuring the quantitative performance. But the main metric is chosen by different segmentation problems. The overlap score \( a_0 \) measures the accuracy of the object, which is more suitable for the 2-label segmentation. In 2-label segmentation, more attention may be attracted by the foreground, i.e., the object in an image. The error rate indicates the segmentation performance of all labels (not only considering the object), which is more suitable for the multi-label segmentation. Thus, the overlap score \( a_0 \), as the main metric of 2-label segmentation, is directly shown in the qualitative results in Fig. 5. Similarly, the error rate is also directly shown in Fig. 6. Next, we will describe these experiments in detail.

In 2-label segmentation, we choose the natural images that only include one object with twigs. Some of them have complex textures like the cheetah, and others possess very thin twigs such as the helicopter. Fig. 5 shows that our algorithm outperforms the other algorithms no matter in quality or in quantity. In the qualitative comparisons, it is obvious that our method not only successfully segments out the most twigs of objects, but also adheres the edges to the boundaries better than the others. This is due to the added label prior which also has an impact on the main part of the object. In the

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>airplane</td>
<td>6.68%</td>
<td>5.97%</td>
<td>5.94%</td>
<td>0.90%</td>
</tr>
<tr>
<td>bird</td>
<td>2.66%</td>
<td>2.66%</td>
<td>3.71%</td>
<td>1.09%</td>
</tr>
<tr>
<td>elk</td>
<td>3.33%</td>
<td>2.64%</td>
<td>3.33%</td>
<td>0.24%</td>
</tr>
<tr>
<td>h t h</td>
<td>7.90%</td>
<td>6.36%</td>
<td>8.23%</td>
<td>2.71%</td>
</tr>
<tr>
<td>bon</td>
<td>2.82%</td>
<td>4.19%</td>
<td>6.52%</td>
<td>1.17%</td>
</tr>
</tbody>
</table>

Average time (s) 0.69 2.12 2.11 2.41
rate in Fig. 6 and the average overlap score in Table II also show the significant improvement of our algorithm. Note that,
the average overlap score is the average of all objects' overlap scores in one image. It is worth mentioning that the label prior of GMM may produce more noise than 2-label segmentation. If we do not reduce the noise (i.e., setting $\gamma = 0$), the segmentation results will be bad or even completely wrong. Thus, we reduce the noise by setting $\gamma = 1$ for most images. As mentioned before, this process may lose some twigs such as the airplane2 in Fig. 6. However, our algorithm still has better performance.

We also compare the run time among these algorithms. All the average run time in Tables I and II are measured in seconds with our unoptimized Matlab implementation on a PC with Intel Core Processor i7-2600 CPU @3.40GHz with 6GB RAM. Our approach is slightly slower than RWR and LRW, since our method additionally builds label prior for achieving better performance, which needs extra time to construct GMMs. However, as shown in Fig. 6, our subMarkov RW algorithm significantly improve the seeded image segmentation performance. The additional computational cost is worthy to obtain better segmentation performance using our subMarkov RW algorithm with Label Prior.

## V. Conclusions

We have presented a novel framework based on the subMarkov random walk for interactive seeded image segmentation in this work. This framework can be explained as a traditional random walker that walks on the graph by adding some new auxiliary nodes, which makes our framework easily interpreted and more flexible. Under this framework, we unify the well-known RW-based algorithms, which satisfy the subMarkov property and build bridges to make it easy to transform the findings between them. Furthermore, we have designed a novel subRW with label prior to solve the twigs segmentation problems by adding prior nodes into our framework. The experimental results have shown that our algorithm outperforms the state-of-the-art RW-based algorithms. This also proves that it is practicable to design a new subRW algorithm by adding new auxiliary nodes into our framework. In the future, we will extend our algorithm to more applications, such as centerline detection at 3D medical images [42] and

<table>
<thead>
<tr>
<th></th>
<th>RWR [27]</th>
<th>LRW [40]</th>
<th>ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>j 1 j 1</td>
<td>0.7317</td>
<td>0.5946</td>
<td>0.7973</td>
</tr>
<tr>
<td>b 1 b 2</td>
<td>0.7855</td>
<td>0.7168</td>
<td>0.9184</td>
</tr>
<tr>
<td>cow</td>
<td>0.7563</td>
<td>1.23%</td>
<td>0  5</td>
</tr>
<tr>
<td>airplane</td>
<td>0.8642</td>
<td>2.87%</td>
<td>0.9232</td>
</tr>
<tr>
<td>plant</td>
<td>0.7863</td>
<td>5.40%</td>
<td>0.934</td>
</tr>
<tr>
<td>bike</td>
<td>0.0708</td>
<td>8.22%</td>
<td>0.7424</td>
</tr>
<tr>
<td>chair</td>
<td>0.5864</td>
<td>6.52%</td>
<td>0.7125</td>
</tr>
</tbody>
</table>

Average time (s) 1.51 2.57

### A. The Proof of the Matrix $E$ in (8) Is Nonsingular

Before the proof, we give some definitions and an important theorem from http://en.wikipedia.org/wiki/ Diagonally_dominant_matrix.

**Definition 2**: The matrix $A$ is diagonally dominant if

$$|a_{ii}| \geq \sum_{j \neq i} |a_{ij}| \quad \text{for all } i,$$

where $a_{ij}$ denotes the entry in the $i$th row and $j$th column.

**Definition 3**: If a strict inequality $(>)$ is used in (46), then this matrix is said to be strictly diagonal dominant.

**Theorem 1**: A strictly diagonally dominant matrix (or an irreducibly diagonally dominant matrix) is non-singular, which is known as the Levy-Desplanques theorem [2].

**Proof**: According to **Theorem 1**, we only need to prove the matrix $E = I - (I - D_c)e$ is strictly diagonal dominant. We first give the formulation of each element in $E$ as:

$$e_{ij} = \begin{cases} 1, & i = j, \frac{(1 - c_i)w_{ij}}{d_i}, & j \sim i, \frac{1}{d_i}, & j \sim i, \end{cases}$$

Thus, if $0 < c_i < 1$, then $\forall i$,

$$|e_{ij}| = \begin{cases} 1, & i = j, \frac{1 - c_i}{d_i}, & j \sim i, \end{cases}$$

Therefore, $|e_{ii}| = 1 - c_i < 1 = |e_{ii}|$.

So according to **Definition 3**, the matrix $E$ is strictly diagonal dominant.

### B. Unique Solution of (39)

**Proof**: If the coefficient matrix $\bar{E} = I - (I - D_c)e$ is non-singular, then the equivalent linear equations of (39) have unique solution, i.e., the solution of (39) is unique. So we only need to prove the matrix $\bar{E}$ is non-singular. Similar to the above proof, we should prove $\bar{E}$ is strictly diagonal dominant. We rewrite this matrix as follows:

$$\bar{e}_{ij} = \begin{cases} 1, & i = j, \frac{(1 - c_i)w_{ij}}{d_i}, & j \sim i, \frac{d_i + \lambda g_i}{d_i}, & j \sim i, \end{cases}$$

Thus, if $0 < c_i < 1$, then $\forall i$,

$$|\bar{e}_{ij}| = \begin{cases} 1, & i = j, \frac{1 - c_i}{d_i}, & j \sim i. \end{cases}$$
classification [41].

APPENDIX A
IMPLEMENTATION DETAILS

In this section, we describe our exact formulations with more detailed explanations for completeness. According to Definition 3, the matrix \( \bar{E} \) is also strictly diagonal dominant.

\[
\begin{align*}
1 &= \sum_{j \neq i} \frac{d_i + \lambda g_i}{d_i} \\
&= (1 - c_i) d_i + \lambda g_i < 1 = |\bar{e}_{ii}| \\
\end{align*}
\] (50)