ORIGINAL RESEARCH

Higher-order clique based image segmentation using evolutionary game theory

Jing Li, Gang Zeng*, Rui Gan, Hongbin Zha, Long Wang

Key Laboratory of Machine Perception, Peking University, Beijing, China

Received: December 2, 2013 **Accepted:** February 19, 2014 **Online Published:** March 10, 2014
DOI: 10.5430/air.v3n2p1 **URL:** http://dx.doi.org/10.5430/air.v3n2p1

Abstract

This paper describes a novel algorithm for labeling problems of image segmentation. Beyond the pairwise model, our proposed method enables exploration on cliques, which are able to capture rich information of the scene. However, the dilemma is that, while our objective is to assign each pixel a label, the cliques are only limited to work on sets of pixels. To address this problem, the interaction between pixel and clique is studied. The labeling problem is solved using iterative scheme incorporating Expectation-Maximization (EM) algorithm that: in the E step, we would like to estimate labeling preference of pixels from clique potentials with known labeling distribution; and then update clique probabilities in the M step. We optimize the proposed function in the framework of evolutionary game theory, where the Public Goods game (PGG) is employed. Taking the advantage of large size cliques, our algorithm is able to solve multi-label segmentation problem with effective and efficiency. Quantitative evaluation and qualitative results show that our method outperforms the state-of-art. Especially, we apply the proposed algorithm on urban scene segmentation, which aims at segmenting geometric inconsistent objects via vertical assumption. We believe that our algorithm can extend to many other labeling problems.

Key Words: Image segmentation, Higher-order clique, Expectation-maximization, Evolutionary game theory, Public goods game

1 Introduction

In recent years, with the emergence of discrete optimization, many low-level computer vision problems are solved via energy minimization algorithms, such as graph cuts,[1,2] tree-reweighted message passing[3,4] and belief propagation.[5,6] These algorithms allow us to perform approximate inference on graphical models, i.e., by maximizing a posterior probability on Markov Random Fields. Applications of these energy minimization methods include image segmentation, stereo, denoising, and etc. Within such framework, one usually seeks the labeling L that minimizes the energy

$$E(L) = \sum_{p \in P} D_p(L_p) + \lambda \sum_{(p,q \in N)} V_{p,q}(L_p, L_q) \quad (1)$$

Here, D_p measures labeling preference of pixel p, and $V_{p,q}$ encourages spatial coherence by penalizing discontinuities between neighboring pixels (p, q). The symbol P denotes pixel set, and N stands for set of neighboring pairs. The parameter λ controls strength of smoothness. However, this model assumes that the energy is represented in terms of unary and pairwise potentials, which severely restricts its representational power, as it is too local to capture rich statistics of natural scenes.

More recently, solving energies with higher-order cliques has received lots of attention. A higher-order clique can simply denoted by a set of pixels within image coordinate, i.e., a $M_1 \times M_2$ RGB patch, or superpixels. Higher-order clique potentials have the capability to model complex interactions of random variables. Compared with the pairwise model, experiments[7–14] showed superior results by introducing higher-order cliques, making it essential to find an efficient algorithm to solve higher-order energies. Although many methods have been proposed, the energy forms are

*Correspondence: Gang Zeng; Email: g.zeng@ieee.org; Address: Key Laboratory of Machine Perception, Peking University, Beijing, 100871, China.

simple and specified, which is far behind the need of effectively describing the underlying problem. Many of existing methods simply added a specified clique term to the pairwise energy, and solved the higher-order energies using either moving making algorithms,[7–9, 13] or belief propagation,[11] or message passing.[10, 12] Such inference algorithms are scale exponentially with the size of the maximal clique in the graph.[15, 16] Another dilemma is that, the underlying labeling problem is pixel-wised, while the higher-order potentials work on clique-level. Previous work only explored how the clique impacts pixel labeling. However, we argue that it is essential to find out the inherent relationship between clique and pixels. In this paper we propose an efficient method to explore large cliques in the perspective of evolutionary game theory. We formulate this algorithm using a novel objective function and incorporate it within EM framework. Our method is capable to explore interactions between pixel-wised labeling and the clique potentials. The Public Goods Game[17] is a widely-used model describing a multi-person game. Many researchers use PGG to explore cooperative behaviors in society. Previous work found that the underlying network topology can promote cooperation.[18–20] One rationale behind this phenomenon is that cooperators form clusters on graphs,[21] thus they can easily spread their strategies to the surround, promoting and sustaining cooperation in the entire population. Motivated by this, our intuition for image segmentation is: the clusters of cooperation or defection can somehow represent different segmenting parts in the image.

In this paper we introduce PGG to solve our proposed higher-order functions, by interacting between pixel preferences and clique potentials. Apart from effectiveness and efficiency of our algorithm, we show its applicability on urban scenes. Contribution of this paper includes:

1) We propose a novel objective function that describes the underlying relationship between pixels and higher-order cliques.

2) Our optimization algorithm is able to solve large size cliques on multi-label image segmentation, with effectiveness and efficiency.

3) The image segmentation problem is solved in the EM framework by utilizing PGG in the framework of evolutionary game theory.

4) We apply our method on urban scene segmentation. We are able to detect objects from geometric and reflective inconsistent sources. And our modified similarity measurement favors reconstructing plane areas in urban scenes.

Remainder of this paper is arranged as follows. Sec. 2 introduces the background and related work. Problem statement is described in Sec. 3, followed by the proposed PGG-based optimization method in Sec. 4. Sec. 5 shows experimental results, with Sec. 5.1 describing quantitative evaluation, Sec. 5.2 qualitative results, and application on urban scene segmentation is shown in Sec. 5.3. Concluding remarks are drawn in the end.

2 Related work and background

2.1 Optimization for image segmentation

Image segmentation has long been studied. In recent years, a bulk of work emerges that solves segmentation problem by minimizing a discrete energy, where each pixel is assigned a certain label. Graph cuts[1] employed the min-cut/max-flow algorithms to minimize the proposed energy that consisting a data term and a smoothness term, as shown in Eqn. 1, which is widely used to achieve image segmentation. Kolmogorov et al.[2] provided necessary and sufficient conditions for such energy function. Geometric properties of regions formed by graph cuts were described in.[22] A large variety of interactive segmentation methods based on graph cuts have also been developed these years.[23, 24] In general, none of them is superior to all the others. And some methods may be more suitable for solving particular segmentation problems than others. Sometimes, automatic methods are not sufficient to locate the object. In this sense, interactive methods are better off because they combine user interactions that can easily locate the object. Usually, an interactive graph based segmentation method contains the following steps: 1) calculate user preferences that provide cues by the user and 2) generate an optimal solution according to user preferences. In situations where automatic segmentation is difficult and cannot guarantee correctness and reliability, the interactive methods are best adopted. Among these methods, Ref. 25-27 admitted shape priors into interactive graph cuts, Ref. 28-30 improved running time of such methods, and Ref. 31-32 applied the interactive methods in medical and some other applications. Grabcut[33] by Rother et al. extracted the foreground of an image, by utilizing a bounding box provided by the user that roughly holds the foreground, and then ran graph cuts iteratively. In the random walker algorithm,[34] some pixels should be pre-classified by the user. Then an unclassified pixel is assigned a label when a random walker has been given the greatest probability on traversing first to the classified pixel from the unclassified pixel.

Graph cuts can obtain the optimal solution for binary problems. However when each pixel can be assigned many labels, finding the solution can be computationally expensive. To address this problem, moving making algorithms[1, 2] based on graph cuts emerges, which can efficiently solve multi-label segmentation problem.

The energy form in Eqn. 1 only describes constraints between pixel pairs. In order to capture rich statistics of the image, Zeng et al.[14] introduced a framework to integrate non-local statistics into the higher-order Markov Random Fields, using additional latent variables to represent the intrinsic dimensions of the higher-order cliques. Jain et al.[35] solved the higher-order clustering problem by combining attributes of both decomposition of higher-order similarity measures for use in spectral clustering and explicitly use low-rank matrix representations. Fix et al.[36] focused on the higher-order labeling problem by addressing the sum-of-submodular functions. Semantic segmentation using contex

models is also extended from pairwise relationship between objects to higher-order semantic relations.[37] Ref. 7,9,13 added a clique term in the pairwise model to enforce pixels within clique to take the same label. Usually the clique is a set of pixels. Ref. 38 introduced an interactive segmentation method using non-parametric higher-order learning algorithm. In their method, they designed two quadratic cost functions of pixel and region likelihoods in a multi-layer graph and estimated them simultaneously. Our method is more related to Ref. 38. The main idea of our algorithm is that the property of cliques and pixels can supplement each other, and we iteratively optimize pixel labeling and clique potentials. The main difference from Ref. 38 is that in our method, each pixel is related to multiple cliques, whereas in Ref. 38 each pixel was linked to one specified region.

2.2 Public goods game

In a typical PGG, up to N players can choose either to cooperate or defect. Cooperators each invest a certain amount c into the public good, whereas defectors nothing. The total contribution is then multiplied by an enhancement factor r (e.g. $1 < r < N$) and then equally distributed to all participates. Hence, each defector would get net benefit rkc/N providing k out of N players choose to cooperate, while that for cooperators should be reduced by her investment c. Simple reasoning tells that an individual defecting ends up getting a higher payoff than cooperating in any given mixed group. However, each player gets the possibly maximal payoff had all players cooperated. The best choice for individual and that for the group conflicts with each other, giving rise to the dilemma. Simulation of a typical evolutionary procedure base on spatial PGG goes like this. For simplicity, we consider a population of size on a regular lattice, with each node locates an individual, and links represent possible interacting relationships. In spatial settings, each focal individual together with her direct neighbors defines a group. Initially, half proportion of the population is randomly assigned to be cooperators and the remaining defectors. Whenever playing the game, an individual would participate in all the groups she joins in. The accumulated payoff for each player decides which strategy to choose in the next round. The evolutionary process goes for a finite number of times until the fraction of cooperation in the population maintains stable.

In mathematics, the promotion of cooperation is interpreted as approximate maximization of the total payoff. In this paper, we treat the game as an optimization problem. Besides the underlying network structure, diversity is intensively studied on how to promote cooperation in PGG. Santos et al.[17] explored how diversity influences evolution of cooperation by considering the limited resource one possesses. Wang et al.[39] studied PGG with diverse contribution in finite populations; and Ref. 40-41, from another perspective, studied evolutionary dynamics on diverse distribution. Results show that diversity does promote the emergency of cooperation. However, Watts[42] argued that population structure is often more complex than a single graph can describe. Base on this observation, Ohtsuki et al.[43,44] and Wu et al.[45] both experimented the idea of two different graphs that inhabit in PGG. Our approach is more related to Ref. 46 where a selective investment scheme by encoding diversity in the investment graph is proposed, which varies at different time steps. However, instead of imposing spatial selective investment, in this paper, we adopt selective investment among different graphs. Specifically, we propose to solve the multi-label segmentation problem via games played on multiple parallel networks. So each player would participant in several separate graphs and at each time step, she is forced to act as cooperator at only one of these graphs and defect on all the other ones. To make segmentation boundaries be consistent with image edges, diverse contribution and distribution among different players are also studied.

2.3 Temporal projection

In urban environments, simple geometric assumption provides efficient reconstruction from street view, such as vertical assumption,[47,48] or piecewise planarity,[49,50] leading to applications to GoogleEarch, StreetView, as well as future navigation applications. However, objects that violate the assumed geometry are ubiquitous in urban scenes. These objects may disturb quality of reconstruction, leading to visually unpleasant artifacts and degrading the visual realism of the resulting 3D city model. In our application, we define geometric inconsistent objects like cars, pedestrians, and plants that defy vertical assumption. These objects are detected using temporal projections that can detect inconsistency regions by measuring photo consistency of the received projections per pixel. In general, previous work of using temporal projections is classified as either pairwise technique that the reference compares with each projection independently or overall technique that a background image compares with the reference. Ref. 51-52 falls into the first category. In Ref. 51, Taneja et al. detected geometric changes in an urban environment, by comparing the old geometry with some new images observing its current stat. And in Ref. 52, the authors applied similar method to model dynamic objects in outdoor environments. Yang et al.,[53] on the other hand, estimated a background image using a median filter to detect dynamic changes in the scene. While the overall technique performs well on dynamic scene, it is inferior in the perspective of a static scene. In this paper, we don't consider dynamic scenes, so inconsistency is detected using the pairwise model.

3 Problem statement

Input of our algorithm contains the reference image be segmented, and the corresponding labelling preferences indi-

cating the pixel probability that each pixel belongs to a certain label. For a m label problem, we would like to partition the input image into m non-overlapping parts, with each part a specific label. This labeling problem can be formulated as: we seek the mapping l_i: $\Omega \mapsto \Delta$, which assigns a label l_i to pixel i. The symbol Ω denotes the set of pixels; and Δ is discrete set of labels.

We define π_i^l pixel probability, indicating how much potential pixel i is assigned label l; and q_c^l the clique probability, denoting the probability of clique c that being labeled l. Although cliques are rich on describing the scene, they are unable to handle pixel-level segmentation. On the other hand, pixels are capable to deal with pixel-level optimization problems, but it lacks the ability to embrace higher-order descriptions. To bias both, we propose to iteratively update pixel and clique probability to get an overall optimal labeling result.

3.1 EM framework

Suppose Y is the observed data denoting the input image, having the likelihood $l_0(\theta; Y)$ depending on parameters θ. Here θ denotes probability distribution for each clique. The labeling set Δ represents the latent data, so the complete data is $T = (Y, \Delta)$ with likelihood $l_0(\theta; T)$. Our EM framework to maximize $l_0(\theta; T)$ goes like this:

1) Take initial guesses for the parameter θ with known labeling and pixel probability π_i^l.

2) Expectation step: compute likelihood of pixel probability

$$l_0(\theta; T) = \sum_{i \in \Omega} \sum_{l \in \Delta} \pi_i^l \tag{2}$$

3) Maximization step: determine the new estimation of θ.

4) Iterate steps 2) and 3) until convergence.

3.1.1 Clique probability and pixel probability

Base on Naive Bayes Assumption that properties among pixels are independent, so the probability that clique c being labeled l is formulated as

$$q_c^l = \frac{\sum_{i \in c} \pi_i^l}{\sum_{i \in c} [l_i = l]} \tag{3}$$

The symbol [.] is a binary function that returns 1 when the condition is true. Actually, clique probability on label l is calculated by averaging probabilities of pixels which have the same label, while omitting all the others. The complete form of clique probability respect to all labels is represented as $q_c = \{q_c^1, \cdots, q_c^m\}$. The symbol E_i^l denotes the event that pixel i belongs to label l. Accordingly, we define E_c^l the event that clique c being assigned label l. Thus $\Pr(E_i^l | E_c^l), i \in c$ is the posterior that when the clique is assigned label l, how much probability the pixel i is labeled the same. In other words, the posterior indicates the clique influence on the pixel of choosing a certain label. According to Bayesian rule, the posterior is proportional to the product of likelihood and prior that $\Pr(E_i^l | E_c^l) \propto \Pr(E_c^l | E_i^l) \Pr(E_i^l)$. Here the prior $\Pr(E_i^l)$ corresponds to pixel probability π_i^l in the former iteration. And the likelihood $\Pr(E_c^l | E_i^l)$ denotes how the pixel would influence the clique. Thus the likelihood determines labeling choice of pixel i, which is proportional to pixel probability π_i^l that

$$\pi_i^l \propto \Pr(E_c^l | E_i^l) \propto \frac{\Pr(E_i^l | E_c^l)}{\Pr(E_i^l)} \tag{4}$$

with the log-likelihood

$$\pi_i^l \propto \log \Pr(E_c^l | E_i^l) = r \log \Pr(E_i^l | E_c^l) - \log \Pr(E_i^l) \tag{5}$$

Here the parameter r is defined as a proportional coefficient that controls impact of the clique. In general, when a pixel embody in several cliques, pixel probability respect to all cliques is defined as

$$\pi_i^l = \sum_{i \in c, c \in c_i}^{n} r \log \Pr(E_i^l | E_c^l) - \log \Pr(E_i^l) \tag{6}$$

The symbol c_i is the set of cliques that contains pixel i.

3.2 Clique structure

In this paper we try to define a clique as a $M_1 \times M_2$ patch, which is also called a local window on image coordinate. On the other hand, we notice that superpixel that obtained from unsupervised segmentation methods is more capable to represent object parts. An ideal superpixel usually constitutes a particular set of pixels that belongs to a certain object and have the same label. However, any segmentation from superpixel algorithms cannot guarantee this. When an inaccurate superpixel contains two or more objects, we expect our segmentation boundary can still locate on object edges. We define the feature of a clique as $F_c = \{F_c^l\}$, with respect to different labels $l \in [1, m]$. And F_c^l is calculated by averaging feature of pixels within clique that are labeled l.

3.3 Game perspective

Eqn. 2 is difficult to optimize as it is non-convex. However, we observe that the formulation is somehow similar to the Public Goods Game in the evolutionary perspective. So here we propose a game-theoretic approach to solve this problem, where pixels are denoted as players, and each clique stands for a group of direct neighbors. In a classical PGG, a cooperator receives $rkc/N - c$. Here rkc/N is the amount of distribution from the group, which can be interpreted as the posterior $\Pr(E_i^l | E_c^l)$. The variable r corresponds to the

enhancement factor, representing how the clique would influence behavior of the individuals. In this perspective, the prior probability $\Pr(E_i^l)$ is analogue to wealth c of each player, which denotes the amount of investment from the cooperator. In such case, the net payoff of a cooperator represents the likelihood that decides what strategy to take in the next round. This observation provides a natural link between the proposed objective function and the game. In the game perspective, our objective is to maximize the total accumulate net payoff of each player.

Consider a population of constant size $h \times w$ locates on the image coordinate G, with each pixel represents one player. We segment G into several cliques and we call each clique a group in the language of game theory. Each player would play with her direct neighbors within the group. Different from existing methods, here clique potentials and pixel probabilities are iteratively updated. Classical PGG explores how the strategy evolves; similarly, in this paper we study how labeling evolves during iteration. So at the evolutionary stable state, the strategy set is equal to the labeling set, which is obtained via maximizing the total payoff.

To better interpret our method, we show a metaphor in Tab. 1 revealing the connection between higher-order energy function and our PGG-based objective function. In general, the unary term in higher-order energy provides information of labeling preferences, and the binary term penalizes neighboring pixels of taking different labels. The additional clique term encourages pixels within clique to take the same label, where cliques are usually defined as superpixels. Existing algorithms to optimize large clique energy is difficult and time-consuming. In fact, our higher-order patch is not limited to a neighborhood of size $M_1 \times M_2$ in our formulation. Because player i in group c would not only benefit from her direct $M_1 \times M_2$ neighbors, she would also benefit from her neighboring groups. So theoretically, player i can at most explore information in a $2M_1 \times 2M_2$ neighborhood. However, we prove later in our experiments that our algorithm is both effective and efficient.

Table 1: Metaphor between higher-order energy and our PGG-based objective function.

	PGG function	Energy function
Nodes	Players	Pixels
Cliques	Patch or superpixel	Superpixel
Label	Strategies	Labels
Higher-order	N-person game	Additional clique term
Objective	Max payoff	Min energy

4 PGG-based optimization

Each player in the population will participant in different groups she joins in. For an object-background segmentation problem, it is intuitive that the pure strategies of whether cooperate or defect represents labeling of the corresponding pixel. However, the case for multi-label problem is more complex. Here we suppose the strategy of each player is the combination of pure strategies. For a m label problem, each player has m candidate strategies that corresponds to labeling of each pixel ranging from 1 to m. A vector sizes m is used to represent labeling $l_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ that, when the k^{th} element in l_i is equal to 1 and others are 0, pixel i is labeled k. We denote strategy of player i as $s_i = l_i$ that the symbol 1 means cooperation and 0 defection. That means, each player in the population is forced to participant m parallel PGGs and get m separate payoffs. A more intuitive explanation is: imaging m parallel Gs, with the nodes at the same location represent the same player but with different strategies. We call each graph a layer. And among the candidate strategies, each player would selectively cooperate on one layer, and defect on all the others.

In our modified PGG, there are two factors that may influence the total payoff. One is strategy of each player of whether cooperate or defect. The other is the amount of investment one player contributes. In our problem, strategy is analogy to the latent data , and the amount of contribution corresponds to pixel probability. We would first estimate payoff for each player with known strategy distribution and investment. Here payoff of each player corresponds to pixel probability that guides the pixel to an optimal label. We treat this procedure as the expectation step. In the maximization step, we would update strategies according to their payoff values. This process continues until the evolutionary stable state is reached. Note that this evolutionary process represents the interaction between pixel and clique probability. Specifically, in the expectation step, we would like to estimate pixel probability from clique potentials, which corresponds to investment and distribution procedure in PGG; and in the maximization step, clique probability that how likely the clique is assigned a certain label is estimated using updated strategies.

4.1 Expectation step in PGG

In the expectation step, we would like to estimate pixel probability from clique potentials. Initially, each player is assigned a random strategy. Whenever playing the game, a player would invest an amount to each of her direct neighbors, including herself, if she cooperates; otherwise for a free-rider of contributing none. In classical PGG the amount of investment $w_{i,c}^l$ from cooperator i to group c on layer l is fixed. In our case, we employ diverse investment that different players would invest differently. In our design, the amount of investment of a player $w_{i,c}^l$ is determined by two factors: one is pixel probability π_i^l, and the other is feature similarity $U_{i,c}^l$ to that clique

$$w_{i,c}^l = \pi_i^l + \alpha U_{i,c}^l \tag{7}$$

Here α is a weighting parameter that controls strength of

similarity constraint and

$$U_{i,c}^{l} = \exp(-\frac{\parallel F_i - F_c^l \parallel_2}{\sigma^2}) \tag{8}$$

measures feature consistency between pixel and the clique, where F_i denotes image feature extracted on pixel i and σ controls relative sensitivity of feature difference. Feature similarity encourages pixels with similar appearance to the clique contribute more in order to form clusters. Note that for defectors, $w_{i,c}^{l}$=0.

In the distribution step, each player would receive multiple distributions from different groups she participants. The accumulate payoff of player i on layer l is denoted as

$$\phi_i^l = \sum_{i \in c, c \in c_i} [r \frac{\sum_{k \in c} w_{k,c}^l}{\mid c \mid} - w_{i,c}^l] \tag{9}$$

where $|c|$ is the total number of players in group c. The total investment to the group is multiplied by an enhancement factor r, and then equally distributed to each of the participants. In our problem, we expect segmentation boundaries be consistent with image edges. So despite fair distribution, we consider a more complex situation of diverse distribution, where the wealth group c allocates to player i is inverse proportional to their feature differences. The distribution probability is represented as

$$\frac{e^{-\eta |F_i - F_c^l|}}{\sum_{j \in c} e^{-\eta |F_j - F_c^l|}} \tag{10}$$

Instead of using $1/|c|$. Here η controls sensitivity of feature contrast. Note that when η=0, this formulation degenerates to fair distribution. From game perspective, our objective is to maximize the total payoff for each player, which is denoted as

$$\Phi = \sum_{i \in \Omega} \sum_{l \in \Delta} \phi_i^l \tag{11}$$

Eqn. 13 is equivalent to sum of pixel probabilities in Eqn. 2 in the EM procedure.

4.2 Maximization step in PGG

In the maximization step, we would like to update the parameter θ, which changes with pixel labeling and probability distribution. At each time step, players play the game and get separate payoffs from each layer. Then each player updates their strategies according to their payoffs and goes into the next round of game. This procedure is repeated until convergence. In the updating procedure, our method would simultaneously update strategy set Δ as well as pixel probability.

For strategy updating, player i learns to cooperate on layer if that payoff is higher than any of the other layers'. Eqn. 14 shows strategy updating rule of player i cooperates on layer v that

$$s_i(t+1) = [0, \ldots, 0, 1, 0, \ldots 0], \pi_i^v(t) > \pi_i^u(t), \forall u \neq v \tag{12}$$

where the cooperative strategy v is chosen when the v^{th} element in the strategy vector is 1. At time step t, $\pi_i^v(t)$ represents the accumulate payoff of player i on layer v, and $S_i(t+1)$ the strategy of player i in the next round. It is reasonable to greedily support strategies with higher payoff to survive, because player payoff denotes fitness in the language of game theory, while in our framework, payoff corresponds to pixel probability of assigning different labels.

This strategy updating rule works fine when initial segmentation cues are unambiguous. However, the segmentation cues may not that significant, either using user interactive methods, or from machine learning algorithms. For example, parts of the object may have higher prior probability of belonging to the background. Previous methods are prone to produce unsatisfying results where different objects are assigned the same label, or a single object is segmented into different parts. We expect our method is more capable to handle these problems. Besides strategy updating, we extend the updating rule to simultaneously update pixel probability to make the final result be reasonable. We consider three different sources of pixel probability updating that

$$\pi_i(t+1) = (1-\beta_1-\beta_2)\pi_i^l(t) + \beta_1 \phi_i^l(t+1) + \beta_2 x_i^l(t+1) \tag{13}$$

where t represents time step. ϕ_i^l corresponds to payoff in the current game, and x_i^l denotes feature-based probability for pixel i. β_1 and β_2 are two weighting factors. To calculate x_i^l, we follow the procedure of K-Means.

At time step t, we would first learn color distribution for each segment. Feature-based probability is then calculated via feature distance to the segment center. We define features as color, texture or more complex ones like SIFT or HOG. However, in this paper, our key is not on feature selection, so we employ only color features.

We also consider clique updating that all pixels within clique would learn the average pixel probability from its direct neighbor on condition that: 1) average payoff of neighboring clique is higher and 2) the two cliques are of similar appearance. We define similarity between neighboring cliques as L2 norm on their average feature difference. Once this similarity is within a threshold , the clique updating occurs. The pixel updating can recover from small errors. However, when the initial pixel preference is completely incorrect, the clique updating would work as long as distinguishable features are selected.

When the selected feature is not suitable for the image, feature-based updating may not that convincing. Here we introduce the concept of time-scale that feature-based updating occurs at a fixed frequency. Time-scale used here is reasonable when bad features are used. For example, the RGB color cannot work when the foreground and background are of similar color distribution. In this case, we'd better avoid to use feature-based updating.

5 Experimental results and discussions

This section describes our experiments. For comparative evaluation of our method, pair-wise graph cuts[1] and graph cuts for higher-order potentials defined on superpixels[9] are implemented. We test our algorithm on different datasets. For quantitative evaluation, we use the Segmentation Evaluation Database[54] of up to images of different resolutions. In this dataset, human segmentation is used as the groundtruth. We also test on the FlickrMFC dataset[55] as well as Object Class Recognition Image Database[56] from Microsoft Research in Cambridge (MSRC). The FlickrMFC dataset is aimed at co-segmentation. It consists 14 different groups of images, each of which contains 12 to 20 images. And the MSRC dataset has a total of 591 images of animals, plants, houses, aeroplanes, faces and vehicles. In addition, we apply our algorithm on challenging real scenes, where images of the urban environment are used and we can automatically segment the foreground objects like cars in the image.

5.1 Quantitative evaluation

For quantitative evaluation of the proposed method, we test it using foreground-background segmentation involving human interactions. Users would first specify foreground and background seeds on the image, and then the probability maps for different labels are estimated via color distances using K-Means. We compare our method with both pair-wise graph cuts[1] and higher-order graph cuts.[9] While graph cuts penalizes neighboring pixels of taking different labels, the higher-order graph cuts would further enforce pixels within superpixel take the same label. We also compare our method with interactive higher-order segmentation,[38] where the higher-order formulation imposed the soft label consistency constraint on pixels within superpixels.

We test on all the 200 images from the Segmentation Evaluation Database.[54] For generalization and convenience, parameters for each image are the same. On calculating the probability maps with human provided seeds, the number of foreground/background clusters for K-Means is set to be 4 for each input image. We run this binary segmentation algorithm on 5×5 square patches as well as TurboPixels[60] obtained using different parameters. For investment, the feature similarity constraint $\alpha = 0.1$, and parameter for feature contrast σ^2=0.2. For distribution, we set the feature-based sensitivity $\eta = 3$. For pixel probability updating, the weighting parameters concerning payoff and feature are set $\beta_1 = 0.1$ and $\beta_2 = 0.5$ respectively. For the superpixel-based PGG, the clique updating threshold is set to be TH=0.005. Our algorithm would not terminate unless the average fraction of strategy changes exceed $5e-5$ or the number of iterations reaches 200. In general, a conventional optimization process would go dozens of iterations.

For pair-wise graph cuts,[1] the parameter λ is issential. In practice, one may spend a significant amount of time to search for the best result of the most suitable parameter, and efforts have also been made to study the selection of λ.[38,57] In our experiment, we tested on different parameters of and find that the best result is obtained when λ=0.3 for all images.

Higher-order graph cuts[9] based on higher-order conditional random fields and used higher-order potentials defined on superpixels. These potentials enforce label consistency in image regions and take the form of the robust P^n model. In their framework, graph cuts based move making algorithms are used to perform labeling inference. In our implementation, we use approximately 600 TurboPixels[60] per image for calculating the higher-order potentials.

The interactive higher-order segmentation[38] algorithm is a generative model in non-parametric way, where the graph is constructed with two layers: pixel-based layer and region-based layer. The two layers are linked in the way that: each pixel on the pixel layer is connected to its superpixel on the region layer. Then the soft constraint is enforced using energy function of pixel and region likelihoods. In our implementation, we use an unsupervised image segmentation algorithm called Mean Shift to generate superpixels.

We use precision to measure performance of these algorithms. Here precision counts for the ratio of correctly labeled pixels to the total number of pixels. This measure is proper for region-based segmentation. However, it is not inferior when the user is interested in obtaining accurate segmentation boundary because only a small fraction of pixels lie on object boundaries, a large qualitative improvement in the quality of the segmentation will result in only a small increase when counting pixel-wise accuracy. So similar to,[9] we evaluate the quality of segmentation by counting the number of correctly labeled pixels in the region surrounding the actual object boundary. We compute the accuracy using different widths of the evaluation region. The evaluation regions for some images from the Segmentation Evaluation Database[54] are shown in Fig. 1. The average precision of different segmentation methods is plotted in the graph shown in Fig. 2(a). Higher accuracy is obtained as we increase the trimap width. On the other hand, patch-based PGG and higher-order graph cuts[9] get comparative results, but are inferior to the interactive higher-order segmentation.[38] Note that in our experiment the initial seeds for calculating interactive higher-order seg-

mentation[38] are carefully selected. The superpixel-based PGG works best compared with previous higher-order methods.[9,38] The patch-based PGG is inferior because different superpixels are more representable on describing real object boundaries compared with square patches. Moreover, we notice that the patch-based PGG method works better on natural scenes with pure background (i.e. blue sky), and the superpixel-based method is more capable to solve images with textured background (i.e. ocean with wave or the grassplot). Fig. 2(b) shows precision comparisons on images with pure background, demonstrating effectiveness of the PGG-based method with arbitrary clique structures. For images with pure background, the superpixel-based PGG also outperforms higher-order graph cuts.[9] The superpixel-based PGG tend to favor images with textured background, as shown in Fig. 2(c), because in such complex scene, multiple superpixels can provide boundary information, which helps for better segmentation, while the patches may be a little bit confused by locating real the boundaries.

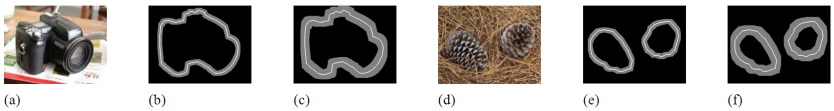

Figure 1: Boundary precision evaluation using trimap segmentations. (a)&(d) shows example images from the MSRC dataset. The remaining images are trimaps used for measuring pixel labelling accuracy. The evaluation region is colored gray and was generated by taking a 6 pixel band (b)&(e) around boundaries of the objects. The corresponding trimaps for an evaluation band width of 12 pixels are shown in (c)&(f).

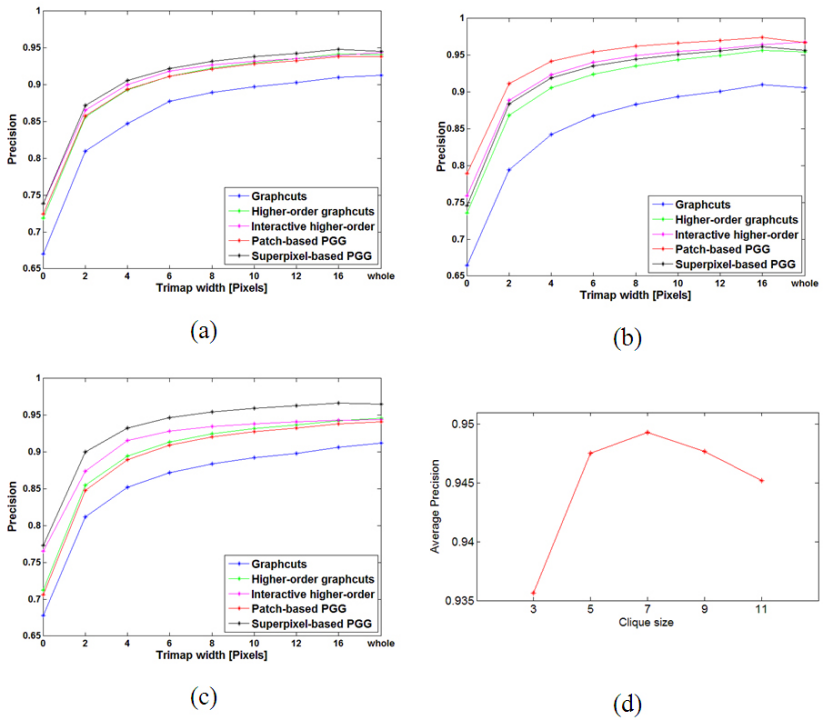

Figure 2: (a)-(c): Quantitative evaluation on Segmentation Evaluation Database[54] of how precision varies as the width of the evaluation region increases. The x-axis is width of the trimap, and the y-axis shows average precision with (a) the whole dataset (b) image set that favors patch-based PGG and (c) image set that favors superpixel-based PGG. (d): Average precision as a function of increasing clique size on all images in the dataset.

To evaluate patch-based PGG, another important parameter is the patch size M. In our experiment, we set $M = M_1 = M_2$. Fig. 2(d) shows how different measuring scores change when varying M. We find that the quality of our algorithm is improved with increasing clique size. However, when clique size exceeds a certain value, the quality may decline. On the one hand, this figure proves that larger clique size can indeed help precise segmentation. On the other hand, we argue that it is not reasonable to set a clique of e large size, especially on low-resolution images. There are two reasons behind this: 1) low-resolution image itself contains insufficient information and 2) extremely large size

cliques may explore disturbing cues and return fallacious results. In our experiment, the average resolution is 277×290, and experimentally the optimal clique size is M=7.

On the one hand, our technique has been designed to accurately segment the object, but is has also been designed with computational efficiency in mind. All of our experiments ran on an Intel(R) Core(TM) i3-2130 CPU, with 8GB available RAM. Tab. 2 shows the average time consumption of different methods. Pair-wise graph cuts runs fast because no clique potential is used, so the results are not that satisfactory. For higher-order exploration, our method is approximately 3 times faster than the other higher-order methods. Furthermore, we also evaluate the computational time when clique size increases. We find that the time consumption in our algorithm is not linearly growing with increasing clique size. In fact, the computational time for a single iteration grows as clique size increases, but the number of iterations for different image varies. A large clique size may lead to quick convergence because of less iteration. Experimentally, the average computational time is 1.53s, 4.15s, 3.08s, 2.29s and 3.59s with respect to patch size of 3, 5, 7, 9 and 11.

Table 2: Average computational time (in seconds) on Segmentation Evaluation Database[54]

Average resolution	Graph cuts [1]	Higher-order graph cuts [9]	Interactive Higher-order [38]	Patch-based PGG	Superpixel-based PGG
277 × 290	0.0713	12.9790	13.7189	4.6457	3.2935

5.2 Qualitative Comparison

Fig. 3 is visualization of the binary segmentation from Segmentation Evaluation Database[54] proving that our method produces visually pleasant and convincing results. Pair-wise graph cuts[1] works fine when we choose the optimal parameter a specific image. In this experiment, we fix the parameter for all the images, which produces unsatisfactory results with over-segmentation in 'boat' image and under-segmentation in 'screw' image. Pair-wise graph cuts[1] fails because of the high reflective surface in 'screw' image and the gradual texture changes of the ocean in 'boat' image. Higher-order graph cuts[9] is superior, however, it explores non-overlapping superpixels, and it still prone to errors at these challenging regions. The PGG-based methods work much better especially using superpixels. The 'dog' image shows an example with fuzz. While the patch-based PGG is superior on getting visually pleasant boundaries, the superpixel-based PGG is more capable on locating object edges. The interactive higher-order segmentation method[38] appears comparable to ours when the initial seeds are carefully marked. However, this task is labor-intensive.

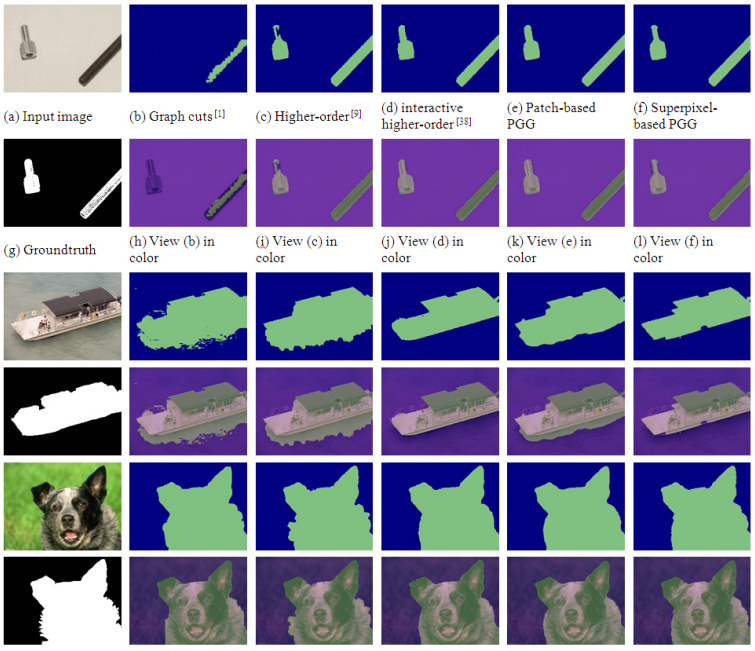

Figure 3: Comparison results on Segmentation Evaluation Database[50] of scene 'screw', 'boat' and 'dog'. For each scene, the first column shows the input image and the groundtruth labeling by the user. The following columns show results using graph cuts,[1] higher-order graph cuts,[9] interactive higher-order segmentation,[38] our proposed patch-based PGG optimization and superpixel-based PGG. The first row shows labeling result that each color represents one label, and the second row shows results viewed on RGB images.

Besides binary segmentation, our framework is also capable to solve multi-label problem. For qualitative comparison on multi-label segmentation, we test our method on the FlickrMFC dataset[55] as well as MSRC dataset.[56]

For MSRC dataset, we adopt TextonBoost[66] to train unary potentials for each class using half of the images in the dataset. We then infer class potentials on the other half.

Fig. 4 shows multi-label segmentation results on MSRC dataset with parameters set to be $\alpha = 0.1, \sigma^2 = 0.3, \eta = 3, \beta_1 = 0.6, \beta_2 = 0.1, TH = 0.003$. Our superpixel-based PGG method outperforms others on two perspectives: 1) the segmentation boundaries align on image edges; and 2) our algorithm can correctly segment the parts when the unary potential is not significant, especially when feature of the object is distant from the background. Results using pairwise graph cuts[1] and higher-order graph cuts[9] cannot get an overall satisfactory result due to incorrect unary potentials, and produces over- or under-segmentation around the mouth in 'goose' image, at the back in 'cow' image, or the left wall in 'urban' image, while the PGG-based method can revises these all. The 'board' image in Fig. 4 shows that it is impossible to segment the board using previous methods because the unary potential is with tremendous large errors. However, by combining different cliques, our method is capable to deal with this. The patch-based PGG produces visually pleasant results with smooth boundaries which look like a human segmentation, and the super-pixel method is prone to follow image edges.

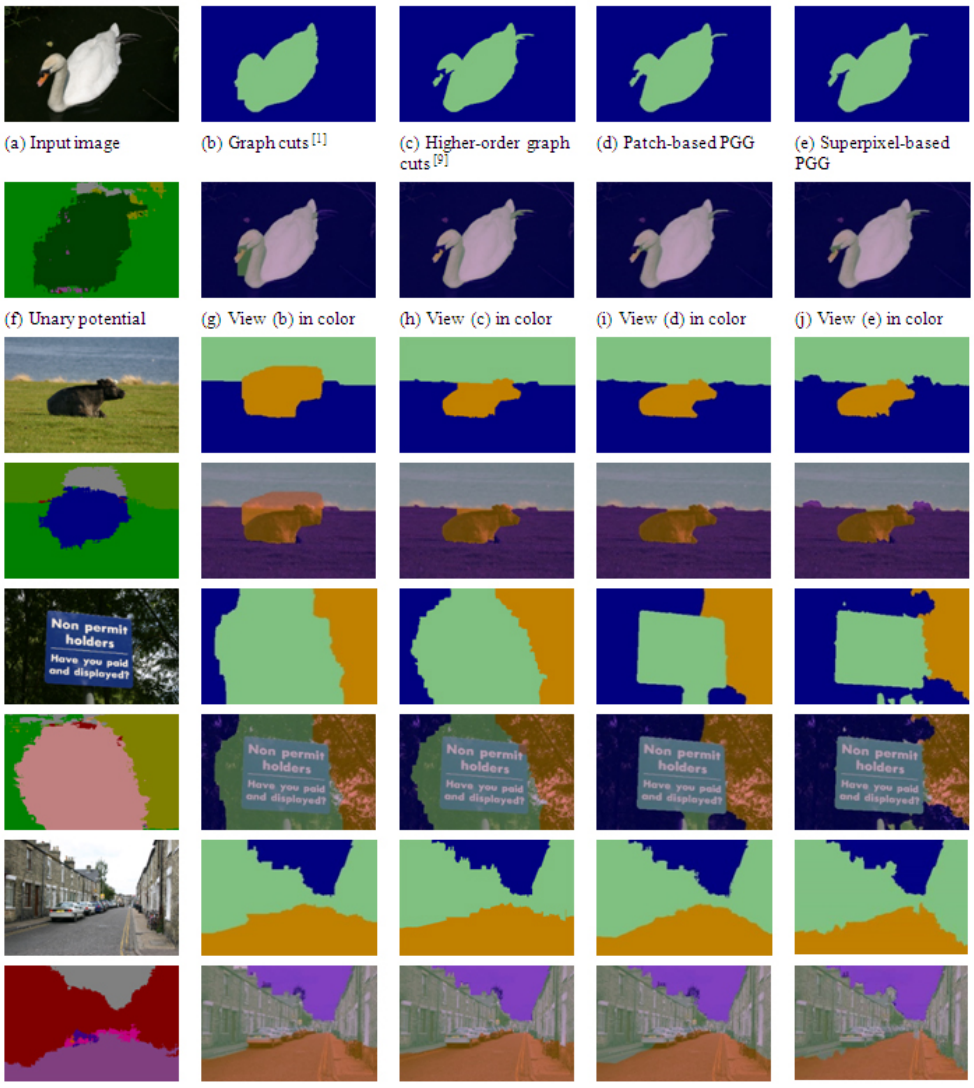

Figure 4: Comparison results on Object Class Recognition Image Database[56] from MSRC of scene 'goose', 'cow', 'board', and 'urban'. For each scene, the first column shows the input image and unary potentials learned from TextonBoost.[66] The following columns show results using graph cuts,[1] higher-order graph cuts,[9] our proposed patch-based PGG optimization and superpixel-based PGG. The first row shows labeling result that each color represents one label, and the second row shows results viewed on RGB images.

To calculate labeling cues on images from FlickrMFC dataset,[55] we use the cosegmentation method,[55] which works within heat diffusion framework, where m finite heat sources corresponded to a m label segmentation that assigns the temperature of every pixel in an image. Here temperature denotes labeling preference. For faster computational speed, labeling preference per pixel is calculated based on superpixels. In our implementation, we use TurboPixels.[60] A graph is then built with the vertex set corresponds to the set of superpixels, and the edge set connects all pairs of adjacent superpixels. For each superpixel, 3-D CIE Lab color and 4-D texture features[61] are extracted. Similarity between neighboring nodes is then computed based on these features. Given a fixed , greedy algorithm is employed to automatically merge the largest and most coherent regions and generate m segmentation cues. For each image, the number of labels m is manually selected.

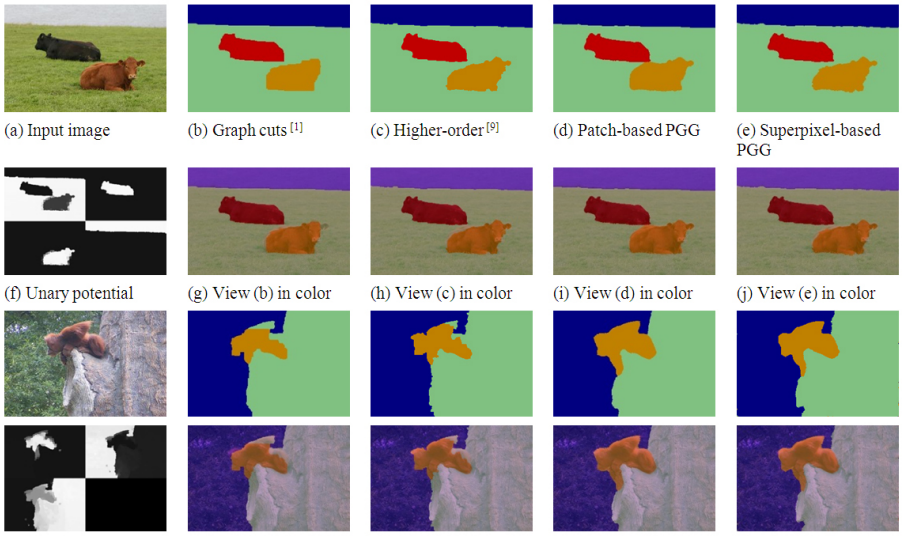

Figure 5: Comparison results on FlickrMFC dataset.[55] For each scene, the first column shows the input image and unary potentials calculated using anisotropic heat diffusion.[59] The following columns show results using graph cuts,[1] higher-order graph cuts,[9] our proposed patch-based PGG optimization and superpixel-based PGG. The first row shows labeling result with respect to different color, and the second row shows results viewed on RGB images.

Fig. 6 shows the results from the FlickrMFC dataset.[55] Under-segmentation occurs when pair-wise graph cuts[1] and higher-order graph cuts[9] are used. We obtain under-segmentation because both segmentation cues and image edges are ambiguous that even superpixels cannot locate at the real object boundary. The PGG-based method works better by exploring more cues from different cliques. The higher-order graph cuts is more sensitive to superpixel edges, while patch-based PGG can usually result in more satisfactory segmentation. Thus the superpixel-based method can somehow be regarded as a combination of higher-order graph cuts and the patch-based PGG.

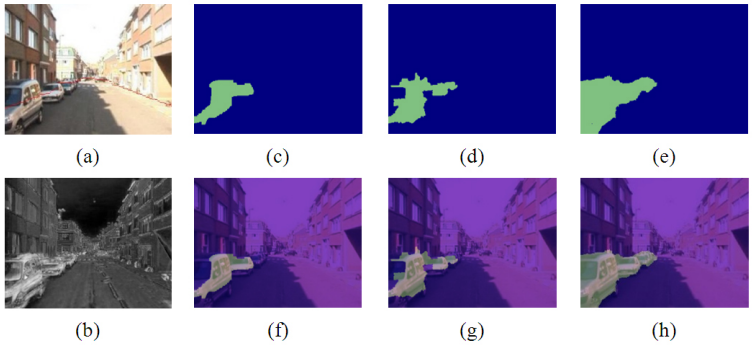

Figure 6: Exemplar results on scene #0300 from Leuven dataset.[65] (a) Rectified input with façade-ground boundary marked red. (b) Geometric inconsistency map of the foreground. We compare our result (e) with pairwise graph cuts[1] (c) and higher-order graph cuts[9] (d). Fig. (f), (g), (h) are blend label with intensity corresponds to (c), (d), (e)

5.3 Urban scene segmentation using PGG

In this experiment our objective is to segment foreground objects in the city, including cars, pedestrians, and plants. The segmentation results can further be used to detailed reconstruction. The inputs of our algorithm are a video sequence or collection of images taken from urban scenes, as well as camera parameters for each image. The output is labeling of the reference image, indicating object and the background. We would first reconstruct the scene using the simplified vertical assumption, followed by the inconsistency detection step via temporal projections to generate the inconsistency map. Finally, the PGG-based optimization is adopted to segment the scene from these inconsistency cues.

5.3.1 Reconstruction

As scene reconstruction could provide cues for image segmentation, via temporal projections, we would first reconstruct simplified 3D model of the scene. Following vertical assumption, each vertical line in space corresponds to a column in the image. However, due to presence of tilt road, or the vertical offset of camera orientation, this correspondence is not always perfectly matched, thus rectification is necessary. Inspired by 62, we first rectify the input images by rotating the camera around its optical center and overlapping the up-directing axis with direction of gravity. We then reconstruct the scene with similar approach as 47.

5.3.2 Inconsistency detection

DetectionOur simplified 3D model is composed of vertical facades and the ground. For the most part, it is intuitive that inconsistency can be detected via photo consistency from temporal projections. There are roughly two sources for such color inconsistency. One is from static objects with Lambertian assumption that violate vertical assumption; the other is from reflective surfaces. We define the former as geometric inconsistency, and the latter reflective inconsistency. Tab. 3 shows theoretical analysis on inconsistency observation from different inconsistency maps. Here 1 denotes regions with high inconsistency value, and 0 otherwise. On the color inconsistency map M_c, we can observe both foreground objects as well as reflective surfaces. Meanwhile, only reflective surfaces is observed on reflective inconsistency map M_r. We denote M_g the geometric inconsistency map, where theoretically, only Lambertian objects can be detected. From Tab. 3, M_g can be obtained from M_c and M_r using XOR operator.

Table 3: Inconsistency observation

	M_c	M_r	$M_g = M_c \wedge \overline{M_r}$
Reflective surface	1	1	0
Background	0	0	0
Lambertian objects	1	0	1

Color inconsistency M_c: Let Γ_0 denotes the reconstructed 3D model from a reference view I_R. We suppose I is the captured image sequence observing Γ_0, except for the reference view. Each observed view $I_t \in I$ is used to render a new image R_t by projecting Γ_0 to I_R. Ideally, if Γ_0 is exact representation of the scene, with all vertical surfaces under Lambertian assumption, I_R and R_t should be identical. In other words, the absolute difference between them should be all zero, where the substraction result M_t defines an inconsistency map. When regions with large difference occur in M_t, we can infer that inconsistency is detected. We assume independence for each inconsistency map, therefore, color inconsistency from temporal projections is calculated as

$$M_c^{i,j}(\Gamma_0) = x = \begin{cases} 0 & n^{i,j} = 0 \\ \prod_{R_t^{i,j} \in P^{i,j}} e^{-M_t^{i,j}} & otherwise \end{cases} \quad (14)$$

where the superscript i, j denotes pixel coordinate, and each pixel may receive several projections $P^{i,j}$ via warping. $n^{i,j}$ represents the number of valid projections in $P^{i,j}$. Eqn. 20 states that, if there is no valid projection, we manually set the corresponding probability $M_c^{i,j}$=0, suggesting larger chance of belonging to the background. On the contrary, inconsistency between the reference view and the rendered view is measured through a normal distribution centered on zero for each independent projection pair. We argue that this pairwise formulation favors inconsistency detection from precious few views.

Reflective inconsistency M_r: As the color inconsistency map M_c may detect both geometric inconsistent objects with Lambertian assumption and reflective surfaces, which remains a problem on how to separate them apart. Theoretically, Lambertian-based objects that betray vertical assumption are, for the most part, occlusions to the background model. These objects would return consistent color among temporal projections as long as the correct geometry is reached. However, for reflective surfaces, temporal projections vary with changing view point, which is independent of the underlying 3D model. Motivated by this observation, we intend to tolerate a narrow band around Γ_0, with 3D points within this narrow band be potential estimation of the scene geometry. For simplicity, we would sample several candidate vertical models within this narrow band, represented as $\Gamma_1, \ldots, \Gamma_h$. Consider a set of 3D points projecting to pixel i, j, we would choose the most consistent one, i.e., $M_r^{i,j} = \min_k M_c^{i,j}(\Gamma_k), k = 0, \ldots, h$. In this way, geometric inconsistency is eliminated, and M_r shows inconsistency map of reflective surfaces.

5.3.3 Results on urban scene segmentation

For urban scene application, we experiment the binary case, by incorporating segmentation cues from geometric inconsistency maps. Our goal is to segment foreground objects

including cars, pedestrians and plants out of the scene. In our experiment we use the Leuven dataset[65] with constant image resolution of 288×360. This dataset was taken by two cameras mounted on a forwarding vehicle. Both cameras face to the front, with their optical centers horizontally located, and their image planes are perpendicular to the ground. The two cameras capture images simultaneously at each time step, which is called a stereo pair. There are totally 2350 stereo pairs, and for each reference image, we use 8 neighboring pairs for temporal projection. When calculating simple geometry of the scene, the maximal disparity d_{max} is set to be 64. For reflective inconsistency detection, each 3D point corresponds to 20 candidates in the narrow band, with each neighboring candidates are three-pixel disparity away. Our probability map is calculated using geometric inconsistency scores. For labeling optimization, we use the patch-based PGG because it is in accordance with most of the structures of man-made scene. As real applications are often much more complex than laboratory images, we argue that small patch sizes is not able to recover from errors. So in all the experiments we adopt a constant patch size of 11×11. For other parameters, we manually set $\alpha = 0.1, \sigma^2 = 0.3, \eta = 3, \beta_1 = 0.1, \beta_2 = 0.1$.

Figure 7: Exemplar results on scene #0300 from Leuven dataset.[65] (a) Rectified input with façade-ground boundary marked red. (b) Geometric inconsistency map of the foreground. We compare our result (e) with pairwise graph cuts[1] (c) and higher-order graph cuts[9] (d). Fig. (f), (g), (h) are blend label with intensity corresponds to (c), (d), (e)

Fig. 6 shows probability maps and segmentation results of scene #0300 from Leuven dataset.[65] The façade-ground boundary is detected by optimizing structure of the scene, which demonstrates effectiveness of this simplified reconstruction. In the foreground geometric inconsistency map, most of the reflective surfaces (i.e., windows) are dark and only occlusion objects with Lambertian reflection show higher probability of belong to the foreground. However,

there are still some misleading cues that may guide the segmentation wrong, makes the graph-based methods unable to segment out the whole object. Higher-order graph cuts[9] works a little bit better than pair-wise graph cuts,[1] however, it is far beyond the need for real applications like detailed reconstruction. Fig. 6(e) shows our result. One the one hand, we can segment out the whole foreground object due to multiple usages of overlapping patches in the image. On the other hand, our segmentation boundaries align with image edges.

Fig. 7 shows more results of urban scene segmentation. While pair-wise graph cuts[1] failed at segmenting the object due to misleading segmentation cues, our method can automatically detect these regions. The higher-order graph cuts[9] base on superpixel cliques, the additional constraints beyond pair-wise graph cuts is that pixels within cliques tend to have the same label. In their method, label of each pixel is determined by only one superpixel instead of multiple cliques, and we argue that single superpixel is usually not enough to describe the scene. Graph-based methods are not able to segment out the car locating at down-right of scene #226 because of occlusions between the car and the background from different viewpoints which results in inconsistent temporal projections. And our method combining updated color distributions generates more satisfactory results. Scene #1852 is more complex because color of the foreground and the background is very similar and our method still works well when we use time-scale. Here we avoid of using color features and the updating relies on payoff of each player. Scene #1304 shows small object that locates far from the camera, thus, temporal projection may fail with small viewpoint changes. Under such challenging situation, our method can also produce a satisfactory result. Note that for pair-wise graph cuts[1] and higher-order graph cuts,[9] we choose the optimal parameters for the displayed results.

6 Conclusion

In this paper we propose a segmentation algorithm within the framework of evolutionary game theory. Our optimization method can efficiently solve functions with higher order cliques to the problem of multi-label segmentation. By interacting between pixel probability and clique potentials, we can get better results compared with previous methods. We also applied our method on urban scene segmentation using geometric cues, which can further assist detailed urban scene reconstruction. Experiments show that our algorithm outperforms the state-of-art. We believe that our method is generic and can be used to solve many other labeling problems.

Acknowledgements

This work is supported by National Natural Science Foundation of China (NSFC) 61375022,National Key Basic Research Program of China (NKBRP) 2011CB302200,Research Fund for the Doctoral Program of Higher Education of China (RFDP) 20100001120023, and National Nature Science Foundation of China (NSFC Grant) 91120004, 61005037, 90920304, 61020106005, 61375120.

References

[1] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," PAMI. 2001; 23(11): 1222–1239. http://dx.doi.org/10.1109/34.969114

[2] V. Kolmogorov and R. Zabin, "What energy functions can be minimized via graph cuts?" PAMI. 2004; 26(2): 147–159. http://dx.doi.org/10.1109/TPAMI.2004.1262177

[3] V. Kolmogorov, "Convergent tree-reweighted message passing for energy minimization," PAMI. 2006; 28(10): 1568–1583. http://dx.doi.org/10.1109/TPAMI.2006.200

[4] M. Wainwright, T. Jaakkola, and A. Willsky, "Tree-based reparameterization for approximate inference on loopy graphs," in NIPS. 2001; 1.

[5] T. Meltzer, C. Yanover, and Y. Weiss, "Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation," in ICCV. 2005; 1: 428–435.

[6] J. Yedidia, W. Freeman, and Y. Weiss, "Generalized belief propagation," in NIPS. MIT. 2001: 689–695.

[7] P. Kohli, M. Kumar, and P. Torr, "P3 & beyond: Solving energies with higher order cliques," in CVPR. IEEE. 2007: 1–8.

[8] P. Kohli, L. Ladicky, and P. Torr, "Graph cuts for minimizing robust higher order potentials," in CVPR. 2008.

[9] P. Kohli, M. Kumar, and P. Torr, "Robust higher order potentials for enforcing label consistency," IJCV. 2009; 82(3): 302–324. http://dx.doi.org/10.1007/s11263-008-0202-0

[10] C. Rother, P. Kohli, W. Feng, and J. Jia, "Minimizing sparse higher order energy functions of discrete variables," in CVPR. IEEE. 2009: 1382–1389.

[11] X. Lan, S. Roth, D. Huttenlocher, and M. Black, "Efficient belief propagation with learned higher-order markov random fields," in ECCV. Springer. 2006: 269–282.

[12] N. Komodakis and N. Paragios, "Beyond pairwise energies: Efficient ptimization for higher-order mrfs," in CVPR. IEEE. 2009: 2985–2992.

[13] L. Ladicky, C. Russell, P. Kohli, and P. Torr, "Associative hierarchical crfs for object class image segmentation," in ICCV. IEEE. 2009: 739–746.

[14] Y. Zeng, C. Wang, S. Soatto, and S.-T. Yau, "Nonlinearly constrained mrfs: Exploring the intrinsic dimensions of higher-order cliques," in CVPR. IEEE, 2013.

[15] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," Foundations and Trends® in Machine Learning. 2008; 1(1-2): 1–305.

[16] T. Werner, "Revisiting the linear programming relaxation approach to gibbs energy minimization and weighted constraint satisfaction," PAMI. 2010; 32(8): 1474–1488. http://dx.doi.org/10.1109/TPAMI.2009.134

[17] F. Santos, M. Santos, and J. Pacheco, "Social diversity promotes the emergence of cooperation in public goods games," Nature. 2008; 454(7201): 213–216.

[18] E. Lieberman, C. Hauert, and M. Nowak, "Evolutionary dynamics on graphs," Nature. 2005; 433: 312–316. http://dx.doi.org/10.1038/nature06940

[19] F. Santos and J. Pacheco, "Scale-free networks provide a unifying framework for the emergence of cooperation," PRL. 2005; 95: 98104. http://dx.doi.org/10.1103/PhysRevLett.95.098104

[20] F. Santos, J. Pacheco, and T. Lenaerts, "Evolutionary dynamics of social dilemmas in structured heterogeneous populations," PNAS. 2006; 103: 3490. http://dx.doi.org/10.1073/pnas.0508201103

[21] Z. Rong, H. Yang, and W. Wang, "Feedback reciprocity mechanism promotes the cooperation of highly clustered scale-free networks," PRE. 2010; 82: 047101. http://dx.doi.org/10.1103/PhysRevE.82.047101

[22] Y. Boykov and V. Kolmogorov, "Computing geodesics and minimal surfaces via graph cuts," in ICCV. IEEE. 2003: 26–33.

[23] Y. Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in nd images," in ICCV. 2001; 1: 105–112.

[24] Y. Boykov and G. Funka-Lea, "Graph cuts and efficient nd image segmentation," IJCV. 2006; 70(2): 109–131. http://dx.doi.org/10.1007/s11263-006-7934-5

[25] D. Freedman and T. Zhang, "Interactive graph cut based segmentation with shape priors," in CVPR. 2005; 1: 755–762.

[26] S. Vicente, V. Kolmogorov, and C. Rother, "Graph cut based image segmentation with connectivity priors," in CVPR. IEEE. 2008: 1–8.

[27] J. Zhang, Y. Wang, and X. Shi, "An improved graph cut segmentation method for cervical lymph nodes on sonograms and its relationship with node's shape assessment," CMIG. 2009; 33(8): 602–607.

[28] H. Lombaert, Y. Sun, L. Grady, and C. Xu, "A multilevel banded graph cuts method for fast image segmentation," in ICCV. 2005; 1: 259–265.

[29] P. Kohli and P. H. Torr, "Dynamic graph cuts for efficient inference in markov random fields," PAMI. 2007; 29(12): 2079–2088. http://dx.doi.org/10.1109/TPAMI.2007.1128

[30] F. R. Schmidt, E. Toppe, and D. Cremers, "Efficient planar graph cuts with applications in computer vision," in CVPR. IEEE. 2009: 351–356.

[31] Y. T. Weldeselassie and G. Hamarneh, "Dt-mri segmentation using graph cuts," in Medical Imaging. International Society for Optics and Photonics. 2007: 65121K–65121K.

[32] J. Malcolm, Y. Rathi, and A. Tannenbaum, "A graph cut approach to image segmentation in tensor space," in CVPR. IEEE. 2007: 1–8.

[33] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," in ACM Transactions on Graphics (TOG). 2004; 23(3): 309–314.

[34] L. Grady, "Random walks for image segmentation," PAMI. 2006; 28(11): 1768–1783. http://dx.doi.org/10.1109/TPAMI.2006.233

[35] S. Jain and V. M. Govindu, "Efficient higher-order clustering on the grassmann manifold," in ICCV. IEEE. 2013.

[36] A. Fix, T. Joachims, S. Park, and R. Zabih, "Structured learning of sum-of-submodular higher order energy functions," in ICCV. IEEE, 2013.

[37] H. Myeong and K. M. Lee, "Tensor-based high-order semantic relation transfer for semantic scene segmentation," in CVPR. IEEE. 2013: 3073–3080.

[38] T. H. Kim, K. M. Lee, and S. U. Lee, "Nonparametric higher-order learning for interactive segmentation", in CVPR, pages 3201–3208. IEEE, 2010.

[39] J. Wang, B. Wu, X. Chen, and L. Wang, "Evolutionary dynamics of public goods games with diverse contributions in finite populations," PRE. 2010; 81(5): 056103. http://dx.doi.org/10.1103/PhysRevE.81.056103

[40] L. Zhong, B. Chen, and C. Huang, "Networking effects on public goods game with unequal allocation," in ICNC. 2008; 1: 217–221.

[41] D. Peng, H. Yang, W. Wang, G. Chen, and B. Wang, "Promotion of cooperation induced by nonuniform payoff allocation in spatial public goods game," EPJ B. 2010; 73(3): 455–459. http://dx.doi.org/10.1140/epjb/e2010-00008-7

[42] D. Watts, "A twenty-first century science," Nature. 2007; 445(7127): 489. http://dx.doi.org/10.1038/445489a

[43] H. Ohtsuki, M. Nowak, and J. Pacheco, "Breaking the symmetry between interaction and replacement in evolutionary dynamics on graphs," PRL. 2007; 98(10): 108106, 2007. http://dx.doi.org/10.1103/PhysRevLett.98.108106

[44] H. Ohtsuki, J. Pacheco, and M. Nowak, "Evolutionary graph theory: breaking the symmetry between interaction and replacement," JTB. 2007; 246(4): 681–694. http://dx.doi.org/10.1016/j.jtbi.2007.01.024

[45] Z. Wu and Y. Wang, "Cooperation enhanced by the difference between interaction and learning neighborhoods for evolutionary spatial prisoners dilemma games," PRE. 2007; 75(4): 041114. http://dx.doi.org/10.1103/PhysRevE.75.041114

[46] J. Li, T. Wu, G. Zeng, and L. Wang, "Selective investment promotes cooperation in public goods game," Physica A, 2012. http://dx.doi.org/10.1016/j.physa.2012.03.016

[47] N. Cornelis, K. Cornelis, and L. Van Gool, "Fast compact city modelling for navigation pre-visualization," in CVPR. 2006; 2: 1339–1344.

[48] M. Pollefeys, D. Nistér, J. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. Kim, P. Merrell et al., "Detailed real-time urban 3d reconstruction from video," IJCV. 2008; 78(2): 143–167. http://dx.doi.org/10.1007/s11263-007-0086-4

[49] S. Sinha, D. Steedly, and R. Szeliski, "Piecewise planar stereo for image-based rendering," in ICCV. 2009: 1881–1888.

[50] B. Micusik and J. Kosecka, "Piecewise planar city 3d modeling from street view panoramic sequences," in CVPR. IEEE. 2009: 2906–2912.

[51] A. Taneja, L. Ballan, and M. Pollefeys, "Image based detection of geometric changes in urban environments," in ICCV. IEEE. 2011: 2336–2343.

[52] A. Taneja, L. Ballan, and M. Pollefeys, "Modeling dynamic scenes recorded with freely moving cameras,"in ACCV. Springer. 2011: 613–626.

[53] W. Yang, G. Zhang, H. Bao, J. Kim, and H. Lee, "Consistent depth maps recovery from a trinocular video sequence," in CVPR. IEEE, 2012.

[54] R. B. Sharon Alpert, Meirav Galun and A. Brandt, "Image segmentation by probabilistic bottom-up aggregation and cue integration," in CVPR, June 2007.

[55] G. Kim and E. P. Xing, "On multiple foreground cosegmentation," in CVPR. IEEE. 2012: 837–844.

[56] J. Winn, A. Criminisi, and T. Minka, "Object categorization by learned universal visual dictionary," in ICCV IEEE. 2005: 2: 1800–1807.

[57] B. Peng and O. Veksler, "Parameter selection for graph cut based image segmentation," in BMVC, 2008.

[58] S. Candemir and Y. Akgül, "Adaptive regularization parameter for graph cut segmentation," IAR. 2010: 117–126.

[59] G. Kim, E. Xing, L. Fei-Fei, and T. Kanade, "Distributed cosegmentation via submodular optimization on anisotropic diffusion," in ICCV. IEEE. 2011: 169–176.

[60] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi, "Turbopixels: Fast superpixels using geometric flows," PAMI. 2009; 31(12): 2290–2297. http://dx.doi.org/10.1109/TPAMI.2009.96

[61] http://www.robots.ox.ac.uk/ vgg/research/texclass/

[62] A. Fusiello, E. Trucco, and A. Verri, "A compact algorithm for rectification of stereo pairs," MVA. 2000; 12(1): 16–22.

[63] A. Flint, D. Murray, and I. Reid, "Manhattan scene understanding using monocular, stereo, and 3d features," in ICCV. IEEE. 2011: 2228–2235.

[64] M. Wainwright, T. Jaakkola, and A. Willsky, "Map estimation via agreement on trees: message-passing and linear programming," TIT. 2005; 51(11): 3697–3717.

[65] B. Leibe, N. Cornelis, K. Cornelis, and L. Van Gool, "Dynamic 3d scene analysis from a moving vehicle," in CVPR. IEEE. 2007: 1–8.

[66] J. Shotton, J. Winn, C. Rother, A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In ECCV. Springer. 2006: 1–15 .