Submodular Function Optimization for Motion Clustering and Image Segmentation

Jianbing Shen, *Senior Member, IEEE*, Xingping Dong, Jianteng Peng, Xiaogang Jin, *Member, IEEE*, Ling Shao, *Senior Member, IEEE*, and Fatih Porikli, *Fellow, IEEE*

***Abstract*—In this paper, we propose a framework of maximizing quadratic submodular energy with a knapsack constraint approximately, to solve certain computer vision problems. The proposed submodular maximization problem can be viewed as a generalization of the classic 0/1 knapsack problem. Importantly, maximization of our knapsack constrained submodular energy function can be solved via dynamic programming. We further introduce a range-reduction step prior to dynamic programming as a two-stage procedure for more efficient maximization. In order to demonstrate the effectiveness of the proposed energy function and its maximization algorithm, we apply it to two representative computer vision tasks: image segmentation and motion trajectory clustering. Experimental results of image segmentation demonstrate that our method out-performs the classic segmentation algorithms of graph cuts and random walks. Moreover, our framework achieves better performance than state-of-the-art methods on the motion trajectory clustering task.**

***Index Terms*—Submodular maximization, knapsack constraint, segmentation, trajectory clustering.**

I. Introduction

Submodular function optimization is a fundamental problem in discrete optimization, and has attracted increasing attention over the last decade due to its popularity in the field of artificial intelligence [21], [2], [46], [50]. Many well-studied problems, such as non-parametric learning, kernel machines, active learning and information gathering, involve objective functions that are submodular. A set function $f : 2^V \rightarrow \mathbb{R}$ is said to be submodular [11] if, and only if, it satisfies

$$f(S) + f(T) \geq f(S \cap T) + f(S \cup T) \tag{1}$$

for all subsets $S, T \subseteq V$. The expression $f(v \mid S) = f(\{v\} \cup S) - f(S)$ denotes the marginal gain in adding $v \in V - S$ to the subset S. Submodular functions satisfy the property of diminishing returns:

$$f(v \mid S) \geq f(v \mid T), \quad for\ all\ S \subseteq T \subseteq V. \tag{2}$$

Besides diminishing returns, more and more properties of submodular functions have been discovered gradually to help optimize these important functions. It has been proven to be able to get the global minimum of submodular function without constraints in linear time complexity. And other algorithm [22], [25] focus on improving its efficiency.

However, maximizing a submodular energy with or without constraints is an NP-hard problem. Recently, a growing number of vision applications have adopted submodular maximization, e.g. object detection [33], [17], video summarization [42], [43] and image saliency [27], [40]. In these works, the problem is formulated as choosing a subset S from the ground set V with the goal of maximizing some submodular utility function $F(S)$. This kind of problem can be treated as submodular maximization with a constraint, using a general form of

$$\max_{S \subseteq V} F(S), \quad \text{s.t.} \ \ g(S) \leq K \tag{3}$$

where the total cost $g(S)$ is kept to be below some ceiling K.

For object detection [33], [17], the set S represents the selected objects, and the set V stands for all the image proposals. For video summarization [42], [43], S is a sequence of key frames and V is the set of all the sub-clips of the video. Jiang *et al.* [27] and Zhong *et al.* [40] adopted the submodular maximization to model the image saliency problem and obtain a set of representative superpixels. Then, the saliency values of these superpixels are transferred to the whole image.

Submodular maximization has been applied for numerous tasks in computer vision, and we have further found that other vision applications such as interactive segmentation and motion trajectories clustering can also be solved with this model. Generally, for submodular maximization problems, the objective functions can be monotone (e.g. coverage and entropy functions) or non-monotone (graph cuts and mutual information). Among them, facility location [27], [33], [40] and information theoretic [38], [31], [41], [42], are two commonly used models of submodular maximization in the field of image processing and computer vision.

The facility location model is usually formulated as:

$$\begin{aligned} &\max \sum_{i \in V} \max_{j \in S} \omega_{ij} - \sum_{i \in S} c(i) \\ &\text{s.t.} \ \ S \subseteq V, \ |S| \leq K \end{aligned} \tag{4}$$

This work was supported in part by the Beijing Natural Science Foundation under Grant 4182056, the Key Research and Development Program of Zhejiang Province under grant 2018C03055, the National Natural Science Foundation of China under grant grant 61732015, the Australian Research Council's Discovery Projects funding scheme under grant DP150104645, and the Fok Ying-Tong Education Foundation for Young Teachers. Specialized Fund for Joint Building Program of Beijing Municipal Education Commission. (Corresponding author: *Jianbing Shen*)

J. Shen, X. Dong, and J. Peng are with the Beijing Laboratory of Intelligent Information Technology, School of Computer Science, Beijing Institute of Technology, Beijing 100081, P. R. China. (email: shenjianbing@bit.edu.cn)

X. Jin are with State Key Lab of CAD&CG, Zhejiang University, Hangzhou, 310058, P. R. China. (email: jin@cad.zju.edu.cn)

L. Shao is with the Inception Institute of Artificial Intelligence, Abu Dhabi, United Arab Emirates. (email: ling.shao@ieee.org)

F. Porikli is with the Research School of Engineering, Australian National University, Canberra, ACT 0200, Australia. (email: fatih.porikli@anu.edu.au)

TABLE I
THE RESTRICTIONS OF THREE SUBMODULAR MAXIMIZATION MODELS.

	non-monotone	cardinality constraint	knapsack constraint
Facility location model (4)	✓	✓	×
Information theoretic model $H(.)$ (5)	×	✓	×
Information theoretic model $M(.)$ (5)	✓	✓	×
Our model	✓	✓	✓

where ω_{ij} is the affinity between two elements i and j in the ground set V, and $c(i)$ is the cost of element i. The constraint $|S| \leq K$ enforces that the selected number of elements is less than a fixed constant K. The first term of the above objective function encourages to find a subset S which can represent all elements as completely as possible; the second term requires the total cost of S be small.

The information theoretic approach is formulated as:

$$\begin{aligned} &\max l(H(S)) \quad \text{or} \quad \max l(M(S, V\backslash S)) \\ &\text{s.t.} \;\; S \subseteq V \;,\; |S| \leq K \end{aligned} \tag{5}$$

where $l(\cdot)$ is a non-decreasing function; $H(\cdot)$ and $M(\cdot)$ represent entropy rate and mutual information, respectively. The formulations of the entropy rate and mutual information are provided in **Appendix I** according to [38], [42]. Using a submodular function, its property of diminishing returns, many researchers [31], [38], [41], [42] have proven the objective functions in (4) and (5) are both submodular.

We propose a new framework of submodular maximization, which enriches the practicability of submodular maximization in computer vision. In the above formulations, both models enforce simple cardinality constraints. Differently, the proposed submodular maximization framework contains a more general constraint: knapsack constraint. Specifically, we show how this new constraint can be used to solve two important vision problems: image segmentation and motion trajectory clustering. In the application of image segmentation, our submodular maximization model is used to select foreground superpixels. While in the motion trajectory clustering application, our submodular maximization model can choose representative trajectories with good quality as clustering centers. Our knapsack constraint in these applications can obtain better performance, since it adds more prior information explicitly. The advantages of our new approach compared to the existing models are summarized in Table *I*.

For convenience, a set of submodular function can be treated as a function over n-dimensional binary column vector variables $\mathbf{x} \in \{0,1\}^n$. Differing from existing formulations (4) and (5), the proposed submodular energy and its constraint are defined as:

$$F(\mathbf{x}) = U\mathbf{x} - \mathbf{x}^T P\mathbf{x}, \quad g(\mathbf{x}) = C\mathbf{x} \tag{6}$$

where U and P are restricted to $\mathbb{Z}+^{1\times n}$ and $\mathbb{N}^{n\times n}$ for numerical optimization but could be in $\mathbb{R}+$ in theory, and $C \in \mathbb{N}^{1\times n}$. The constraint $g(\mathbf{x}) \leq K$ is a knapsack constraint and more general than the cardinality constraint of the previous facility location and information theoretic models. Specifically, we set C as an all-one vector, and the knapsack constraint $C\mathbf{x} \leq K$ is degenerated into a cardinality constraint $||\mathbf{x}||_0 \leq K$.

A generic method for approximately maximizing constrained submodular functions is the greedy algorithm [1], [3]. The greedy algorithm always has performance guarantee $1-1/e$, when the objective submodular function is monotone. The knapsack constraint $g(\mathbf{x}) \leq K$ subsumes the cardinality constraint, i.e., setting $g(\mathbf{x}) = 1^T\mathbf{x}$ recovers the cardinality constraint. Some algorithms [26], [7] are designed specifically for submodular maximization with knapsack constraints. However, the objective function $f(\mathbf{x})$ in these methods must be a polymatroid function, which is monotone and non-decreasing. In order to solve more interesting vision problems, we develop a new method to maximize submodular functions of (6) by dynamic programming. Our framework includes more general constraints than the cardinality constraint, and can maximize both monotone and non-monotone quadratic objective functions. Additionally, our optimizing method obtains more accurate solutions than the greedy algorithm.

In the context of previous work on submodular maximization, our main contributions are summarized as follows:

- A new framework of submodular functions with a more general knapsack constraint is presented. This framework is different from the facility location and entropy model, which can deal with both monotone and non-monotone quadratic objective functions and thus enriches the field of submodular optimization in computer vision.
- A novel submodular maximization method based on dynamic programming is proposed. Experimental results show that our new optimization method can outperform the classic greedy algorithm.
- Our new framework is applied to two computer vision tasks: image segmentation and video trajectory clustering. Experimental results demonstrate that our new framework can achieve competitive performance.

II. RELATED WORK

In this section, we review some existing work related to submodular optimization and its applications. The popular optimization method for maximizing a submodular function with a constraint is the greedy algorithm [1], [3]. The greedy algorithm performs iteratively, where each element in the solution vector $\mathbf{x}$ is set to 0 initially. In each iteration, only one element of $\mathbf{x}$ is given label 1 so as to obtain the largest energy marginal gain, while keeping the total cost under its limit K. The greedy algorithm can support arbitrary constraints, such as the case where the constraints satisfy a matroid.

Different from the greedy algorithm, more approaches aim at maximizing submodular energy with a specific form or limitation. Calinescu *et al.* [15] proposed a constrained submodular maximization method with a higher accuracy than the greedy

algorithm for a positive non-decreasing submodular function. A submodular function f is positive, if $f : 2^N \rightarrow \mathbb{R}+$. Stochastic-Greedy [30] is a linear-time algorithm for maximizing a general monotone submodular function subject to a cardinality constraint. Based on the greedy algorithm, some methods focused on achieving better performance on large-scale data. For example, Wei *et al.* [35] designed a multi-stage algorithm to yield 1000 times speedup. However, these methods required the submodular energy to be monotone and subject to cardinality constraints. Feige *et al.* [16] proposed the constant-factor approximation algorithm for maximizing non-negative submodular functions. In particular, they design a deterministic local search with 1/3-approximation and a randomized 2/5-approximation algorithm. Caprara *et al.* [4] obtained the exact solution of the quadratic knapsack problem. However, it requires the coefficients of pairwise terms in the objective functions should be non-negative, which means the energy is not submodular. Sviridenko *et al.* [9] maximized the non-decreasing submodular energy. Both [23] and [19] focus on maximizing the non-monotone submodular function with knapsack constraints. However, these methods assume the function is non-negative. Our method can deal with negative and non-negative quadratic submodular objective functions, which can be non-monotone.

The proposed method also needs the energy and the constraint to satisfy specific forms in (6). However, our solution is more accurate than the greedy algorithm. Unlike others, our method is not based on the greedy algorithm, the stochastic form or continuous transformation. We employ dynamic programming to maximize the submodular energy functions to enrich the optimization methods of submodular functions.

As for the applications of submodular functions in computer vision, an increasing number of researchers are working on this field with a huge progress. Some algorithms adopt the facility location to formulate the problem. Jiang *et al.* [27] and Zhong *et al.* [40] used submodular maximization to obtain image saliency. They build a facility location form energy to select the superpixels which are easy to determine their accurate saliency values. Then precise saliency values of those superpixels are obtained and diffused to all pixels of the whole image according to the appearance model. Zhu *et al.* [33] designed a submodular function to perform object detection.

Other important works solve submodular optimization by different methods. Xu *et al.* [42] presented a gaze-enabled egocentric video summarization method by using the constrained submodular maximization algorithm. Yang *et al.* [41] measured the entropy gains of images to model the retrieval problem. Liu *et al.* [38] designed a submodular energy function with entropy rate, and maximize it to extract superpixels by clustering pixels. Qian *et al.* [50] applied a multi-objective evolutionary algorithm to maximize monotone k-submodular functions. Balkanski *et al.* [46] explored the problem of minimizing a submodular function from training data. They show some submodular functions cannot be minimized when given access to the polynomially-many samples.

Our new quadratic submodular maximization framework is further adopted to solve two vision problems: image segmentation with user interaction and motion trajectory clustering for videos. Some basic Interactive segmentation methods, including graph cuts [10] and random walks [14], have shown their effectiveness. By contrast, motion trajectory clustering for videos is a relative new problem. Trajectories can describe the motion information of all main objects in videos. Therefore, it is important to obtain their clusters for modeling all moving entities in the video. Many popular methods such as [18], [28], [36], [45] have been proposed to solve this problem.

III. Our approach

In this section, we first introduce the typical 0/1 knapsack problem and its extension in Section III-A. Then, in Section III-B, we generalize the knapsack problem to our new model for maximizing a quadratic submodular energy with a knapsack constraint. And we design its optimization method based on dynamic programming. Finally, we adopt a solution range reduction process to accelerate our algorithm in Section III-C.

A. Typical 0/1 knapsack problem and its extension

First, we introduce the typical 0/1 knapsack problem. Assume we have a set of n items and a knapsack with a limited carrying capacity M. Each item i has mass m_i and value v_i. We need to find the collection of items with a maximum total value while keeping the total weight no larger than the given limit M. Define a vector $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T \in \{0,1\}^n$, where $x_i = 1$ indicates that item i is included in the collection and $x_i = 0$ indicates that it is excluded. Then, the 0/1 knapsack problem can be formulated as:

$$\begin{aligned} &\mathbf{x}^* = \arg\max_{\mathbf{x}} \quad F(\mathbf{x}) \\ &\text{s.t.} \;\; g(\mathbf{x}) \leq M \end{aligned} \tag{7}$$

$$\begin{aligned} &F(\mathbf{x}) = v_1x_1 + v_2x_2 + \cdots + v_nx_n \\ &g(\mathbf{x}) = m_1x_1 + m_2x_2 + \cdots + m_nx_n \\ &v_i > 0, \;\; m_i > 0, \;\; i = 1, 2, \ldots, n \end{aligned} \tag{8}$$

where $\mathbf{x}^*$ is the optimal solution. The typical 0/1 knapsack problem is NP-hard. An effective method for solving the 0/1 knapsack problem is dynamic programming. Because the function $g(\mathbf{x})$ is linear, the problem in (7) is actually submodular maximization with a knapsack constraint. Inspired by the above analysis and the nonserial dynamic programming in [8], we will solve a more general problem of maximizing a quadratic submodular energy subject to a knapsack constraint approximated by dynamic programming in section III-B.

Let us expend the objective submodular function $F(\mathbf{x})$ to the following form, and then adopt the dynamic programming to maximize it as follows:

$$F(\mathbf{x}) = \sum_{i=1}^{n} v_i x_i - \sum_{i=1}^{n-1} s_i x_i x_{i+1} \tag{9}$$

where s_i is a coefficient.

According to the property of diminishing returns, if a submodular function contains quadratic terms, their coefficients must be non-positive. Therefore, we have $s_i \geq 0$ in (9). According to the definition of dynamic programming, we need to build the subproblem of this maximization problem. Let

$B_i[x_i, M_i]$ denote the maximum energy of the best partial assignment of $\{x_1, \ldots, x_{i-1}\}$, when the value of x_i is fixed and the constraint $g(x_1, \ldots, x_i) \leq M_i$ is satisfied. This can be formulated as:

$$\begin{aligned} &B_i[x_i, M_i] = \max_{Y} F(x_1, \ldots, x_{i-1}, x_i) \\ &\text{s.t.} \quad g(x_1, \ldots, x_{i-1}, x_i) \leq M_i \\ &Y = \{x_1, \ldots, x_{i-1}, x_i\} \backslash \{x_i\} \end{aligned} \tag{10}$$

That is, $B_n[x_n, M_n] = \max_{x_n}\{F(\mathbf{x}) : g(\mathbf{x}) \leq M\}$, and $M_n = M$. Hence, the dynamic programming recursion becomes:

$$\begin{aligned} B_i[x_i, M_i] &= \begin{cases} \psi_i(x_i, M_i), & \text{if } m_i x_i \leq M_i \\ -\infty, & \text{otherwise} \end{cases} \\ \psi_i(x_i, M_i) &= v_i x_i + \max_{x_{i-1}} \Big\{ B_{i-1}[x_{i-1}, M_{i-1}] \\ &\qquad - s_{i-1} x_i x_{i-1} \Big\} \\ M_{i-1} &= M_i - m_i x_i \end{aligned} \tag{11}$$

The smallest sub-problem can be directly solved as:

$$B_1[x_1, M_1] = \begin{cases} v_1 x_1, & \text{if } m_1 x_1 \leq M_1 \\ -\infty, & \text{otherwise} \end{cases} \tag{12}$$

The optimal solution $\mathbf{x}^*$ is obtained by the following iteration, and we get the value of x_i after determining the value of x_{i+1}:

$$\begin{aligned} x_n^* &= \arg\max_{x_n} B_n[x_n, M] \\ x_i^* &= \arg\max_{x_i} \Big\{ B_i[x_i, M_i] - s_i x_i x_{i+1} \Big\}, \quad \text{for } i < n \end{aligned} \tag{13}$$

B. Quadratic submodular maximization with a knapsack constraint

We further generalize the objective function $F(\mathbf{x})$ and constraint function $g(\mathbf{x})$ in (7) to:

$$\begin{aligned} F(\mathbf{x}) &= \sum_{i=1}^{n} v_i x_i - \sum_{1 \leq i < j \leq n} s_{ij} x_i x_j \\ g(\mathbf{x}) &= \sum_{i=1}^{n} m_i x_i \\ &v_i > 0, \quad m_i > 0, \quad s_{ij} \geq 0 \end{aligned} \tag{14}$$

This maximization problem is no longer a knapsack problem. It can be rewritten in a compact matrix form as

$$\begin{aligned} &\mathbf{x}^* = \arg\max_{\mathbf{x}} \quad U\mathbf{x} - \mathbf{x}^T P \mathbf{x}, \\ &\text{s.t.} \quad C\mathbf{x} \leq M \end{aligned} \tag{15}$$

where $U = [v_1, v_2, \ldots, v_n]$, $C = [m_1, m_2, \ldots, m_n]$, and the element (i, j) of matrix P is s_{ij}. (15) has the same pattern as we put forward in (6). And all the elements in vector/matrix U, P, c and M are non-negative integers. The proofs for the submodularity of function $F(\mathbf{x})$ in (8), (9) and (14) are given in **Appendix II**.

We can build a graph to abstract a quadratic submodular energy. Each variable x_i can be treated as a vertex and quadratic term $s_{ij} x_i x_j$ as an edge connecting vertexes i and j with a weight s_{ij}. The extracted graph of (9) is a line (see Fig. 1(b)), and we can easily find a sequence of vertexes to build the subproblem for dynamic programming. However, using the same process, the related graph of (14) is more generic and likely to contain cycles. We overcome this obstacle by redefining a more general recurrence formula. Before this, we need to extend the definition of $B_i[x_i, M_i]$ in (10) with a new representation $\mathcal{B}_i[E, M_i]$

$$\begin{aligned} &\mathcal{B}_i[E, M_i] = \max_{Y} F(x_1, \ldots, x_i) \\ &\text{s.t.} \quad g(x_1, \ldots, x_i) \leq M_i \\ &Y = \{x_1, \ldots, x_i\} \backslash E \end{aligned} \tag{16}$$

where E is a variable subset of $\{x_1, \ldots, x_i\}$. $\mathcal{B}_i[E, M_i]$ represents the maximum value of best assigned labels of the first i items, where the values of variables in set E are fixed.

Finally, we design the new recurrence formula as follows:

$$\begin{aligned} \mathcal{B}_i[\{x_i\}, M_i] &= \begin{cases} \psi_i'(x_i, M_i), & \text{if } m_i x_i \leq M_i \\ -\infty, & \text{otherwise} \end{cases} \\ \psi_i'(x_i, M_i) &= v_i x_i + \max_{\mathcal{N}(x_i)} \Big\{ \mathcal{B}_{i-1}[\mathcal{N}(x_i), M_{i-1}] \\ &\qquad - \sum_{x_j \in \mathcal{N}(x_i)} s_{ij} x_i x_j \Big\} \\ M_{i-1} &= M_i - m_{i-1} x_{i-1} \\ \mathcal{N}(x_i) &= \{x_j \mid j < i, s_{ij} > 0\} \end{aligned} \tag{17}$$

where $\mathcal{N}(x_i)$ is a set of variables. It contains all variables with a smaller subscript than i and existing pairwise terms with x_i. $\mathcal{B}_{i-1}[\mathcal{N}(x_i), M_{i-1}]$ represents the maximum value of the best assignment of labels of the first $i-1$ items, where the values of variables in $\mathcal{N}(x_i)$ are fixed. Mathematically, we have

$$\begin{aligned} &\mathcal{B}_{i-1}[\mathcal{N}(x_i), M_{i-1}] = \max_{Y} F(x_1, \ldots, x_{i-1}) \\ &\text{s.t.} \quad g(x_1, \ldots, x_{i-1}) \leq M_{i-1} \\ &Y = \{x_1, \ldots, x_{i-1}\} \setminus \mathcal{N}(x_i) \end{aligned} \tag{18}$$

The main motivation of using quadratic submodular energy function is that it can represent more computer vision problem than (8) and (9). Energy in (8) only contains unary terms, however, most of vision problems need quadratic terms to model. The quadratic terms in (9) are limited, since the variables need to be i and $i+1$. The proposed quadratic submodular function has extended these two limitations, which can model more vision problems well.

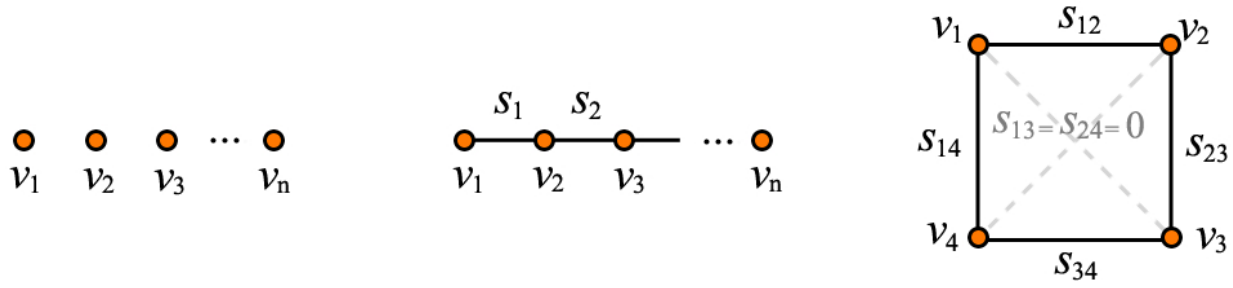
Fig. 1(c) gives an example graph, which is extracted from an objective function with four variables $F(x_1, x_2, x_3, x_4)$:

$$\begin{aligned} &F(x_1, x_2, x_3, x_4) = U \cdot [x_1, x_2, x_3, x_4]^T - \\ &\{s_{12} x_1 x_2 + s_{23} x_2 x_3 + s_{34} x_3 x_4 + s_{14} x_1 x_4\} \end{aligned} \tag{19}$$

The unary parameter U does not relate to building the graph, and it does not contain specific meaning in this equation and can be formed by arbitrary non-negative integers.

According to (17), we have:

$$\begin{aligned} &\mathcal{N}(x_4) = \{x_1, x_3\} \\ &\psi_4'(x_4, M) = v_4 x_4 + \max_{x_1, x_3} \Big\{ \mathcal{B}_3[\mathcal{N}(x_4), M - m_4 x_4] \\ &\qquad - s_{14} x_1 x_4 - s_{34} x_3 x_4 \Big\} \end{aligned} \tag{20}$$



(a) (b) (c)

Fig. 1. The graphs extracted from objective functions (7), (9) and (19).

In many applications, the set $\mathcal{N}(x_i)$ contains a large number of elements, which will make the computational cost of (17) increasingly expensive. We address this problem by decreasing the number of variables in each recurrence and building the approximated iterative equations as follows:

$$\begin{aligned} B_i[x_i, M_i] &= \begin{cases} \psi_i^{\text{new}}(x_i, M_i), & \text{if } m_i x_i \le M_i \\ -\infty, & \text{otherwise} \end{cases} \\ \psi_i^{\text{new}}(x_i, M_i) &\ge v_i x_i + \max_{x_{i-1}} \Big\{ B_{i-1}[x_{i-1}, M_{i-1}] \\ &\quad - s_{(i-1,i)} x_i x_{i-1} \\ &\quad - \sum_{x_j \in \mathcal{N}(x_i) \setminus \{x_{i-1}\}} s_{ij} x_i X_j(x_{i-1}, M_{i-1}) \Big\} \\ X_j(x_i, M) &= \arg\max_{x_j \in \{x_1, \ldots, x_{i-1}\}} B_i[x_i, M] \end{aligned} \tag{21}$$

where $X_j(x_i, M) \in \{0, 1\}$, and it stands for the value of x_j when the maximum value $B_i[x_i, M]$ is obtained from the first i items with the label of x_i fixed.

Similar to (13), the value of $X_j(x_i, M)$ can also be calculated via dynamic programming. In every iteration, we need to calculate $X_j(x_i, M)$ for solving more general submodular energy function (14) by our new approximate algorithm.

Algorithm 1: Knapsack constrained submodular maximization

1. Input: $f(\cdot)$, $g(\cdot)$ and M
2. for $m = 1 : M$
3. Calculate $B_1[x_1, m]$ by (12)
4. end for
5. for $i = 2 : n$
6. for $m = 1 : M$
7. Find all the elements j, where $j < i$ and $s_{ji} > 0$
8. Calculate $X_j(x_{i-1}, m - v_i x_i)$ and $B_i[x_i, M]$ by (21)
9. end for
10. end for
11. Calculate $\mathbf{x}$ by (13)
12. Output: $\mathbf{x}$ and $\max_{x_n} B_n[x_n, M]$

According to the example of (19) and the related graph in Fig. 1 (c), we obtain its new iterative equation as follows:

$$\begin{aligned} \psi_4^{\text{new}}(x_4, M) &\ge v_4 x_4 + \max_{x_3} \Big\{ B_3[x_3, M_3] \\ &\quad - s_{43} x_4 x_3 - s_{14} x_4 X_1(x_3, M_3) \Big\} \\ M_3 &= M - m_4 x_4 \\ X_1(x_3, M_3) &= \arg\max_{x_1} B_3[x_3, M_3] \end{aligned} \tag{22}$$

Algorithm 1 gives the pseudo-code of our submodular maximization method. Since we obtain all the energies when M_i gets its value from 0 to M, it considers all the situations when the variables get different values. In the example of (19), we first adopt DP in Algorithm 1 to obtain the maximum of $F(x_1, x_2, x_3, x_4)$ without knowing the specific value of x_4. Then, the value of x_4 is obtained by (13). Thus, the term $M - m_4 x_4$ in (22) can be calculated after that.

Our function is solved by dynamic programming, which is not optimized by iterations. Thus, our method can be judged as convergent or not. And our time complexity is $O(nM)$ according to the property of dynamic programming, where n is the number of variables.

C. Solution range reduction

In our experiments, we found that dynamic programming is based on heavy heuristics, where the computational cost of one recursion is high with lots of related variables, especially when matrix P of the pairwise term in (15) is not sparse. Therefore, we adopt an efficient method [5] to further reduce the range of solution prior to performing the dynamic programming. This is originally designed for obtaining minimum energy of a supermodular function without a constraint. The solution range reduction is employed on the original submodular energy, and it will confirm some values of a subset of the variables. We get a new function with fewer variables and a more sparse pairwise parameter P. The new function is then optimized by our submodular maximization method. And this procedure is far more efficient than performing the maximization on the original submodular function.

For a submodular function $f(T), T \subseteq V$, V is the ground set. Considering two subsets of V: S and L, if they satisfy

$$\emptyset \subseteq S \subseteq L \subseteq V, \tag{23}$$

We define a set of subsets called subinterval as follows:

$$[S, L] = \{T | S \subseteq T \subseteq L\} \tag{24}$$

The maximum value of function $f(T)$ on interval $[S, L]$ is denoted by $f^*[S, L]$:

$$f^*[S, L] = \max_{T \in [S,L]} f(T) \tag{25}$$

Function f is submodular on $[S, L]$, if $\forall \beta, \gamma \in [S, L]$, and it has the property that $f(\beta) + f(\gamma) \geq f(\beta \cup \gamma) + f(\beta \cap \gamma)$. Expressions of forms $S \setminus \{k\}$ and $S \cup \{k\}$ can be written as $S - k$ and $S + k$.

THEOREM 1. Let f be a submodular function on interval $[S, L] \subseteq [\emptyset, V]$, and let $k \in L \backslash S$. Then the following assertions hold:

$$\begin{aligned} &\text{(a).} \quad f^*[S, L-k] - f^*[S+k, L] \geq f(S) - f(S+k) \\ &\text{(b).} \quad f^*[S, L-k] - f^*[S+k, L] \leq f(L-k) - f(L) \end{aligned} \tag{26}$$

Proof. (a) Let $\gamma \in [S, L-k]$ with $f(\gamma+k) = f^*[S+k, L]$. It then follows from the definition of submodularity that:

$$\begin{aligned} f(S+k) + f(\gamma) &\geq f((S+k) \cup \gamma) + f((S+k) \cap \gamma) \\ &= f(\gamma + k) + f(S) \end{aligned} \tag{27}$$

Hence, using the above derivation, we have:

$$f^*[S, L-k] \geq f(\gamma) \geq f(\gamma+k) + f(S) - f(S+k) \tag{28}$$

Thus, putting $-f^*[S+k, L]$ at both sides of (28), we get:

$$\begin{aligned} f^*[S, L-k] - f^*[S+k, L] &= f^*[S, L-k] - f(\gamma+k) \\ &\geq f(S) - f(S+k) \end{aligned} \tag{29}$$

The proof of (b) is similar.

COROLLARY 1. Let f be a submodular function on interval $[S, L] \subseteq [\emptyset, V]$, and let $k \in L \setminus S$. Then the following assertions hold:

(a). First Preservation (FP) Rule: If $f(S+k) \leq f(S)$, then $f^*[S, L] = f^*[S, L-k] \geq f^*[S+k, L]$.

(b). Second Preservation (SP) Rule: If $f(L-k) \leq f(L)$, then $f^*[S, L] = f^*[S+k, L] \geq f^*[S, L-k]$.

Proof. From Theorem 1.(a) we have that: $f^*[S, L-k] - f^*[S+k, L] \geq f(S) - f(S+k)$. By assumption $f(S+k) \leq f(S)$, $f^*[S, L-k] \geq f^*[S+k, L]$, and thus $f^*[S, L] = f^*[S, L-k]$. The proof of (b) is similar.

Corollaries (a) and (b) can only reduce the solution range of maximizing a submodular energy function without any constraints. However, our framework in (15) contains a constraint. Because a part of solution may not satisfy the constraint, these two determination methods cannot be used directly. As a result, when one variable can be ensured by corollary (a) or (b), we determine whether this variable can satisfy the constraint. If not, we terminate the range reduction procedure. Our new maximization algorithm for (15) is given in Algorithm 2.

The steps before line 17 in Algorithm 2 belong to the range reduction, which can get a partial solution of the original problem. Line 17 in Algorithm 2 is to solve the new problem with a smaller scale. For example, we consider the following maximization problem with n variables:

$$\begin{aligned} &\max x_1 + 2x_2 - x_1x_2 - 3x_2x_3 + f(x_3, \ldots, x_n), \\ &\text{s.t.} \quad 2x_1 + 3x_2 + g(x_3, \ldots, x_n) \leq 20 \end{aligned} \tag{30}$$

TABLE II
THE AVERAGE MAXIMUM ENERGIES OF 1000 FUNCTIONS BY THE GREEDY ALGORITHM, OUR ALGORITHM 1 AND ALGORITHM 2, RESPECTIVELY.

	Greedy	Algorithm 1	Algorithm 2
Average energy	72.920	77.439	77.408
Average time	0.0537	0.0569	0.0557

Algorithm 2: The final algorithm with range reduction

```
1.  Input: f(.), g(.) and M
2.  S = ∅, L = V
3.  while
4.     P_0 = P_1 = ∅
5.     for x_i ∈ L \ S
6.           If f(S + x_i) ≤ f(S)
7.                P_0 = P_0 + x_i
8.           If f(L − x_i) ≤ f(L)
9.                P_1 = P_1 + x_i
10.    end for
11.    If g(S ∪ P_0) ≤ M
12.          Update f(.), g(.) and M
13.          S = S ∪ P_0, L = L ∪ P_1
14.    else
15.          break
16. end while
17. Get the updated f(.), g(.) and M
18. Use Algorithm 1 to get the final solution
```

We suppose two variables are used to obtain their solutions ($x_1 = 0, x_2 = 1$) by the range reduction step, then the updated problem is defined as follows:

$$\begin{aligned} &\max -3x_3 + f(x_3, \ldots, x_n), \\ &\text{s.t.} \quad g(x_3, \ldots, x_n) \leq 17 \end{aligned} \tag{31}$$

If the objective submodular energy is monotonically decreasing, we can get its solution directly by the range reduction step. If the energy is not monotone, the range reduction step in Algorithm 2 can get the partial solution, and thus decrease the scale of the original problem. Our Algorithm 1 by dynamic programming can maximize a new function with less variables and obtain the solution of original energy efficiently. The experimental results in Fig. 2 (a) and Table *II* will demonstrate the time acceleration of this range reduction step.

IV. EXPERIMENTAL RESULTS

In this section, we first build synthetic submodular functions with the MNIST dataset as their parameters. By counting the maximum energies of these functions, we compare our optimization methods (with/without the range reduction step) with the classic greedy algorithm. Then, we apply our submodular maximization model to two vision problems: image segmentation and motion trajectory clustering.

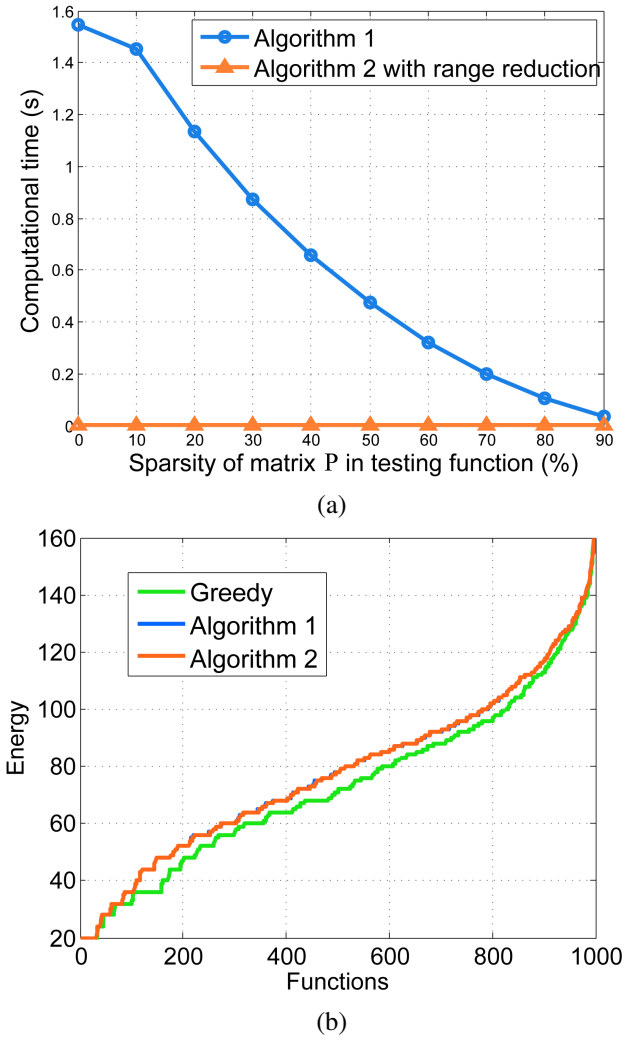


Fig. 2. Comparison of time cost and accuracy for our Algorithms 1, 2 and the greedy algorithm on synthetic data.(a) The computational time of Algorithms 1 and 2. When the sparsity of matrix P in (15) decreases, these two methods cost more time. However, the range reduction step in Algorithm 2 will decrease the number of variables before DP. Therefore, it takes less time than Algorithm 1. (b) Illustration of getting maximum energies of 1000 synthetic functions by the greedy algorithm, Algorithms 1 and 2. In most cases, both our Algorithm 1 and Algorithm 2 outperform the greedy algorithm.

A. Synthetic data experiment

In the first experiment, we build 1000 submodular energy functions to evaluate the proposed method with random initialization, whose random sequence is taken from a public MNIST dataset *(http://yann.lecun.com/exdb/mnist)*. This dataset contains 70000 handwriting digits in random order and their features, and it is originally used for character recognition. Since the parameters for matrices P, U, C and M in (15) of our model need to be non-negative numbers, the number labels of digits (from 0 to 9) in the MNIST dataset are suitable to build submodular functions. The numbers in this dataset are filled in parameter matrices P, U, C and M of our testing energy functions. We divide all 70K numerical digits into 1000 data blocks, where 70 numbers in each block are used to build one submodular function and its constraint. The procedure of building a testing submodular energy with one data block is provided in **Appendix III**. By maximizing these synthetic functions, we compare our methods without/with the range reduction step (Algorithm 1/Algorithm 2) and the classic greedy algorithm.

The reason of choosing a public dataset instead of random numbers to build synthetic functions is that it is easy to re-implement. We use the MNIST dataset because it contains digits from 0 to 9 with almost equal numbers. We compare the average maximum energies and time of 1000 synthetic functions of the greedy algorithm with our methods (without/with the range reduction step) in Table *II*. Our methods achieve higher average energy than the greedy algorithm, which means the proposed DP process obtains more accurate solutions. In these results, our Algorithm 2 obtains larger energy values than the greedy algorithm in 331 functions. In each iteration of the greedy algorithm, only one variable is set to 1, and it will never be changed in the later iterations. Therefore, in many situations, the greedy algorithm will be stuck in local maxima. While in our dynamic programming method, we keep track of maximum values in each iteration, as parameter M increases from 0 to M. As a result, dynamic programming has a larger chance to jump out of the local maximum to get a larger energy. The range reduction step can reduce the computational cost of our DP algorithm. Since the average energy of Algorithm 2 is just slightly smaller than that of our original Algorithm 1, we then adopt the more efficient Algorithm 2 as our optimization method to solve vision problems in the later experiments. As shown in Fig. 2(a), the range reduction step in Algorithm 2 reduces the computational time enormously. The time difference between Algorithm 1 and Algorithm 2 is not large in Table *II*, since the matrix P in this synthetic data experiment is not sparse. Fig. 2(b) provides the maximum energies of these three methods from all 1000 functions, and both Algorithms 1 and 2 perform better than the greedy algorithm.

Now we give another example to demonstrate the comparison of our method with greedy algorithm. The function (43) in **Appendix III** is the submodular function, and its knapsack constraint is built by the first data block. Its maximum energies are 14 and 32, which are optimized by greedy algorithm and our DP method with range reduction, respectively. The time costs of greedy algorithm and our DP method are 0.000131 and 0.001057, respectively. Therefore, our method has the comparable computational time to achieve a higher energy, and obtains a more accurate solution than greedy algorithm.

B. Interactive image segmentation

Now we apply our submodular maximization model to image segmentation with user strokes. A lot of segmentation methods have been proposed [10], [14], [6], [49], [13], [29], [32]. All these segmentation methods adopted additional prior information or complicated assumptions to get better results. The proposed segmentation method with the submodular maximization does not rely on sophisticated features. We only add a knapsack constraint explicitly to our new submodular energy to increase the segmentation accuracy. Thus, we compare our method with classic graphcut and random walk for implementation simplicity.

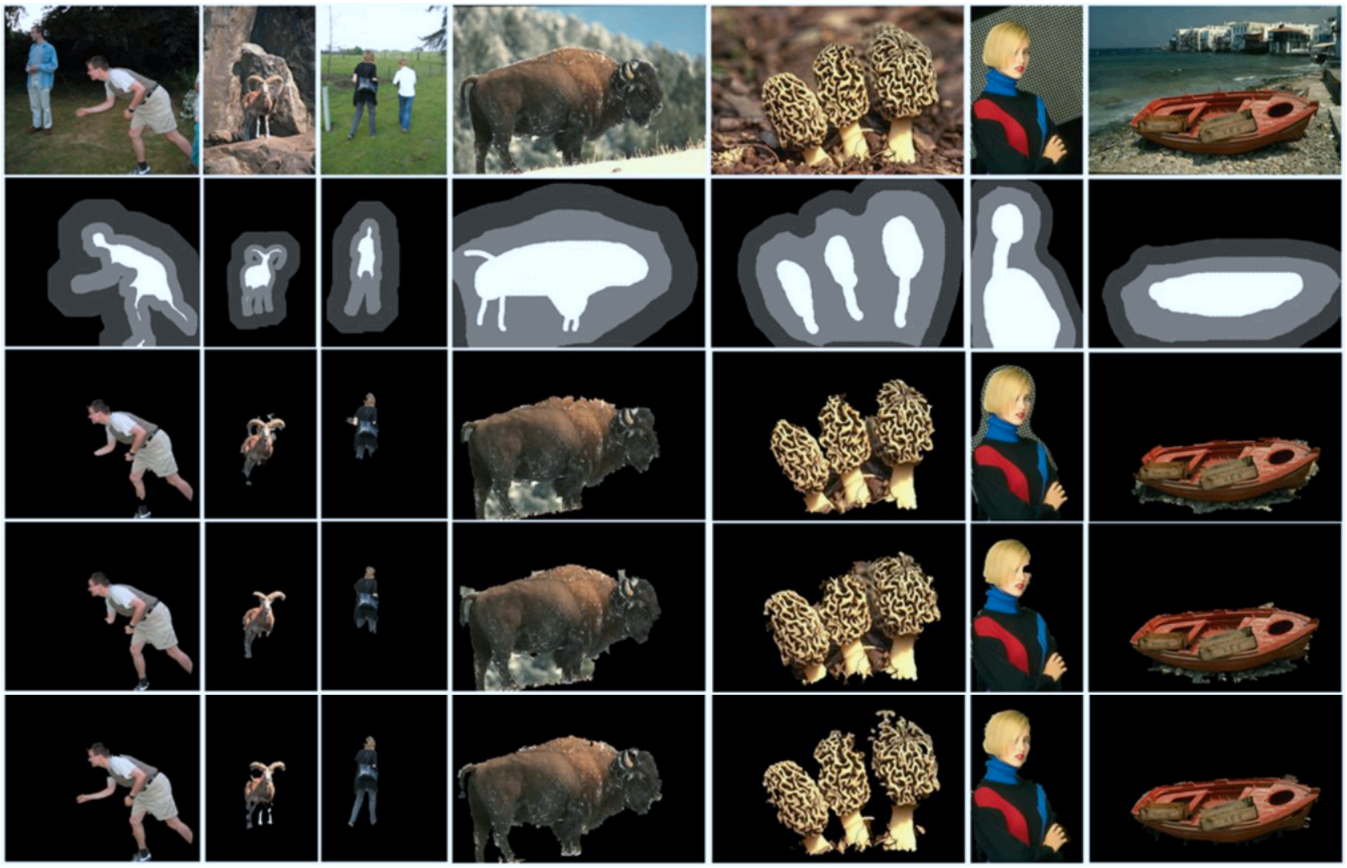


Fig. 3. Segmentation results by the Graphcut (3rd row), random walk (4th row) and the proposed submodular model (5th row). The first and second rows are the input images and user strokes from the GrabCut dataset (column 1∼3) and the selected 'BSD-70' dataset (column 4∼7).

We first obtain the set of superpixels by SLIC [24]. One superpixel i with the label 1 or 0 means that superpixel belongs to foreground or background. According to the segmentation model of graphcut, if two adjacent superpixels present different color features, they should be assigned to different labels. Therefore, our quadratic submodular maximization is adopted to build the following energy with variables $\mathbf{x}$:

$$\begin{aligned} F(\mathbf{x}) = &\sum_i \{(\mathrm{Fg}(i) - \mathrm{Bg}(i))x_i + \mathrm{Bg}(i)\} \\ &+ \sum_{(i,j)\in\mathcal{N}} \mathrm{diff}(i,j)\{\lambda_d x_i(1-x_j) + \lambda_d(1-x_i)x_j \\ &- \lambda_1 x_i x_j - \lambda_0(1-x_i)(1-x_j)\} \\ = &\sum_i (\mathrm{Fg}(i) - \mathrm{Bg}(i))x_i \\ &+ \sum_{(i,j)\in\mathcal{N}} \mathrm{diff}(i,j)\{(\lambda_d + \lambda_0)(x_i + x_j) \\ &- (2\lambda_d + \lambda_1 + \lambda_0)x_i x_j\} + o \end{aligned} \tag{32}$$

where $\mathrm{Fg}(i)$ and $\mathrm{Bg}(i)$ are probabilities of superpixel i belonging to foreground or background, which are calculated by the Gaussian Mixture Model (GMM) obtained from user scribbles. o is a constant term, which can be omitted when maximizing the above energy. $\mathcal{N}$ is the set of all adjacent superpixel pairs. $\mathrm{diff}(i,j)$ is the color difference of superpixels i and j. λ_1, λ_0 and λ_d are three parameters for balancing unary and pairwise terms. In our experiments, we set $(\lambda_1, \lambda_0, \lambda_d) = (1, 0, 0)$. Different from the classic Graphcut [10] or random walk [14] methods, our submodular model adds extra constraints explicitly in order to include more prior information.

i) We can restrict the maximum number of foreground superpixels. Therefore, a cardinality constraint can be added to the original problem (32):

$$g(\mathbf{x}) = \sum_i x_i \leq K \tag{33}$$

ii) We can assume superpixels at the center of an image have a larger probability to be foreground. As a result, the following constraint can obtain a better segmentation result:

$$g(\mathbf{x}) = \sum_i \mathrm{dist}(i) x_i \leq K \tag{34}$$

where $\mathrm{dist}(i)$ describes the Euler distance of superpixel i to the center of an image. And constant K is also used to constrain the total number of foreground superpixels.

We use the GrabCut dataset and the randomly selected 70 images from the BSD dataset to show the effectiveness of our new model in (32) with a knapsack constraint in (34). The GrabCut dataset only includes 20 images, and the user strokes are next to the foreground objects. In order to demonstrate the effectiveness of segmentation, we randomly select 70 images including more complex scenes from the BSD

TABLE III
THE AVERAGE ACCURACY OF THE GRAPHCUT, RANDOM WALK, SUBRW AND THE PROPOSED METHOD ON THE GRABCUT AND 'BSD-70' DATASETS.

	Graphcut	Random walk	SubRW	The proposed
Grabcut	98.15 %	98.43 %	98.43 %	98.91 %
BSD-70	80.78 %	80.52 %	80.94 %	81.36 %

Fig. 4. An illustration of failure case with our method. The left image is an input image from 'BSD-70' dataset and the middle one is its corresponding user stroke. Our segmentation result is shown in the right image.

dataset. As shown in Fig. 3, we compare our model with two optimization methods: Graphcut [10] and random walk [14]. We also measure accuracies of these methods on the whole GrabCut and our 'BSD-70' dataset, which are presented in Table *III*. All three methods can obtain high accuracies on the GrabCut dataset, which mostly includes the simple scenes of foreground objects. In contrast, our submodular segmentation method obtains better accuracies on the 'BSD-70' dataset with more complex scenes. The extra constraint in our model makes the center superpixels to obtain a larger probability to be selected as foreground, and thus the proposed method achieves better segmentation results.

We also illustrate a failure case in Fig. 4. In this case, there exist some fragments of background, such as the fragments of stone. The color of these fragments is very similar with the foreground object so that our method cannot distinguish them from foreground. The reason may be that the prior model GMM fails to discriminate the similar background and foreground. Actually, our energy function (32) contains data term (storing global information of foreground and background, and smooth term (including $diff(i,j)$) used to keep consistent among neighbors. In this failure case, these fragments of the stones are too similar with the foreground in the view of GMM model, and so that the data term plays the main role in energy function. In the future, better appearance model (such as deep model [20], [34]) will reduce the data term in these situations.

C. Motion trajectory clustering

Most of the popular methods for moving object segmentation [28], [36], [39], [45], [44], [48], [37] are based on the motion trajectory clustering, and many researchers focused on improving the accuracy of the trajectory clusters. Instead of applying clustering directly on trajectories, we aim to select better clustering centers from all the trajectories. Our submodular energy model first finds the centers of small trajectory clusters, which are called trajectory fragments. Then, a aggregation step is used to merge those fragments into the final trajectory clusters. $T = \{t_i, t_2, \cdots, t_n\}$ denotes the set of all the initial trajectories, where n is the number of trajectories. We assume the similarity of t_i and t_j as $w(i,j)$, and all those similarity measures construct an affinity matrix W. A submodular function is built to select a set of representative trajectories S with good quality, and the trajectories in S will be used as the clustering centers. We define a variable $x_i \in \{0,1\}$, and $x_i = 1$ denotes that t_i should be included in S. Therefore, the set S is obtained by maximizing the following $F(\mathbf{x})$ subject to $g(\mathbf{x}) \leq K$:

$$\begin{aligned} F(\mathbf{x}) &= \sum_i q(i)x_i - \lambda \sum_{i,j} w(i,j)x_i x_j \\ g(\mathbf{x}) &= \sum_i x_i \leq K \end{aligned} \tag{35}$$

where $q(i)$ measures the quality of t_i. A smaller $q(i)$ indicates that t_i may be connected by incorrect optical flows.

When generating trajectories, we judge whether the forward and backward optical flows are matched. If two optical flows are mismatched, the trajectory is connected by incorrect optical flows. λ is a parameter for balancing these two terms, which is set to 2 in our experiments. By maximizing this energy, we get the selected set of centers. Then, we distribute every trajectory to one center with the largest similarity, using the affinity matrix W. And all trajectories assigned to one center will build a trajectory fragment. Later, we employ the fast bottom-up aggregation clustering method to get the final clustering result by merging all fragments. In our experiments, we only use N_s trajectories to select the clustering centers to reduce computing load. These trajectories are sampled every N/N_s trajectories from all N trajectories.

We compare our submodular maximization model with two state-of-the-art trajectory clustering algorithms: the Spectral Clustering based method (SC) [36] and Minimum Cost Multicuts (MC) [45] on the FBMS-59 dataset. The authors in [45] proposed several variants for experiments. Here we select the version 'Multicut on enriched graph (**MCe**) sparse (4), prior 0.5' for comparison. The FBMS-59 dataset contains 59 videos with ground truths. Fig. 5 gives two representative clustering results by these methods. In video 'cars2' (the first row, left), both SC and MC regard the two foreground moving cars as one motion entity. Because the motion difference of these two cars is too small to be detected by their spectral clustering or cut algorithms. However, our method divides them into two moving objects, since these two cars have different clustering

TABLE IV
RESULTS OF SC, MC AND THE PROPOSED METHOD ON THE FBMS-59 DATASET.

Training set	D	P	R	F	O
SC [36]	3.71 %	82.33 %	64.26 %	72.27 %	17/65
MC [45]	3.47 %	86.79 %	73.36 %	79.51 %	28/65
Ours	4.64 %	86.74 %	74.15 %	79.95 %	31/65
Test set					
SC [36]	3.95 %	76.15 %	61.11 %	67.81 %	22/69
MC [45]	3.72 %	86.81 %	67.96 %	76.24 %	25/69
Ours	4.53 %	87.09 %	68.30 %	76.57 %	26/69

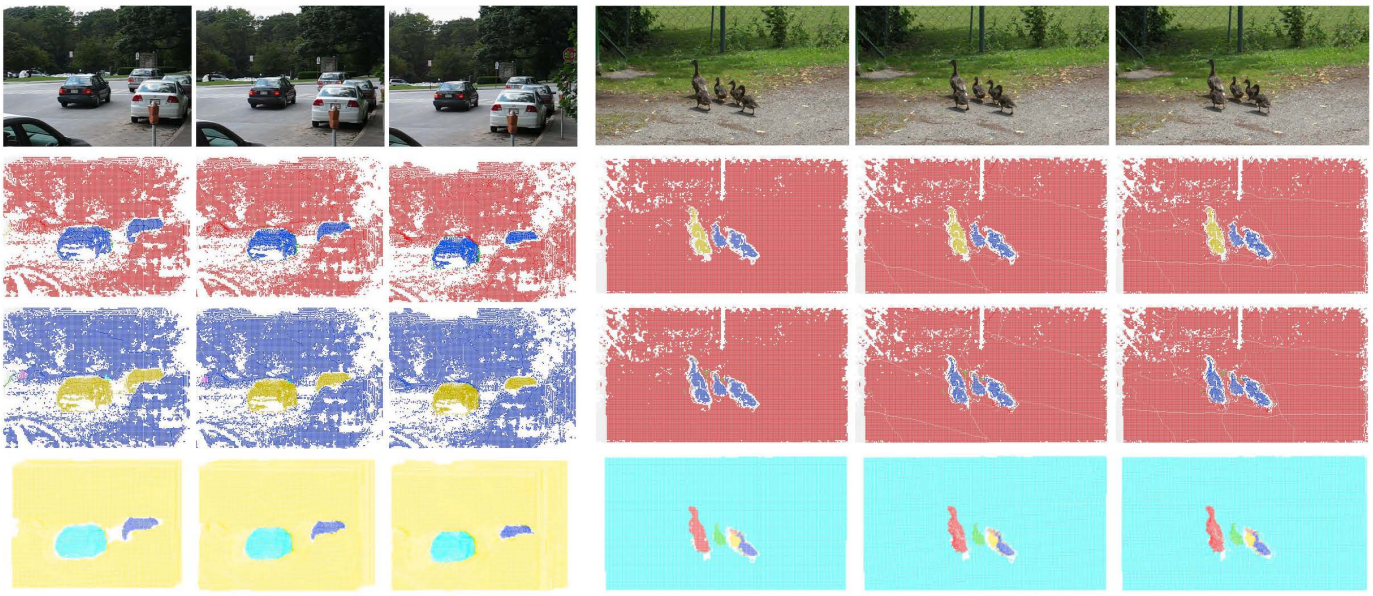


Fig. 5. Trajectory clustering results by the SC method [36], the MC method [45] and our submodular maximization approach. The video frames in 1st row are taken from 'car2' and 'ducks01'. The 2nd row is results from SC; the 3rd row is results from MC. Our results are shown in the 4th row. Our method can detect different clusters of moving objects, which is better than both SC and MC.

centers, which are obtained by maximizing (35). As for video 'ducks01' (the first row, right), SC only divides all ducks into two clusters, and MC cannot separate them correctly. Our method can select the proper trajectory clustering centers with good quality. All ducks have been distributed to more than one centers, and then four ducks can be correctly segmented out in our results. The leftmost two ducks have similar motions and textures, thus they are treated as one class.

Following the measurements in [45], we also use five criteria to illustrate the performance of those results quantitatively, which include the average region density (D), average precision (P), average recall (R), F-measure (F) and extracted objects with $F \geq 75\%$ (O). The numeric comparison is shown in Table *IV*. Our trajectory clustering method selects the most representative trajectories with good quality as the centers of clusters, which means the 'bad' trajectories (connected by incorrect optical flows) will not be selected. Every trajectory is assigned to its nearest center, as a result, the 'bad' trajectories will not affect the clustering results. Therefore, our method achieves better clusters with accurate classification and less noise around the moving objects.

It is worth to mention that there are some clustering algorithms based on submodular optimization, such as [12], [38]. Zhao *et al.* [12] formulate clustering as a multiway partition problem to solve it by minimizing a submodular energy. These two methods are based on submodular minimization while our method is based on submodular maximization framework. Liu *et al.* [38] build a submodular function based on entropy-rate and maximize it to get the clusters, which is further applied to superpixels. However, this method requires the number of clusters and features as inputs. In motion trajectory clustering, the number of clusters usually can not be determined, and we only obtain the adjacent matrix built by similarities between motion trajectories. Therefore, these methods are not suitable for solving the motion trajectory clustering, and they are more appropriate for general data clustering.

V. Conclusion

In this paper, we proposed a new class of submodular maximization model: a quadratic submodular energy function subject to a knapsack constraint. Different from the facility location and information theoretic models, our model contains a more general knapsack constraint, which is able to represent more vision prior information. Our maximization algorithm was developed using the principle of dynamic programming where we approximated the recurrence formula to make the problem tractable (Algorithm 1). We then introduced the solution range reduction step (Algorithm 2) to obtain more accurate results than the standard greedy algorithm. The utility of our approach was demonstrated by designing novel energies for the tasks of interactive image segmentation and selecting cluster centers for motion trajectories. Experimental results of image segmentation demonstrate that our method out-performs the classic algorithms of graphcut and random walk. Moreover, our framework achieves better performance than state-of-the-art methods on the motion trajectory clustering task. In the future, we plan to investigate more computer vision applications based on our submodular optimization framework. In addition, we will explore the possibility of adopting dynamic programming to solve more non-submodular energies and multi-objective optimization problems [47].

Appendix I: The expressions of $H(\cdot)$ and $M(\cdot)$ in (5) of Section I

Entropy rate function $H(\cdot)$ is defined on a random walk of a graph $G = (C, V)$. V is the ground set of the set function $H(\cdot)$, and it represents the set of all edges in graph G. C is

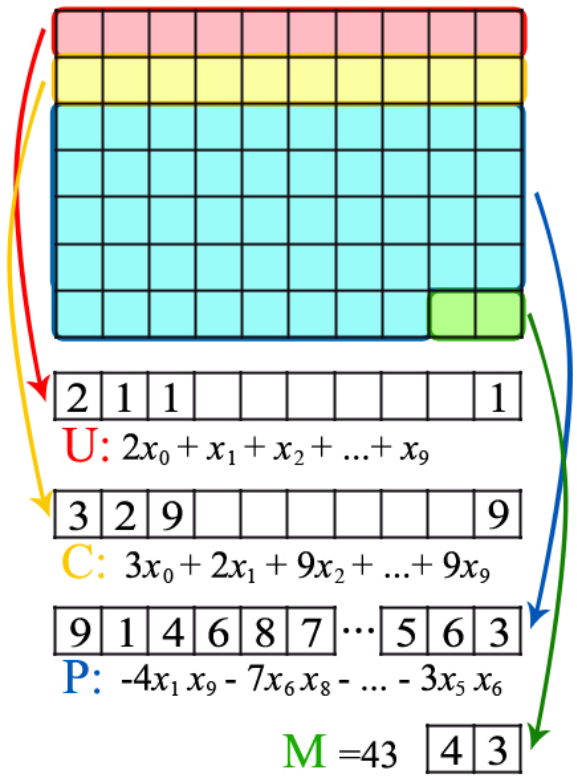


Fig. 6. An illustration of building a quadratic submodular function subject to a knapsack constraint by a data block from the MNIST dataset.

the set of all vertices of the graph, and $\mu_i(i = 1, 2, \ldots, |C|)$ is the stationary distribution for this random walk. S is a subset of V. $p_{i,j}(S)$ is the transition probability from vertex i to j. As a result, the definition of entropy rate $H(\cdot)$ is:

$$H(S) = -\sum_i \mu_i \sum_j p_{i,j}(S) \log(p_{i,j}(S)) \tag{36}$$

The function of mutual information $M(\cdot)$ between the sets S and $V \backslash S$ is defined as follows:

$$M(S, V\backslash S) = \mathcal{H}(S) + \mathcal{H}(V\backslash S) - \mathcal{H}(V) \tag{37}$$

where $\mathcal{H}(.)$ is the entropy function. Assume n is the number of elements in the ground set V, and L is the covariance matrix $\mathbb{R}^{n\times n}$ of V. For $S \subseteq V$, let L_S be the principal submatrix of L indexed by S. Then, the entropy $\mathcal{H}(S)$ is defined as:

$$\mathcal{H}(S) = \frac{1 + \log(2\pi)}{2}|S| + \frac{1}{2}\log(\det(L_S)) \tag{38}$$

where $\det$ is the determinant of the matrix.

Appendix II: The Submodularity of Functions (8), (9) and (14) in Section *III*

Definition: The energy functions (8), (9) and (14) in Section *III* are submodular.

Proof: We assume an energy function with n variables takes the form of (8):

$$\begin{aligned} &F(x_1, x_2, \ldots, x_n) = v_1x_1 + v_2x_2 + \cdots + v_nx_n \\ &v_i > 0, i = 1, 2, \ldots, n \end{aligned} \tag{39}$$

For any two variables x_s and x_t, $1 \le s < t \le n$, we have:

$$\begin{aligned} &F(x_1, \ldots, x_s = 1, \ldots, x_t = 0, \ldots, x_n) + \\ &F(x_1, \ldots, x_s = 0, \ldots, x_t = 1, \ldots, x_n) - \\ &F(x_1, \ldots, x_s = 0, \ldots, x_t = 0, \ldots, x_n) - \\ &F(x_1, \ldots, x_s = 1, \ldots, x_t = 1, \ldots, x_n) = \\ &v_s + v_t - v_s - v_t = 0 \end{aligned} \tag{40}$$

Therefore, the function (8) is submodular, according to the definition of submodular energy. We define a binary quadratic energy function $f(x_i, x_j) = -sx_ix_j, s \ge 0$. It is submodular since $f(1,0) + f(0,1) - f(0,0) - f(1,1) = s \ge 0$. If two functions f_1 and f_2 are both submodular, then $\alpha_1 f_1 + \alpha_2 f_2(\alpha_1, \alpha_2 \ge 0)$ is also submodular. For any two variables x_i and x_j, we design a new energy:

$$F(x_1, \ldots, x_n) = \sum_{i=1}^{n} v_ix_i - s_{ij}x_ix_j, s_{ij} \ge 0 \tag{41}$$

The above function (41) is submodular. Functions (9) and (14) can be viewed as the sum of linear terms and submodular quadratic terms, and thus they are both submodular.

Appendix III: Building Submodular Energies Using Digits from MNIST Dataset in Synthetic Experiment of Section *IV*

The MNIST dataset contains 70000 numerical digits from '0' to '9'. We divide all the numerical digits into 1000 data blocks, where 70 digits in each block are used to build one submodular energy function with 10 variables and its constraint. The prototype of a quadratic submodular maximization problem is defined in (15), and 70 digits are used to build the matrices U, P, C and M as follows. We use the first data block in MNIST dataset as an example in Fig. 6. This block is divided into four parts. The first row of 10 digits (red part) builds the matrix U for 10 variables. The second row of 10 numbers (yellow part) constructs the matrix C in the knapsack constraint. M is formed by the last two digits of the block (green part). And a vector P_b from the 21-st to the 68-th digits of the block (blue part) builds the parameters of the quadratic term P. Every 3 digits in the vector P_b determine one pairwise term. In one data triplet, the first 2 numbers are the subscripts of the quadratic term, and the last number is its coefficient. For example, digits '9''1''4' build a term $-4x_1x_9$. Finally, the maximization problem by the first block is obtained as:

$$\begin{aligned} &\max F(x_0, x_1, \ldots, x_9) \\ &\text{s.t.} \quad g(x_0, x_1, \ldots, x_9) \le M \end{aligned} \tag{42}$$

where

$$\begin{aligned} F(x_0, x_1, \ldots, x_9) &= 2x_0 + x_1 + \cdots + x_9 \\ &\quad - 4x_1x_9 - 7x_6x_8 - \cdots - 3x_5x_6 \\ g(x_0, x_1, \ldots, x_9) &= 3x_0 + 2x_1 + \cdots + 9x_9, \quad M = 43 \end{aligned} \tag{43}$$

References

[1] G. L. Nemhauser, L. A. Wolsey and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions-I," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.

[2] L. Liu, L. Shao, Sequential compact code learning for unsupervised image hashing, *IEEE Trans. on Neural Networks and Learning Systems*, vol. 27, no. 12, pp. 2526-2536, 2016.

[3] M. Conforti and G. Cornuéjols, "Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the Rado-Edmonds theorem," *Discrete applied mathematics*, vol. 7, no. 3, pp. 251-274, 1984.

[4] A. Caprara, D. Pisinger, P. Toth, "Exact solution of the quadratic knapsack problem," *Informs Journal on Computing*, vol. 11, no. 2, pp. 125-137, 1998

[5] B. Goldengorin, G. Sierksma, G. A. Tijssen, and M. Tso, "The data-correcting algorithm for the minimization of supermodular functions," *Management Science*, vol. 45, no. 11, pp. 1539-1551, 1999.

[6] J. Shen, Y. Du, W. Wang, and X. Li, "Lazy random walks for superpixel segmentation," *IEEE Trans. on Image Processing*, vol. 23, no. 4, pp. 1451-1462, 2014.

[7] S. Khuller, A. Moss and J. S. Naor, "The budgeted maximum coverage problem," *Information Processing Letters*, vol. 70, no. 1, pp. 39-45, 1999.

[8] J. Hooker, "Nonserial dynamic programming," *Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction*, pp. 423-441, 2000.

[9] M. Sviridenko, "A note on maximizing a submodular set function subject to a knapsack constraint," *Operations Research Letters*, vol. 32, no. 1, pp. 41-43, 2004.

[10] V. Kolmogorov and R. Zabin, "What energy functions can be minimized via graph cuts?" *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 26, no. 2, pp. 147-159, 2004.

[11] S. Fujishige, "Submodular functions and optimization," *Elsevier Science*, 2nd Edition, 2005.

[12] L. Zhao, H. Nagamochi, T. Ibaraki, "Greedy splitting algorithms for approximating multiway partition problems," *Mathematical Programming*, vol. 102, no. 1, pp. 167-183, 2005.

[13] X. Dong, J. Shen, L. Shao, and L. Van Gool. Sub-markov random walk for image segmentation. *IEEE Trans. on Image Processing*, 25(2):516–527, 2016.

[14] G. Leo, "Random walks for image segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 28, no. 11, pp. 1768-1783, 2006.

[15] G. Calinescu, C. Chekuri, M. Pál and J. Vondrak, "Maximizing a submodular set function subject to a matroid constraint," *International Conference on Integer Programming and Combinatorial Optimization*, pp. 182-196, 2007.

[16] U. Feige, V. Mirrokni, J. Vondrák, "Maximizing non-monotone submodular functions," *IEEE Symposium on Foundations of Computer Science (FOCS)*, IEEE Computer Society, pp. 461-471, 2007.

[17] D. Liu, K. Zhou, D. Bernhardt, and D. Comaniciu, "Search strategies for multiple landmark detection by submodular maximization," *In Proceedings of IEEE CVPR*, pp. 2831-2838, 2010.

[18] T. Brox and J. Malik, "Object segmentation by long term analysis of point trajectories," *In: Proceedings of ECCV*, pp. 282-295, 2010.

[19] J. Lee, V. Mirrokni, V. Nagarajan, M. Sviridenko, "Maximizing non-monotone submodular functions under matroid or knapsack constraints," *Siam Journal on Discrete Mathematics*, vol. 23, no. 4, pp. 2053-2078, 2010.

[20] W. Wang, and J. Shen, "Deep visual attention prediction," *IEEE Trans. on Image Processing*, vol. 27, no. 5, pp. 2368-2378, 2018.

[21] Y. Yue and C. Guestrin, "Linear submodular bandits and their application to diversified retrieval," *In Proceedings of NIPS*, pp. 2483-2491, 2011.

[22] S. Jegelka, H. Lin and J. A. Bilmes, "On fast approximate submodular minimization," *In Proceedings of NIPS*, pp. 460-468, 2011.

[23] A. Kulik, H. Shachnai, T. Tamir, "Approximations for monotone and non-monotone submodular maximization with knapsack constraints," *Mathematics of Operations Research*, vol. 38, no. 4, pp. 729-739, 2011.

[24] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274-2282, 2012

[25] A. Delong, O. Veksler, A. Osokin and Y. Boykov, "Minimizing sparse high-order energies by submodular vertex-cover," *In Proceedings of NIPS*, pp. 962-970, 2012.

[26] R. Iyer and J. A. Bilmes, "Submodular optimization with submodular cover and submodular knapsack constraints," *In Proceedings of NIPS*, pp. 2436-2444, 2013.

[27] Z. Jiang and L. S. Davis, "Submodular salient region detection," *In Proceedings of IEEE CVPR*, pp. 2043-2050, 2013.

[28] F. Shi, Z. Zhou, and J. Xiao and W. Wu, "Robust trajectory clustering for motion segmentation," *In Proceedings of IEEE ICCV*, pp. 3088-3095, 2013.

[29] J. Shen, X. Hao, Z. Liang, Y. Liu, W. Wang, and L. Shao, "Real-time Superpixel Segmentation by DBSCAN Clustering Algorithm," *IEEE Trans. on Image Processing*, vol. 25, no. 12, pp. 5933-5942, 2016.

[30] B. Mirzasoleiman, A. Badanidiyuru, A. Karbasi, J. Vondrak and A. Krause, "Lazier than lazy greedy," *In Proceedings of AAAI*, pp. 1812-1818, 2015.

[31] J. Zheng, Z. Jiang, R. Chellappa and J. P. Phillips, "Submodular attribute selection for action recognition in video," *In Proceedings of NIPS*, pp. 1341-1349, 2014.

[32] J. Shen, J. Peng, X. Dong, L. Shao, and F. Porikli, "Higher-order energies for image segmentation," *IEEE Trans. on Image Processing*, vol. 26, no. 10, pp. 4911-4922, 2017.

[33] F. Zhu, Z. Jiang and L. Shao, "Submodular object recognition," *In Proceedings of IEEE CVPR*, pp. 2457-2464, 2014.

[34] W. Wang, J. Shen, and H. Lin, "A deep network solution for attention and aesthetics aware photo cropping," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2018

[35] K. Wei, R. Iyer and J. Bilmes, "Fast multi-stage submodular maximization," *In Proceedings of ICML*, pp. 1494-1502, 2014.

[36] P. Ochs, J. Malik and T. Brox, "Segmentation of moving objects by long term video analysis," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 36, no. 6, pp. 1187-1200, 2014.

[37] J. Shen, D. Yu, L. Deng, and X. Dong, "Fast online tracking with detection refinement," *IEEE Trans. on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 162-173, 2018.

[38] M. Y. Liu, O. Tuzel, S. Ramalingam and R. Chellappa, "Entropy-rate clustering: Cluster analysis via maximizing a submodular function subject to a matroid constraint," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 36, no. 1, pp. 99-112, 2014.

[39] P. Ji, H. Li, M. Salzmann and Y. Dai, "Robust motion segmentation with unknown correspondences," *In Proceedings of ECCV*, pp. 204-219, 2014.

[40] G. Zhong, B. Li, J. Cao and Z. Su, "Object Level Saliency by Submodular Optimization," *In Proceedings of IEEE International Conference on Digital Home*, pp. 105-110, 2014.

[41] F. Yang, Z. Jiang and L. S. Davis, "Submodular reranking with multiple feature modalities for image retrieval," Computer Vision–ACCV 2014, Springer International Publishing, pp. 19-34, 2015.

[42] J. Xu, L. Mukherjee, Y. Li and J. Warner, "Gaze-enabled egocentric video summarization via constrained submodular maximization," *In Proceedings of IEEE CVPR*, pp. 2235-2244, 2015.

[43] M. Gygli, H. Grabner, and G. L. Van, "Video summarization by learning submodular mixtures of objectives," *In Proceedings of IEEE CVPR*, pp. 3090–3098, 2015.

[44] J. Shen,J. Peng, and L. Shao, "Submodular trajectories for better motion segmentation in videos," *IEEE Trans. on Image Processing*, 27(6):2688-2700, 2018.

[45] M. Keuper, B. Andres, and T. Brox, "Motion trajectory segmentation via minimum cost multicuts," *In Proceedings of IEEE ICCV*, pp. 3271-3279, 2015.

[46] E. Balkanski and Y. Singer. Minimizing a submodular function from samples. In *Advances in Neural Information Processing Systems*, pages 814–822, 2017.

[47] Y.-M. Cheung, F. Gu, and H.-L. Liu. Objective extraction for many-objective optimization problems: Algorithm and test problems. *IEEE Trans. on Evolutionary Computation*, 20(5):755–772, 2016.

[48] W. Wang, J. Shen, F. Porikli, and R. Yang, Semi-supervisedvideo object segmentation with super-trajectories, IEEETrans. on Pattern Analysis and Machine Intelligence, 2018.

[49] X. Liu, S.-J. Peng, Y.-m. Cheung, Y. Y. Tang, and J.-X. Du. Active contours with a joint and region-scalable distribution metric for interactive natural image segmentation. *IET Image Processing*, 8(12):824–832, 2014.

[50] C. Qian, J.-C. Shi, K. Tang, and Z.-H. Zhou. Constrained monotone k-submodular function maximization using multi-objective evolutionary algorithms with theoretical guarantee. *IEEE Trans. on Evolutionary Computation*, 2017.

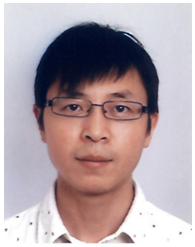
Jianbing Shen (M'11-SM'12) is a Professor with the School of Computer Science, Beijing Institute of Technology, Beijing, China. He is also acting as the Lead Scientist at the Inception Institute of Artificial Intelligence, Abu Dhabi, United Arab Emirates. He received his Ph.D. from the Department of Computer Science, Zhejiang University in 2007. He has published more than 100 top journal and conference papers, six papers are selected as the ESI Hightly Cited or ESI Hot Papers. His current research interests are in the areas of deep learning for video analysis, computer vision for autonomous driving, deep reinforcement learning, and machine learning for intelligent systems.

Prof. Shen is a Senior Member of IEEE. He obtained many flagship honors including the Fok Ying Tung Education Foundation from Ministry of Education, the Program for Beijing Excellent Youth Talents from Beijing Municipal Education Commission, and the Program for New Century Excellent Talents from Ministry of Education. He is an Associate Editor of *IEEE Transactions on Neural Networks and Learning Systems*, *Neurocomputing*, *the Visual Computer*, and other journals.

Xingping Dong received the B.S. degree in computational mathematics and the second B.S. degree in computer science and technology from Xiamen University. He is currently working toward the Ph.D. degree in the School of Computer Science, Beijing Institute of Technology, Beijing, China. His current research interests include deep reinforcement learning and visual object tracking algorithms.

Jianteng Peng is currently pursuing the PhD degree in the School of Computer Science, Beijing Institute of Technology, Beijing, China. His current research interests include submodular optimization and high-order energy optimization.

Xiaogang Jin is a Professor in the State Key Laboratory of CAD&CG, Zhejiang University. He received the B.Sc. degree in computer science and the M.Sc. and Ph.D degrees in applied mathematics from Zhejiang University, P. R. China, in 1989, 1992, and 1995, respectively. His current research interests include digital geometry processing, geometric modeling, 3D printing, virtual try-on, insect swarm simulation, traffic simulation, implicit surface modeling and applications, creative modeling, sketch-based modeling, and image processing. He received an ACM Recognition of Service Award in 2015. He is a member of the IEEE and ACM.



Ling Shao (M'09-SM'10) is the Chief Scientist of the Inception Institute of Artificial Intelligence, Abu Dhabi, United Arab Emirates. He is an Associate Editor of *IEEE Transactions on Image Processing*, *IEEE Transactions on Neural Networks and Learning Systems*, *IEEE Transactions on Circuits and Systems for Video Technology*, and other journals. His research interests include Computer Vision, Machine Learning and Medical Imaging. He is a Fellow of the IAPR, the IET and the BCS.

Fatih Porikli is an IEEE Fellow and a Professor with the Research School of Engineering, Australian National University, Canberra, ACT, Australia. He is also acting as the Chief Scientist at Huawei, Santa Clara. He received the Ph.D. degree from NYU, New York, NY, USA, in 2002. He has contributed broadly to object detection, motion estimation, tracking, image-based representations, and video analytics. He is an Associate Editor of five journals including *IEEE Signal Processing Magazine*, *IEEE Trans. on Multimedia*, *SIAM Imaging Sciences*, and *Springer Journal on Machine Vision Applications*.