Automatic Foreground Extraction from Imperfect Backgrounds using Multi-Agent Consensus Equilibrium

Xiran Wang, *Student Member, IEEE*, Jason Juang and Stanley H. Chan, *Senior Member, IEEE*

Abstract—Extracting accurate foreground objects from a scene is an essential step for many video applications. Traditional background subtraction algorithms can generate coarse estimates, but generating high quality masks requires professional softwares with significant human interventions, e.g., providing trimaps or labeling key frames. We propose an automatic foreground extraction method in applications where a static but imperfect background is available. Examples include filming and surveillance where the background can be captured before the objects enter the scene or after they leave the scene. Our proposed method is very robust and produces significantly better estimates than state-of-the-art background subtraction, video segmentation and alpha matting methods. The key innovation of our method is a novel information fusion technique. The fusion framework allows us to integrate the individual strengths of alpha matting, background subtraction and image denoising to produce an overall better estimate. Such integration is particularly important when handling complex scenes with imperfect background. We show how the framework is developed, and how the individual components are built. Extensive experiments and ablation studies are conducted to evaluate the proposed method.

Index Terms—Foreground extraction, alpha matting, video matting, Multi-Agent Consensus Equilibrium, background subtraction

1 Introduction

Extracting accurate foreground objects is an essential step for many video applications in filming, surveillance, environment monitoring and video conferencing [1], [2], as well as generating ground truth for performance evaluation [3]. As video technology improves, the volume and resolution of the images have grown significantly over the past decades. Manual labeling has become increasingly difficult; even with the help of industry-grade production softwares, e.g., NUKE, producing high quality masks in large volume is still very time-consuming. The standard solution in the industry has been chroma-keying [4] (i.e., using green screens). However, setting up green screens is largely limited to indoor filming. When moving to outdoor, the cost and manpower associated with the hardware equipment is enormous, not to mention the uncertainty of the background lighting. In addition, some dynamic scenes prohibit the usage of green screens, e.g., capturing a moving car on a road. Even if we focus solely on indoor filming, green screens still have limitations for 360-degree cameras where cameras are surrounding the object. In situations like these, it is unavoidable that the equipments become part of the background.

This paper presents an alternative solution for the aforementioned foreground extraction task. Instead of using a green screen, we assume that we have captured the background image before the object enters or after it leaves the scene. We call this background image as the *plate* image. Using the plate image is significantly less expensive than using the green screens. However, plate images are never perfect. The proposed method is to robustly extract the foreground masks in the presence of imperfect plate images.

X. Wang and S. Chan are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907, USA. Email: {wang470, stanchan}@purdue.edu. *This work is supported, in part, by the National Science Foundation under grants CCF-1763896 and CCF-1718007.*

J. Juang is with HypeVR Inc., San Diego, CA 92108, USA. Email: jason@hypevr.com

1.1 Main Idea

The proposed method, named Multi-Agent Consensus Equilibrium (MACE), is illustrated in Figure 1. At the high level, MACE is an information fusion framework where we construct a strong estimator from several weak estimators. By calling individual estimators as *agents*, we can imagine that the agents are asserting forces to pull the information towards themselves. Because each agent is optimizing for itself, the system is never stable. We therefore introduce a *consensus agent* which averages the individual signals and broadcast the feedback information to the agents. When the individual agents receive the feedback, they adjust their internal states in order to achieve an overall equilibrium. The framework is theoretically guaranteed to find the equilibrium state under mild conditions.

In this paper, we present a novel design of the agents, F_1, F_2, F_3. Each agent has a different task: Agent 1 is an alpha mating agent which takes the foreground background pair and try to estimate the mask using the alpha matting equation. However, since alpha matting sometimes has false alarms, we introduce Agent 2 which is a background estimation agent. Agent 2 provides better edge information and hence the mask. The last agent, Agent 3, is a denoising agent which promotes smoothness of the masks so that there will not be isolated pixels.

As shown in Figure 1, the collection of the agents is the operator $\mathcal{F}$ which takes some signal and updates it. There is $\mathcal{G}$ which is the consensus agent. Its goal is to take

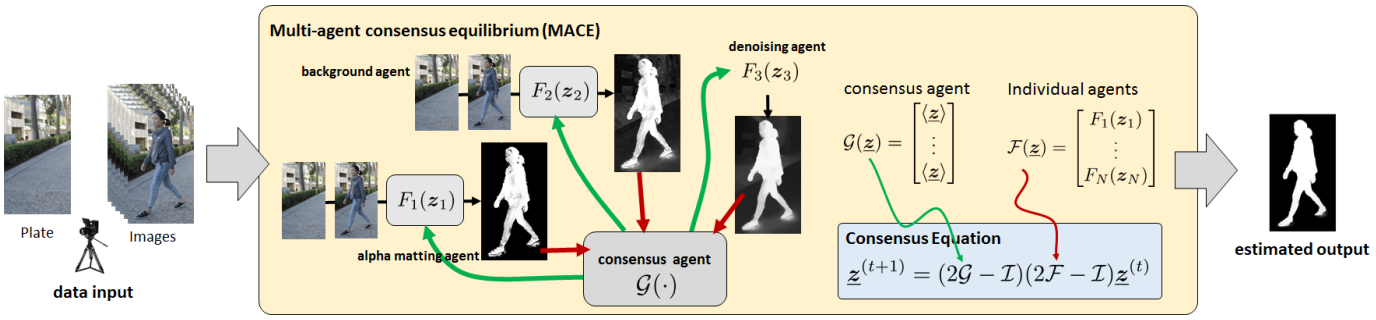


Fig. 1. Conceptual illustration of the proposed MACE framework. Given the input image and the background plate, the three MACE agents individually computes their estimates. The consensus agent aggregates the information and feeds back to the agents to ask for updates. The iteration continues until all the agents reach a consensus which is the final output.

averages of the incoming signals and broadcast the average as a feedback. The red arrows represent signals flowing into the consensus agent, and the green arrows represent signals flowing out to the individual agents. The iteration is given by the consensus equation. When the iteration terminates, we obtain the final estimate.

1.2 Contributions

While most of the existing methods are based on developing a single estimator, this paper proposes to use a consensus approach. The consensus approach has several advantages, which correspond to the contributions of this paper.

- MACE is a training-free approach. We do not require training datasets like those deep learning approaches. MACE is an optimization-based approach. All steps are transparent, explainable, interpretable, and can be debugged.
- MACE is fully automatic. Unlike classical alpha matting algorithms where users need to feed manual scribbles, MACE does not require human in the loop. As will be shown in the experiments, MACE can handle a variety of imaging conditions, motion, and scene content.
- MACE is guaranteed to converge under mild conditions which will be discussed in the theory section of this paper.
- MACE is flexible. While we propose three specific agents for this problem and we have shown their necessity in the ablation study, the number and the type of agents can be expanded. In particular, it is possible to use deep learning methods as agents in the MACE framework.
- MACE offers the most robust result according to the experiments conducted in this paper. We attribute this to the complementarity of the agents when handling difficult situations.

In order to evaluate the proposed method, we have compared against more than 10 state-of-the-art video segmentation and background subtraction methods, including several deep neural network solutions. We have created a database with ground truth masks and background images. The database will be release to the general public on our project website. We have conducted an extensive ablation study to evaluate the importance of individual components.

2 Background

2.1 Related Work

Extracting foreground masks has a long history in computer vision and image processing. Survey papers in the field are abundant, e.g., [8]–[10]. However, there is a subtle but important difference between the problem we are studying in this paper with the existing literature, as illustrated in Figure 2 and Table 1. At the high level, these differences can be summarized in three forms: (1) Quality of the output masks. (2) Requirement of the input (3) Automation and supervision. We briefly describe the comparison with existing methods.

- **Alpha Matting**. Alpha matting is a supervised method. Given a user labeled "trimap", the algorithm uses a linear color model to predict the likelihood of a pixel being foreground or background as shown in Figure 2. A few better known examples include Poisson matting [11], closed-form matting [12], shared matting [13], Bayesian matting [14], and robust matting [15]. More recently, deep neural network based approaches are proposed, e.g., [16], [7], [17] and [18]. The biggest limitation of alpha matting is that the trimaps have to be *error-free*. As soon there is a false alarm of miss in the trimap, the resulting mask will be severely distorted. In video setting, methods such as [19]–[21] suffer similar issues of error-prone trimaps due to temporal propagation. Two-stage methods such as [22] requires initial segmentation [23] to provide the trimap and suffer the same problem. Other methods [24], [25] require additional sensor data, e.g., depth, which is not always available.

- **Background Subtraction**. Background subtraction is unsupervised. Existing background subtraction methods range from the simple frame difference method to the more sophisticated mixture models [26], contour saliency [27], dynamic texture [28], feedback models [5] and attempts to unify several approaches [29]. Most background subtraction methods are used to track objects instead of extracting the alpha mattes. They are fully-automated and are real time, but the foreground masks generated are usually of low quality.
  It is also important to mention that there work about initializing the background image. These methods are particularly useful when the pure background plate image is not available. Bouwmans et al. has a comprehensive

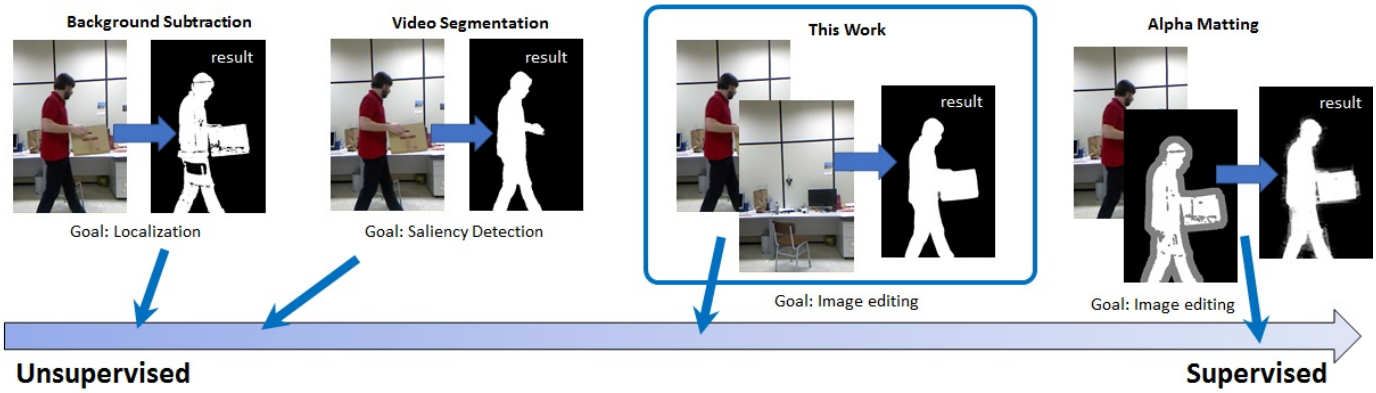


Fig. 2. The landscape of the related problems. We group the methods according to the level of supervision they need during the inference stage. Background Subtraction (e.g., [5]) aims at locating objects in a video, and most of the available methods are un-supervised. Video Segmentation (e.g., [6]) aims at identifying the salient objects. The training stage is supervised, but the inference stage is mostly un-supervised. Alpha matting (e.g., [7]) needs a high-quality trimap to refine the uncertain regions of the boundaries, making it a supervised method. Our proposed method is in the middle. We do not need a high-quality trimap but we assume an imperfect background image.

TABLE 1
Objective and Assumptions of (a) Alpha matting, (b) Background Subtraction, (c) Unsupervised Video Segmentation, and (d) Our work.

Method	Alpha Matting	Bkgnd Subtraction	Video Segmentation	Ours
Goal	foreground estimation	object detection	saliency detection	foreground estimation
Input	image+trimap	video	video	image+plate
Accuracy	high	low (binary)	medium	high
Automatic	semi	full	full	full
Supervised	semi	no	no	semi

survey about the approaches [30]. In addition, a number of methods should be noted, e.g., [31]–[36].

- **Unsupervised Video Segmentation**. Unsupervised video segmentation, as it is named, is unsupervised and fully automatic. The idea is to use different saliency cues to identify objects in the video, and then segments them out. Early approaches include key-segments [37], graph model [38], contrast based saliency [39], motion cues [40], non-local statistics [41], co-segmentation [42], convex-optimization [43]. State-of-the-art video segmentation methods are based on deep neural networks, such as using short connection [44], pyramid dilated networks [45], super-trajectory [46], video attention [6], and feature pyramid [47], [48]. Readers interested in the latest developments of video segmentation can consult the tutorial by Wang et al. [49]. Online tools for video segmentation are also available [50]. One thing to note is that most of the deep neural network solutions require post-processing methods such as conditional random field [51] to fine tune the masks. If we directly use the network output, the results are indeed not good. In contrast, our method does not require any post-processing.

Besides the above mentioned papers, there are some existing methods using the consensus approach, e.g., the early work of Wang and Suter [52], Han et al. [53], and more recently the work of St. Charles et al. [54]. However, the notion of consensus in these papers are more about making votes for the mask. Our approach, on the other hand, are focusing on information exchange across agents. Thus, we are able to offer theoretical guarantees whereas the existing consensus approaches are largely rule-based heuristics.

We emphasize that the problem we study in this paper does *not* belong to any of the above categories. Existing alpha matting algorithms have not been able to handle imperfect plate images, whereas background subtraction is targeting a completely different objective. Saliency based unsupervised video segmentation is an overkill, in particular those deep neural network solutions. More importantly, currently there is no training sets for plate image. This makes learning-based methods impossible. In contrast, the proposed method does not require training. Note that the plate image assumption in our work is the practical reality for many video applications. The problem remains challenging because the plate images are imperfect.

2.2 Challenges of Imperfect Background

Before we discuss the proposed method, we should explain the difficulty of an imperfect plate. If the plate were perfect (i.e., static and matches perfectly with the target images), then a standard frame difference with morphographic operations (e.g., erosion / dilation) would be enough to provide a trimap, and thus a sufficiently powerful alpha matting algorithm would work. When the plate image is imperfect, then complication arises because the frame difference will be heavily corrupted.

The imperfectness of the plate images comes from one or more of the following sources:

- **Background vibration**. While we assume that the plate does not contain large moving objects, small vibration of the background generally exists. Figure 3 Case I shows an example where the background tree vibrates.
- **Color similarity**. When foreground color is very similar to the background color, the trimap generated will have false alarms and misses. Figure 3 Case II shows an example where the cloth of the man has a similar color to the wall.
- **Auto-exposure**. If auto-exposure is used, the background intensity will change over time. Figure 3 Case III shows an example where the background cabinet becomes dimmer when the man leaves the room.

As shown in the examples, error in frame difference can be easily translated to false alarms and misses in the trimap. While we can increase the uncertainty region of the trimap to rely more on the color constancy model of the

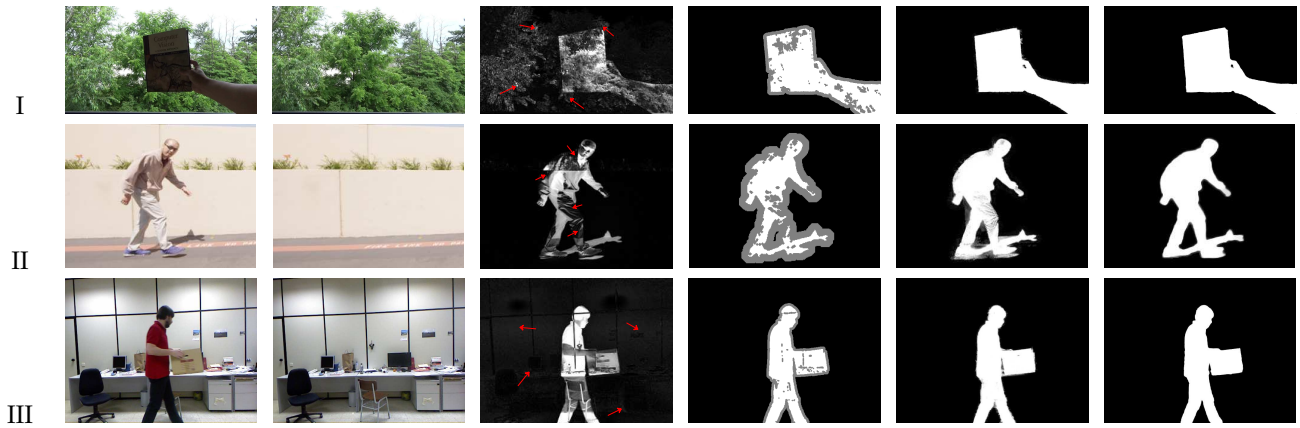


Fig. 3. Three common issues of automatic foreground extraction. Case I: Vibrating background. Notice the small vibration of the leaves in the background. Case II. Similar foreground / background color. Notice the missing parts of the body of the man, and the excessive large uncertainty region of the trimap. Case III. Auto-exposure. Notice the false alarm in the background of the frame difference map. We compare our method with DCNN [7], a semi-supervised alpha matting method using the generated trimaps. The video data of Case III is from [55].

alpha matting, in general the alpha matting performs worse when the uncertainty region grows. We have also tested more advanced background estimation algorithms, e.g., [5] in OpenCV. However, the results are similar or sometimes even worse. Figure 4 shows a comparison using various alpha matting algorithms.

3 Multi-Agent Consensus Equilibrium

Our proposed method is an information fusion technique. The motivation for adopting a fusion strategy is that for complex scenarios, no single estimator can be uniformly superior in all situations. Integrating weak estimators to construct a stronger one is likely more robust and can handle more cases. The weak estimators we use in this paper are the alpha matting, background subtraction and image denoising. We present a principled method to integrate these estimators.

The proposed fusion technique is based on the Multi-Agent Consensus Equilibrium (MACE), recently developed by Buzzard et al. [60]. Recall the overview diagram shown in Figure 1. The method consists of three individual agents which perform specific tasks related to our problem. The agents will output an estimate based on their best knowledge and their current state. The information is aggregated by the consensus agents and broadcast back to the individual agents. The individual agents update their estimates until all three reaches a consensus which is the final output. We will discuss the general principle of MACE in this section, and describe the individual agents in the next section.

3.1 ADMM

The starting point of MACE is the alternating direction method of multipliers (ADMM) algorithm [61]. The ADMM algorithm aims at solving a constrained minimization:

$$\underset{\boldsymbol{x}_1,\boldsymbol{x}_2}{\text{minimize}}\ f_1(\boldsymbol{x}_1)+f_2(\boldsymbol{x}_2),\quad \text{subject to}\quad \boldsymbol{x}_1=\boldsymbol{x}_2, \tag{1}$$

where $\boldsymbol{x}_i \in \mathbb{R}^n$, and $f_i : \mathbb{R}^n \to \mathbb{R}$ are mappings, typically a forward model describing the image formation process and a prior distribution of the latent image. ADMM solves the problem by solving a sequence of subproblems as follows:

$$\boldsymbol{x}_1^{(k+1)} = \underset{\boldsymbol{v}\in\mathbb{R}^n}{\text{argmin}}\ \ f_1(\boldsymbol{v}) + \frac{\rho}{2}\|\boldsymbol{v} - (\boldsymbol{x}_2^{(k)} - \boldsymbol{u}^{(k)})\|^2, \tag{2a}$$

$$\boldsymbol{x}_2^{(k+1)} = \underset{\boldsymbol{v}\in\mathbb{R}^n}{\text{argmin}}\ \ f_2(\boldsymbol{v}) + \frac{\rho}{2}\|\boldsymbol{v} - (\boldsymbol{x}_1^{(k+1)} + \boldsymbol{u}^{(k)})\|^2, \tag{2b}$$

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + (\boldsymbol{x}_1^{(k+1)} - \boldsymbol{x}_2^{(k+1)}). \tag{2c}$$

In the last equation (2c), the vector $\boldsymbol{u}^{(k)} \in \mathbb{R}^n$ is the Lagrange multiplier associated with the constraint. Under mild conditions, e.g., when f_1 and f_2 are convex, close, and proper, global convergence of the algorithm can be proved [62]. Recent studies show that ADMM converges even for some non-convex functions [63].

When f_1 and f_2 are convex, the minimizations in (2a) and (2b) are known as the proximal maps of f_1 and f_2, respectively [64]. If we define the proximal maps as

$$F_i(\boldsymbol{z}) = \underset{\boldsymbol{v}\in\mathbb{R}^n}{\text{argmin}}\ \ f_i(\boldsymbol{v}) + \frac{\rho}{2}\|\boldsymbol{v} - \boldsymbol{z}\|^2, \tag{3}$$

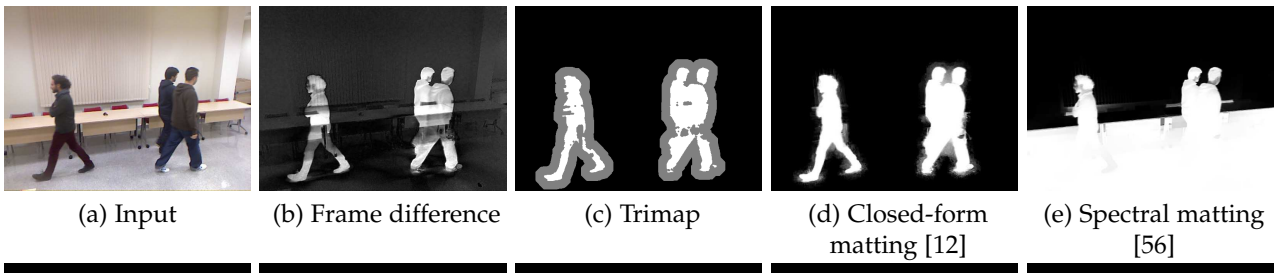
then it is not difficult to see that at the optimal point, (2a) and (2b) become

$$F_1(\boldsymbol{x}^* - \boldsymbol{u}^*) = \boldsymbol{x}^*, \tag{4a}$$

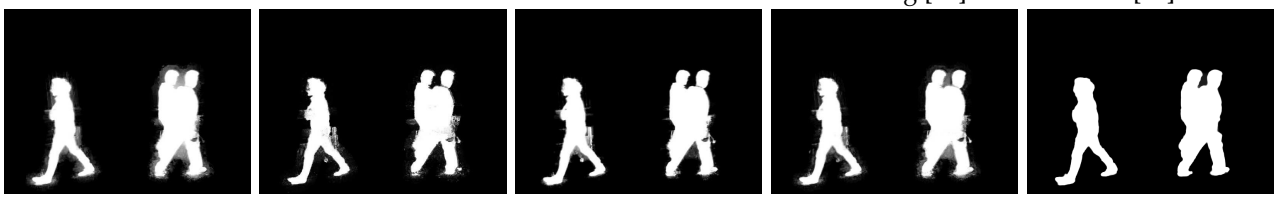
$$F_2(\boldsymbol{x}^* + \boldsymbol{u}^*) = \boldsymbol{x}^*, \tag{4b}$$

where $(\boldsymbol{x}^*, \boldsymbol{u}^*)$ are the solutions to the original constrained optimization in (1). (4a) and (4b) shows that the solution $(\boldsymbol{x}^*, \boldsymbol{u}^*)$ can now be considered as a fixed point of the system of equations.

Rewriting (2a)-(2c) in terms of (4a) and (4b) allows us to consider agents F_i that are not necessarily proximal maps, i.e., f_i is not convex or F_i may not be expressible as optimizations. One example is to use an off-the-shelf image denoiser for F_i, e.g., BM3D, non-local means, or neural network denoisers. Such algorithm is known as the Plug-and-Play ADMM [62], [63], [65] (and variations thereafter [60], [66]).



(a) Input (b) Frame difference (c) Trimap (d) Closed-form matting [12] (e) Spectral matting [56]



(f) Learning-based matting [57] (g) K-nearest neighbors [58] (h) Comprehensive sampling [59] (i) DCNN [7] (j) Ours (without trimap)

Fig. 4. Comparison with existing alpha-matting algorithms on real images with a frame-difference based trimap. (a) Input image. (b) Frame difference. (c) Trimap generated by morphographic operation (dilation / erosion) of the binary mask. (d) - (i) Alpha matting algorithms available on `alphamatting.com`. (j) Proposed method. This sequence is from the dataset of [55].

3.2 MACE and Intuition

MACE generalizes the above ADMM formulation. Instead of minimizing a sum of two functions, MACE minimizes a sum of N functions $f_1, \ldots, f_N$:

$$\underset{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N}{\text{minimize}} \quad \sum_{i=1}^{N} f_i(\boldsymbol{x}_i), \quad \boldsymbol{x}_1 = \ldots = \boldsymbol{x}_N. \tag{5}$$

In this case, the equations in (4a)-(4b) are generalized to

$$\begin{array}{ll} F_i(\boldsymbol{x}^* + \boldsymbol{u}_i^*) & = \boldsymbol{x}^*, \quad \text{for } i = 1, \ldots, N \\ \sum_{i=1}^{N} \boldsymbol{u}_i^* & = 0. \end{array} \tag{6}$$

What does (6) buy us? Intuitively, (6) suggests that in a system containing N agents, each agent will create a tension $\boldsymbol{u}_i^* \in \mathbb{R}^n$. For example, if F_1 is an inversion step whereas F_2 is a denoising step, then F_1 will not agree with F_2 because F_1 tends to recover details but F_2 tends to smooth out details. The agents $F_1, \ldots, F_N$ will reach an equilibrium state where the sum of the tension is zero. This explains the name "consensus equilibrium", as the the algorithm is seeking a consensus among all the agents.

How does the equilibrium solution look like? The following theorem, shown in [60], provides a way to connect the equilibrium condition to a fixed point of an iterative algorithm.

Theorem 1 (MACE solution [60]). *Let* $\underline{\boldsymbol{u}}^* \overset{def}{=} [\boldsymbol{u}_1^*; \ldots; \boldsymbol{u}_N^*]$. *The consensus equilibrium* $(\boldsymbol{x}^*, \underline{\boldsymbol{u}}^*)$ *is a solution to the MACE equation* (6) *if and only if the points* $\boldsymbol{v}_i^* \overset{def}{=} \boldsymbol{x}^* + \boldsymbol{u}_i^*$ *satisfy*

$$\frac{1}{N} \sum_{i=1}^{N} \boldsymbol{v}_i^* = \boldsymbol{x}^* \tag{7}$$

$$(2\mathcal{G} - \mathcal{I})(2\mathcal{F} - \mathcal{I})\underline{\boldsymbol{v}}^* = \underline{\boldsymbol{v}}^*, \tag{8}$$

where $\underline{\boldsymbol{v}}^* \overset{def}{=} [\boldsymbol{v}_1^*; \ldots; \boldsymbol{v}_N^*] \in \mathbb{R}^{nN}$, *and* $\mathcal{F}, \mathcal{G} : \mathbb{R}^{nN} \to \mathbb{R}^{nN}$ *are mappings defined as*

$$\mathcal{F}(\underline{\boldsymbol{z}}) = \begin{bmatrix} F_1(\boldsymbol{z}_1) \\ \vdots \\ F_N(\boldsymbol{z}_N) \end{bmatrix}, \quad \textit{and} \quad \mathcal{G}(\underline{\boldsymbol{z}}) = \begin{bmatrix} \langle \underline{\boldsymbol{z}} \rangle \\ \vdots \\ \langle \underline{\boldsymbol{z}} \rangle \end{bmatrix}, \tag{9}$$

where $\langle \underline{\boldsymbol{z}} \rangle \overset{def}{=} \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{z}_i$ *is the average of* $\underline{\boldsymbol{z}}$.

Algorithm 1 MACE Algorithm

1: Initialize $\underline{\boldsymbol{v}}^t = [\boldsymbol{v}_1^t, \ldots, \boldsymbol{v}_N^t]$.
2: **for** $t = 1, \ldots, T$ **do**
3: % Perform agent updates, $(2\mathcal{F} - \mathcal{I})(\underline{\boldsymbol{v}}^t)$
4:

$$\begin{bmatrix} \boldsymbol{z}_1^t \\ \vdots \\ \boldsymbol{z}_N^t \end{bmatrix} = \begin{bmatrix} 2F_1(\boldsymbol{v}_1^t) - \boldsymbol{v}_1^t \\ \vdots \\ 2F_N(\boldsymbol{v}_N^t) - \boldsymbol{v}_N^t \end{bmatrix} \tag{10}$$

5:
6: % Perform the data aggregation $(2\mathcal{G} - \mathcal{I})(\underline{\boldsymbol{z}}^t)$
7:

$$\begin{bmatrix} \boldsymbol{v}_1^{t+1} \\ \vdots \\ \boldsymbol{v}_N^{t+1} \end{bmatrix} = \begin{bmatrix} 2\langle \underline{\boldsymbol{z}}^t \rangle - \boldsymbol{z}_1^t \\ \vdots \\ 2\langle \underline{\boldsymbol{z}}^t \rangle - \boldsymbol{z}_N^t \end{bmatrix} \tag{11}$$

8: **end for**
9: Output $\langle \underline{\boldsymbol{v}}^T \rangle$.

Theorem 1 provides a full characterization of the MACE solution. The operator $\mathcal{G}$ in Theorem 1 is a consensus agent that takes a set of inputs $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_N$ and maps them to their average $\langle \underline{\boldsymbol{z}} \rangle$. In fact, we can show that $\mathcal{G}$ is a projection and that $(2\mathcal{G} - \mathcal{I})$ is its self-inverse [60]. As a result, (8) is equivalent to $(2\mathcal{F} - \mathcal{I})\underline{\boldsymbol{v}}^* = (2\mathcal{G} - \mathcal{I})\underline{\boldsymbol{v}}^*$. That is, we want the individual agents $F_1, \ldots, F_N$ to match with the consensus agent $\mathcal{G}$ such that the equilibrium holds: $(2\mathcal{F} - \mathcal{I})\underline{\boldsymbol{v}}^* = (2\mathcal{G} - \mathcal{I})\underline{\boldsymbol{v}}^*$.

The algorithm of the MACE is illustrated in Algorithm 1. According to (8), $\underline{\boldsymbol{v}}^*$ is a fixed point of the set of equilibrium equations. Finding the fixed point can be done by iteratively updating $\underline{\boldsymbol{v}}^{(t)}$ through the procedure

$$\underline{\boldsymbol{v}}^{(t+1)} = (2\mathcal{G} - \mathcal{I})(2\mathcal{F} - \mathcal{I})\underline{\boldsymbol{v}}^{(t)}. \tag{12}$$

Therefore, the algorithmic steps are no more complicated than updating the individual agents $(2\mathcal{F} - \mathcal{I})$ in parallel, and then aggregating the results through $(2\mathcal{G} - \mathcal{I})$.

The convergence of MACE is guaranteed when $\mathcal{T}$ is non-expansive [60]summarized in the proposition below.

Proposition 1. *Let $\mathcal{F}$ and $\mathcal{G}$ be defined as* (9)*, and let $\mathcal{T} \stackrel{def}{=} (2\mathcal{G}-\mathcal{I})(2\mathcal{F}-\mathcal{I})$. Then the following results hold:*

(i) $\mathcal{F}$ is firmly non-expansive if all F_i's are firmly non-expansive.
(ii) $\mathcal{G}$ is firmly non-expansive.
(iii) $\mathcal{T}$ is non-expansive if $\mathcal{F}$ and $\mathcal{G}$ are firmly non-expansive.

Proof. See Appendix. □

4 Designing MACE Agents

After describing the MACE framework, in this section we discuss how each agent is designed for our problem.

4.1 Agent 1: Dual-Layer Closed-Form Matting

The first agent we use in MACE is a modified version of the classic closed-form matting. More precisely, we define the agent as

$$F_1(\boldsymbol{z}) = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \ \boldsymbol{\alpha}^T \widetilde{\boldsymbol{L}} \boldsymbol{\alpha} + \lambda_1 (\boldsymbol{\alpha}-\boldsymbol{z})^T \widetilde{\boldsymbol{D}} (\boldsymbol{\alpha}-\boldsymbol{z}), \quad (13)$$

where $\widetilde{\boldsymbol{L}}$ and $\widetilde{\boldsymbol{D}}$ are matrices, and will be explained below. The constant λ_1 is a parameter.

Review of Closed-Form Matting. To understand the meaning of (13), we recall that the classical closed-form matting is an algorithm that tries to solve

$$J(\boldsymbol{\alpha}, \boldsymbol{a}, \boldsymbol{b}) = \sum_{j\in I} \left(\sum_{i\in w_j} \left(\alpha_i - \sum_c a_j^c I_i^c - b_j \right)^2 + \epsilon \sum_c (a_j^c)^2 \right). \quad (14)$$

Here, (a^r, a^g, a^b, b) are the linear combination coefficients of the color line model $\alpha_i \approx \sum_{c\in\{r,g,b\}} a^c I_i^c + b$, and α_i is the alpha matte value of the ith pixel [12]. The weight w_j is a 3×3 window of pixel j. With some algebra, we can show that the marginalized energy function $J(\boldsymbol{\alpha}) \stackrel{\text{def}}{=} \min_{\boldsymbol{a},\boldsymbol{b}} J(\boldsymbol{\alpha}, \boldsymbol{a}, \boldsymbol{b})$ is equivalent to

$$J(\boldsymbol{\alpha}) \stackrel{\text{def}}{=} \min_{\boldsymbol{a},\boldsymbol{b}} J(\boldsymbol{\alpha}, \boldsymbol{a}, \boldsymbol{b}) = \boldsymbol{\alpha}^T \boldsymbol{L} \boldsymbol{\alpha}, \quad (15)$$

where $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ is the so-called matting Laplacian matrix. When trimap is given, we can regularize $J(\boldsymbol{\alpha})$ by minimizing the overall energy function:

$$\widehat{\boldsymbol{\alpha}} = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \ \boldsymbol{\alpha}^T \boldsymbol{L} \boldsymbol{\alpha} + \lambda (\boldsymbol{\alpha}-\boldsymbol{z})^T \boldsymbol{D} (\boldsymbol{\alpha}-\boldsymbol{z}), \quad (16)$$

where $\boldsymbol{D}$ is a binary diagonal matrix with entries being one for pixels that are labeled in the trimap, and zero otherwise. The vector $\boldsymbol{z} \in \mathbb{R}^n$ contains specified alpha values given by the trimap. Thus, for large λ, the minimization in (16) will force the solution to satisfy the constraints given by the trimap.

Dual-Layer Matting Laplacian $\widetilde{\boldsymbol{L}}$. In the presence of the plate image, we have two pieces of complementary information: $\boldsymbol{I} \in \mathbb{R}^{n\times 3}$ the color image containing the foreground object, and $\boldsymbol{P} \in \mathbb{R}^{n\times 3}$ the plate image. Correspondingly, we have alpha matte $\boldsymbol{\alpha}^I$ for $\boldsymbol{I}$, and the alpha matte $\boldsymbol{\alpha}^P$ for $\boldsymbol{P}$. When $\boldsymbol{P}$ is given, we can redefine the color line model as

$$\begin{bmatrix} \alpha_i^I \\ \alpha_i^P \end{bmatrix} \approx \sum_{c\in\{r,g,b\}} a^c \begin{bmatrix} I_i^c \\ P_i^c \end{bmatrix} + b. \quad (17)$$

In other words, we ask the coefficients (a^r, a^g, a^b, b) to fit simultaneously the actual image $\boldsymbol{I}$ and the plate image $\boldsymbol{P}$. When (17) is assumed, the energy function $J(\boldsymbol{\alpha}, \boldsymbol{a}, \boldsymbol{b})$ becomes

$$\widetilde{J}(\boldsymbol{\alpha}^I, \boldsymbol{\alpha}^P, \boldsymbol{a}, \boldsymbol{b}) = \sum_{j\in I} \Bigg\{ \sum_{i\in w_j} \left(\alpha_i^I - \sum_c a_j^c I_i^c - b_j \right)^2 + \eta \sum_{i\in w_j} \left(\alpha_i^P - \sum_c a_j^c P_i^c - b_j \right)^2 + \epsilon \sum_c (a_j^c)^2 \Bigg\}, \quad (18)$$

where we added a constant η to regulate the relative emphasis between $\boldsymbol{I}$ and $\boldsymbol{P}$.

Theorem 2. *The marginal energy function*

$$\widetilde{J}(\boldsymbol{\alpha}) \stackrel{def}{=} \min_{\boldsymbol{a},\boldsymbol{b}} \widetilde{J}(\boldsymbol{\alpha}, \boldsymbol{0}, \boldsymbol{a}, \boldsymbol{b}) \quad (19)$$

can be equivalently expressed as $\widetilde{J}(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T \widetilde{\boldsymbol{L}} \boldsymbol{\alpha}$, where $\widetilde{\boldsymbol{L}} \in \mathbb{R}^{n\times n}$ is the modified matting Laplacian, with the (i,j)th element

$$\widetilde{L}_{i,j} = \sum_{k|(i,j)\in w_k} \Bigg\{ \delta_{ij} - \frac{1}{2|w_k|} \Bigg(1 + (\boldsymbol{I}_i - \boldsymbol{\mu}_k)^T \left(\boldsymbol{\Sigma}_k - n(1+\eta)\boldsymbol{\mu}_k \boldsymbol{\mu}_k^T \right)^{-1} (\boldsymbol{I}_j - \boldsymbol{\mu}_k) \Bigg) \Bigg\}. \quad (20)$$

Here, δ_{ij} is the Kronecker delta, $\boldsymbol{I}_i \in \mathbb{R}^3$ is the color vector at the ith pixel. The vector $\boldsymbol{\mu}_k \in \mathbb{R}^3$ is defined as

$$\boldsymbol{\mu}_k = \frac{1}{2|w_k|} \sum_{j\in w_k} (\boldsymbol{I}_j + \boldsymbol{P}_j), \quad (21)$$

and the matrix $\boldsymbol{\Sigma}_k \in \mathbb{R}^{3\times 3}$ is

$$\boldsymbol{\Sigma}_k = \frac{1}{2} \Bigg\{ \frac{1}{|w_k|} \sum_{j\in w_k} (\boldsymbol{I}_j - \boldsymbol{\mu}_k)(\boldsymbol{I}_j - \boldsymbol{\mu}_k)^T + \frac{1}{|w_k|} \sum_{j\in w_k} (\boldsymbol{P}_j - \boldsymbol{\mu}_k)(\boldsymbol{P}_j - \boldsymbol{\mu}_k)^T \Bigg\}. \quad (22)$$

Proof. See Appendix. □

Because of the plate term in (18), the modified matting Laplacian $\widetilde{\boldsymbol{L}}$ is positive definite. See Appendix for proof. The original $\boldsymbol{L}$ in (15) is only positive semi-definite.

Dual-Layer Regularization $\widetilde{\boldsymbol{D}}$. The diagonal regularization matrix $\widetilde{\boldsymbol{D}}$ in (13) is reminiscent to the binary matrix $\boldsymbol{D}$ in (16), but $\widetilde{\boldsymbol{D}}$ is defined through a sigmoid function applied to the input $\boldsymbol{z}$. To be more precise, we define $\widetilde{\boldsymbol{D}} \stackrel{\text{def}}{=} \operatorname{diag}(\widetilde{d}_i)$, where

$$\widetilde{d}_i = \operatorname{diag} \left\{ \frac{1}{1+\exp\{-\kappa(z_i-\theta)\}} \right\} \in \mathbb{R}^{n\times n}, \quad (23)$$

and z_i is the i-th element of the vector $\boldsymbol{z} \in \mathbb{R}^n$, which is the argument of F_1. The scalar constant $\kappa > 0$ is a user defined parameter specifying the stiffness of the sigmoid function,

and $0 < \theta < 1$ is another user defined parameter specifying the center of the transient. Typical values of (κ, θ) for our MACE framework are $\kappa = 30$ and $\theta = 0.8$.

A closer inspection of $\boldsymbol{D}$ and $\widetilde{\boldsymbol{D}}$ reveals that $\boldsymbol{D}$ is performing a hard-threshold whereas $\widetilde{\boldsymbol{D}}$ is performing a soft-threshold. In fact, the matrix $\boldsymbol{D} \overset{\text{def}}{=} \text{diag}(d_i)$ has diagonal entries

$$d_i = \begin{cases} 0, & \theta_1 < z_i < \theta_2, \\ 1, & \text{otherwise.} \end{cases} \tag{24}$$

for two cutoff values θ_1 and θ_2. This hard-threshold is equivalent to the soft-threshold in (23) when $\kappa \to \infty$.

There are a few reasons why (23) is preferred over (24), especially when we have the plate image. First, the soft-threshold in (23) tolerates more error present in $\boldsymbol{z}$, because the values of $\widetilde{\boldsymbol{D}}$ represent the probability of having foreground pixels. Second, the one-sided threshold in (23) ensures that the background portion of the image is handled by the plate image rather than the input $\boldsymbol{z}$. This is usually beneficial when the plate is reasonably accurate.

4.2 Agent 2: Background Estimator

Our second agent is a background estimator, defined as

$$F_2(\boldsymbol{z}) = \underset{\boldsymbol{\alpha}}{\text{argmin}} \quad \|\boldsymbol{\alpha} - \boldsymbol{r}_0\|^2 + \lambda_2 \|\boldsymbol{\alpha} - \boldsymbol{z}\|^2 + \gamma \boldsymbol{\alpha}^T (1 - \boldsymbol{\alpha}). \tag{25}$$

The reason of introducing F_2 is that in F_1, the matrix $\widetilde{\boldsymbol{D}}$ is determined by the current estimate $\boldsymbol{z}$. While $\widetilde{\boldsymbol{D}}$ handles part of the error in $\boldsymbol{z}$, large missing pixels and false alarms can still cause problems especially in the interior regions. The goal of F_2 is to complement F_1 for these interior regions.

Initial Background Estimate $\boldsymbol{r}_0$. Let us take a look at (25). The first two terms are quadratic. The interpretation is that given some fixed initial estimate $\boldsymbol{r}_0$ and the current input $\boldsymbol{z}$, $F_2(\boldsymbol{z})$ returns a linearly combined estimate between $\boldsymbol{r}_0$ and $\boldsymbol{z}$. The initial estimate $\boldsymbol{r}_0$ consists of two parts:

$$\boldsymbol{r}_0 = \boldsymbol{r}_c \odot \boldsymbol{r}_e, \tag{26}$$

where $\odot$ means elementwise multiplication. The first term $\boldsymbol{r}_c$ is the *color* term, measuring the similarity between foreground and background colors. The second term $\boldsymbol{r}_e$ is the *edge* term, measuring the likelihood of foreground edges relative background edges. In the followings we will discuss these two terms one by one.

Defining the Color Term $\boldsymbol{r}_c$. We define $\boldsymbol{r}_c$ by measuring the distance $\|\boldsymbol{I}_j - \boldsymbol{P}_j\|^2 = \sum_{c \in \{r,g,b\}} (I_j^c - P_j^c)^2$ between a color pixel $\boldsymbol{I}_j \in \mathbb{R}^3$ and a plate pixel $\boldsymbol{P}_j \in \mathbb{R}^3$. Ideally, we would like $\boldsymbol{r}_c$ to be small when $\|\boldsymbol{I}_j - \boldsymbol{P}_j\|^2$ is large.

In order to improve the robustness of $\|\boldsymbol{I}_j - \boldsymbol{P}_j\|^2$ against noise and illumination fluctuation, we modify $\|\boldsymbol{I}_j - \boldsymbol{P}_j\|^2$ by using the bilateral weighted average over a small neighborhood:

$$\Delta_i = \sum_{j \in \Omega_i} w_{ij} \|\boldsymbol{I}_j - \boldsymbol{P}_j\|^2, \tag{27}$$

where Ω_i specifies a small window around the pixel i. The bilateral weight w_{ij} is defined as

$$w_{ij} = \frac{\widetilde{w}_{ij}}{\sum_j \widetilde{w}_{ij}}, \tag{28}$$

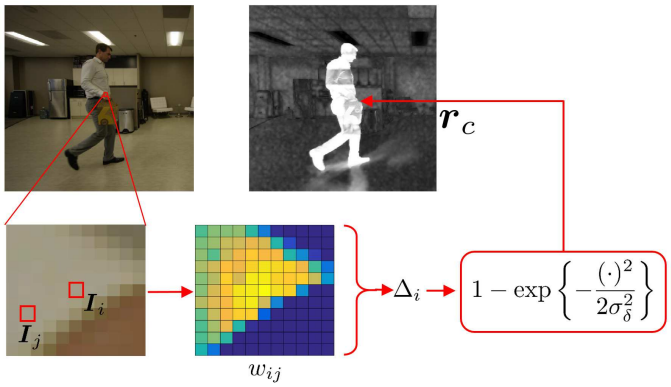


Fig. 5. Illustration of how to construct the estimate $\boldsymbol{r}_c$. We compute the distance between the foreground and the background. The distance has a bilateral weight to improve robustness. The actual $\boldsymbol{r}_0$ represents the probability of having a foreground pixel.

where

$$\widetilde{w}_{ij} = \exp\left\{-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2h_s^2}\right\} \exp\left\{-\frac{\|\boldsymbol{I}_i - \boldsymbol{I}_j\|^2}{2h_r^2}\right\}. \tag{29}$$

Here, $\boldsymbol{x}_i$ denotes the spatial coordinate of pixel i, $\boldsymbol{I}_i \in \mathbb{R}^3$ denotes the ith color pixel of the color image $\boldsymbol{I}$, and (h_s, h_r) are the parameters controlling the bilateral weight strength. The typical values of hs and hr are both 5.

We now need a mapping which maps the distance $\boldsymbol{\Delta} \overset{\text{def}}{=} [\Delta_1, \ldots, \Delta_n]^T$ to a vector of numbers $\boldsymbol{r}_c$ in $[0, 1]^n$ so that the term $\|\boldsymbol{\alpha} - \boldsymbol{r}_0\|^2$ makes sense. To this end, we choose a simple Gaussian function:

$$\boldsymbol{r}_c = 1 - \exp\left\{-\frac{\boldsymbol{\Delta}^2}{2\sigma_\delta^2}\right\}, \tag{30}$$

where σ_δ is a user tunable parameter. We tested other possible mappings such as the sigmoid function and the cumulative distribution function of a Gaussian. However, we do not see significant difference compared to (30). The typical value for σ_δ is 10.

Defining the Edge Term $\boldsymbol{r}_e$. The color term $\boldsymbol{r}_c$ is able to capture most of the difference between the image and the plate. However, it also generates false alarms if there is illumination change. For example, shadow due to the foreground object is often falsely labeled as foreground. See the shadow near the foot in Figure 5.

In order to reduce the false alarm due to minor illumination change, we first create a "super-pixel" mask by grouping similar colors. Our super-pixels are generated by applying a standard flood-fill algorithm [67] to the image $\boldsymbol{I}$. This gives us a partition of the image $\boldsymbol{I}$ as

$$\boldsymbol{I} \to \{\boldsymbol{I}^{S_1}, \boldsymbol{I}^{S_2}, \ldots, \boldsymbol{I}^{S_m}\}, \tag{31}$$

where $S_1, \ldots, S_m$ are the m super-pixel index sets. The plate image is partition using the same super-pixel indices, i.e., $\boldsymbol{P} \to \{\boldsymbol{P}^{S_1}, \boldsymbol{P}^{S_2}, \ldots, \boldsymbol{P}^{S_m}\}$.

While we are generating the super-pixels, we also compute the gradients of $\boldsymbol{I}$ and $\boldsymbol{P}$ for every pixel $i = 1, \ldots, n$. Specifically, we define $\nabla \boldsymbol{I}_i = [\nabla_x \boldsymbol{I}_i, \nabla_y \boldsymbol{I}_i]^T$ and $\nabla \boldsymbol{P}_i = [\nabla_x \boldsymbol{P}_i, \nabla_y \boldsymbol{P}_i]^T$, where $\nabla_x \boldsymbol{I}_i \in \mathbb{R}^3$ (and $\nabla_y \boldsymbol{I}_i \in \mathbb{R}^3$) are the

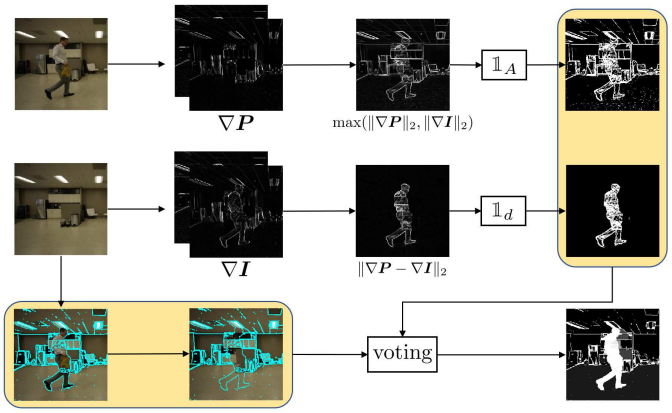
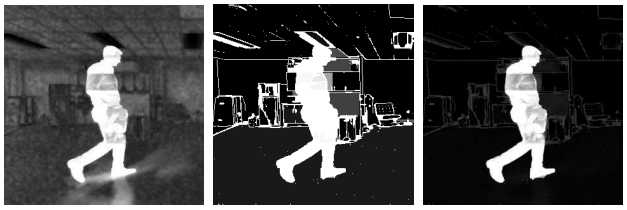


Fig. 6. Illustration of how to construct the estimate $\boldsymbol{r}_e$.



(a) $\boldsymbol{r}_c$ (b) $\boldsymbol{r}_e$ (c) $\boldsymbol{r}_0$

Fig. 7. Comparison between $\boldsymbol{r}_c$, $\boldsymbol{r}_e$, and $\boldsymbol{r}_0$.

two-tap horizontal (and vertical) finite difference at the i-th pixel. To measure how far $\boldsymbol{I}_i$ is from $\boldsymbol{P}_i$, we compute

$$\theta_i = \|\nabla \boldsymbol{I}_i - \nabla \boldsymbol{P}_i\|_2. \tag{32}$$

Thus, θ_i is small for background regions because $\boldsymbol{I}_i \approx \boldsymbol{P}_i$, but is large when there is a foreground pixel in $\boldsymbol{I}_i$. If we set a threshold operation after θ_i, i.e., set $\theta_i = 1$ if $\theta_i > \tau_\theta$ for some threshold τ_θ, then shadows can be removed as their gradients are weak.

Now that we have computed θ_i, we still need to map it back to a quantity similar to the alpha matte. To this end, we compute a normalization term

$$A_i = \max\left(\|\nabla \boldsymbol{I}_i\|_2, \|\nabla \boldsymbol{P}_i\|_2\right), \tag{33}$$

and normalize $\mathbb{1}\{\theta_i > \tau_\theta\}$ by

$$(r_e)_i \overset{\text{def}}{=} \frac{\sum_{j \in S_i} \mathbb{1}\{A_i > \tau_A\}\mathbb{1}\{\theta_i > \tau_\theta\}}{\sum_{j \in S_i} \mathbb{1}\{A_i > \tau_A\}}, \tag{34}$$

where $\mathbb{1}$ denotes the indicator function, and τ_A and τ_θ are thresholds. In essence, (34) says in the i-th super-pixel S_i, we count the number of edges $\mathbb{1}\{\theta_i > \tau_\theta\}$ that have strong difference between $\boldsymbol{I}_i$ and $\boldsymbol{P}_i$. However, we do not want to count every pixel but only pixels that already contains strong edges, either in $\boldsymbol{I}$ or $\boldsymbol{P}$. Thus, we take the weighted average using $\mathbb{1}\{A_i > \tau_A\}$ as the weight. This defines $\boldsymbol{r}_e$, as the weighted average $(r_e)_i$ is shared among all pixels in the super-pixel S_i. Figure 6 shows a pictorial illustration.

Why is $\boldsymbol{r}_e$ helpful? If we look at $\boldsymbol{r}_c$ and $\boldsymbol{r}_e$ in Figure 7, we see that the foreground pixels of $\boldsymbol{r}_c$ and $\boldsymbol{r}_e$ coincide but background pixels roughly cancel each other. The reason is that while $\boldsymbol{r}_c$ creates weak holes in the foreground, $\boldsymbol{r}_e$ fills the gap by ensuring the foreground is marked.

Regularization $\boldsymbol{\alpha}^T(1-\boldsymbol{\alpha})$. The last term $\boldsymbol{\alpha}^T(1-\boldsymbol{\alpha})$ in (25) is a regularization to force the solution to either 0 or 1. The effect of this term can be seen from the fact that $\boldsymbol{\alpha}^T(1-\boldsymbol{\alpha})$ is a symmetric concave quadratic function with a value zero for $\boldsymbol{\alpha} = \mathbf{1}$ or $\boldsymbol{\alpha} = \mathbf{0}$. Therefore, it introduces penalty for solutions that are away from 0 or 1. For $\gamma \leq 1$, one can show that the Hessian matrix of the function $f_2(\boldsymbol{\alpha}) = \|\boldsymbol{\alpha} - \boldsymbol{r}_0\|^2 + \gamma \boldsymbol{\alpha}^T(1-\boldsymbol{\alpha})$ is positive semidefinite. Thus, f_2 is strongly convex with parameter γ.

4.3 Agent 3: Total Variation Denoising

The third agent we use in this paper is the total variation denoising:

$$F_3(\boldsymbol{z}) = \underset{\boldsymbol{\alpha}}{\text{argmin}} \quad \|\boldsymbol{\alpha}\|_{\text{TV}} + \lambda_3 \|\boldsymbol{\alpha} - \boldsymbol{z}\|^2, \tag{35}$$

where λ_3 is a parameter. The norm $\|\cdot\|_{\text{TV}}$ is defined in space-time:

$$\|\boldsymbol{v}\|_{\text{TV}} \overset{\text{def}}{=} \sum_{i,j,t} \sqrt{\beta_x(\nabla_x \boldsymbol{v})^2 + \beta_y(\nabla_y \boldsymbol{v})^2 + \beta_t(\nabla_t \boldsymbol{v})^2}, \tag{36}$$

where $(\beta_x, \beta_y, \beta_t)$ controls the relative strength of the gradient in each direction. In this paper, for spatial total variation we set $(\beta_x, \beta_y, \beta_t) = (1, 1, 0)$, and for spatial-temporal total variation we set $(\beta_x, \beta_y, \beta_t) = (1, 1, 0.25)$.

A denoising agent is used in the MACE framework because we want to ensure smoothness of the resulting matte. The choice of the total variation denoising operation is a balance betweeen complexity and performance. Users can use stronger denoisers such as BM3D. However, these patch based image denoising algorithms rely on the patch matching procedure, and so they tend to under-smooth repeated patterns of false alarm / misses. Neural network denoisers are better candidates but they need to be trained with the specifically distorted alpha mattes. From our experience, we do not see any particular advantage of using CNN-based denoisers. Figure 8 shows some comparison.

(a) Input (b) TV [68] (c) BM3D [69] (d) IRCNN [70]

Fig. 8. Comparison of different denoisers used in MACE. Shown are the results when MACE converges. The shadow near the foot is a typical place of false alarm, and many denoisers cannot handle.

4.4 Parameters and Runtime

The typical values for parameters of the proposed method are presented in Table 4.4. λ_1 and λ_2 are rarely changed, while λ_3 determines the denoising strength of Agent 3. γ has a default value of 0.05. Inceasing γ causes more binary results with clearer boundaries. τ_A and τ_θ determine the edge term $\boldsymbol{r}_e$ in Agent 2 and are fixed. σ_δ determines the color term $\boldsymbol{r}_c$ in Agent 2. Large σ_δ produces less false negative but more false positive. Overall, the performance is reasonably stable to these parameters.

TABLE 2
Typical values for parameters

Parameter	λ_1	λ_2	λ_3	γ	τ_A	τ_θ	σ_δ
Value	0.01	2	4	0.05	0.01	0.02	10

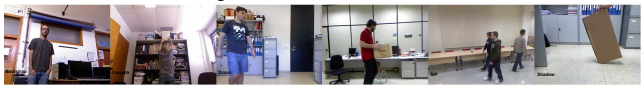
In terms of runtime, the most time-consuming part is Agent 1 because we need to solve a large-scale sparse least squares problem. Its runtime is determined by the number of foreground pixels. Table 3 shows the runtime of the sequences we tested. In generating these results, we used an un-optimized MATLAB code on a Intel i7-4770k. The typical runtime is about 1-3 minutes per frame. From our experience working with professional artists, even with professional film production software, e.g., NUKE, it takes 15 minutes to label a ground truth label using the plate and temporal cues. Therefore, the runtime benefit offered by our algorithm is substantial. The current runtime can be significantly improved by using multi-core CPU or GPU. Our latest implementation on GPU achieves 5 seconds per frame for images of size 1280×720.

5 Experimental Results

5.1 Dataset

To evaluate the proposed method, we create a Purdue dataset containing 8 video sequences using the HypeVR Inc. 360 degree camera. The original image resolution is 8192×4320 at a frame rate of 48fps, and these images are then downsampled and cropped to speed up the matting process. In addition to these videos, we also include 6 videos sequences from a public dataset [55], making a total of 14 video sequences. Snapshots of the sequences are shown in Figure 9. All video sequences are captured without camera motion. Plate images are available, either during the first or the last few frames of the video. To enable objective evaluation, for each video sequence we randomly select 10 frames and manually generate the ground truths. Thus totally there are 140 frames with ground truths.

(a) Snapshots of the Purdue Dataset



(b) Snapshots of a public dataset [55]

Fig. 9. Snapshots of the videos we use in the experiment. Top row: Building, Coach, Studio, Road, Tackle, Gravel, Office, Book. Bottom row:Bootstrap, Cespatx, Dcam, Gen, MP, Shadow.

The characteristics of the dataset is summarized in Table 3. The Purdue dataset has various resolution, and the Public dataset has one resolution 480×640. The foreground percentage for the Purdue dataset videos ranges from 1.03% to 55.10%, whereas that public dataset has similar foreground percentage around 10%. The runtime of the algorithm (per frame) is determined by the resolution and the foreground percentage. In terms of content, the Purdue dataset focuses on outdoor scenes whereas the public dataset are only indoor. The `shadow` column indicates the presence of shadow. `Lighting issues` include illumination change due to auto-exposure and auto-white-balance. The `background vibration` only applies to outdoor scenes where the background objects have minor movements, e.g., moving grass or tree branches. The `camouflage` column indicates the similarity in color between the foreground and background, which is a common problem for most sequences. The `green screen` column shows which of the sequences have green screens to mimic the common chroma-keying environment.

5.2 Competing methods

We categorize the competing methods into four different groups. The key ideas are summarized in Table 5.2.

- **Video Segmentation**: We consider four unsupervised video segmentation methods: Visual attention (AGS) [6], pyramid dilated bidirectional ConvLSTM (PDB) [45], motion adaptive object segmentation (MOA) [50], non-local consensus voting (NLVS) [41]. These methods are fully-automatic and do not require a plate image. All algorithms are downloaded from the author's websites and are run under default configurations.
- **Background Subtraction**: We consider two background subtraction algorithms Pixel-based adaptive segmenter (PBAS) [5], Visual background extractor (ViBe) [29]. Both algorithms are downloaded from the author's websites and are run under default configurations.
- **Alpha matting**: We consider one of the state-of-the-art alpha matting algorithm using CNN [7]. The trimaps are generated by applying frame difference between the plate and color images, followed by morphological and thresholding operations.
- **Others**: We consider the bilateral space video segmentation (BSVS) [19] which is a semi-supervised method. It requires the user to provide ground truth labels for key frames. We also modified the original Grabcut [23] to use the plate image instead of asking for user input.

5.3 Metrics

The following four metrics are used.

• **Intersection-ver-union (IoU)** measures the overlap between the estimate mask and the ground truth mask:

$$\text{IoU} = \frac{\sum_i \min(\widehat{x}_i, x_i)}{\sum_i \max(\widehat{x}_i, x_i)},$$

where $\widehat{x}_i$ is the i-pixel of the estimated alpha matte, and x_i is that of the ground truth. Higher IoU score is better.

• **Mean-absolute-error (MAE)** measures the average absolute difference between the ground truth and the estimate. Lower MAE is better.

• **Contour accuracy (F)** [72] measures the performance from a contour based perspective. Higher F score is better.

• **Structure measure (S)** [73] simultaneously evaluates region-aware and object-aware structural similarity between the result and the ground truth. Higher S score is better.

• **Temporal instability (T)** [72] that performs contour matching with polygon representations between two adjacent frames. Lower T score is better.

TABLE 3
Description of the video sequences used in our experiments.

		resolution	FGD %	time/Fr (sec)	indoor/ outdoor	shadow	lighting issues	Backgrd vibration	camouflage	green screen	ground truth
Purdue Dataset	Book	540x960	19.75%	231	outdoor			✓	✓		✓
	Building	632x1012	4.03%	170.8	outdoor	✓		✓	✓		✓
	Coach	790x1264	4.68%	396.1	outdoor			✓		✓	✓
	Studio	480x270	55.10%	58.3	indoor						✓
	Road	675x1175	1.03%	232.9	outdoor	✓		✓	✓		✓
	Tackle	501x1676	4.80%	210.1	outdoor	✓		✓		✓	✓
	Gravel	790x1536	2.53%	280.1	outdoor	✓		✓	✓		✓
	Office	623x1229	3.47%	185.3	indoor	✓			✓		✓
Public Dataset [55]	Bootstrap	480x640	13.28%	109.1	indoor	✓	✓		✓		✓
	Cespatx	480x640	10.31%	106.4	indoor	✓	✓		✓		✓
	DCam	480x640	12.23%	123.6	indoor	✓	✓		✓		✓
	Gen	480x640	10.23%	100.4	indoor	✓	✓		✓		✓
	Multipeople	480x640	9.04%	99.5	indoor	✓	✓		✓		✓
	Shadow	480x640	11.97%	115.2	indoor	✓	✓				✓

TABLE 4
Description of the competing methods.

	Methods		Supervised	Key idea
Unsupervised Video Segmentation	NLVS	[41]	no	non-local voting
	AGS	[6]	no	visual attention
	MOA	[50]	no	2-stream adaptation
	PDB	[45]	no	pyramid ConvLSTM
Alpha matting	Matting	[71]	trimap	Trimap generation + alpha matting
Background subtraction	ViBe	[29]	no	pixel model based
	PBAS	[5]	no	non-parametric
Other	BSVS	[19]	key frame	bilateral space
	Grabcut	[23]	plate	iterative graph cuts

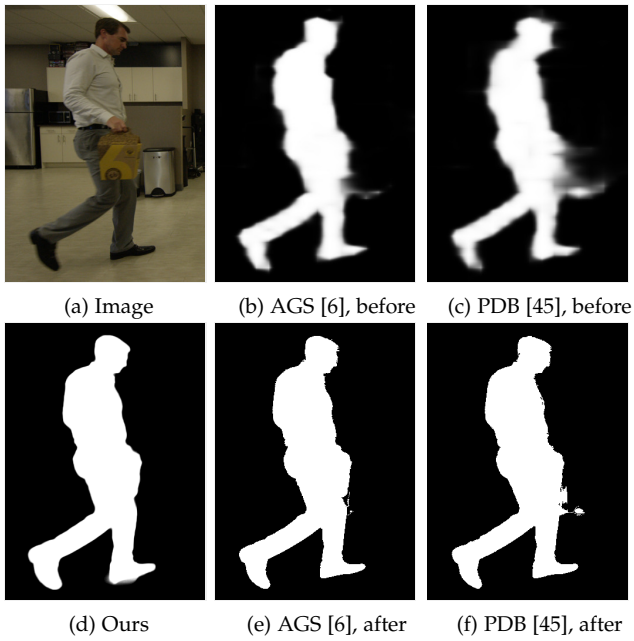
5.4 Results

• **Comparison with video segmentation methods**: The results are shown in Table 5.4, where we list the average IoU, MAE, F, S and T scores over the datasets. In this table, we notice that the deep-learning solutions AGS [6], MOA [50] and PDB [45] are significantly better than classical optical flow based NLVS [41] in all the metrics. However, since the deep-learning solutions are targeting for saliency detection, foreground but unsalient objects will be missed. AGS performs the best among the three with a F measure of 0.91, S measure of 0.94 and T measure of 0.19. PDB performs better than MOA in most metrics other than the T measure, with PDB scoring 0.2 while MOA scoreing 0.19.

We should also comment on the reliance on conditional random field of these deep learning solutions. In Figure 10 we show the raw outputs of AGS [6] and PDB [45]. While the salient object is correctly identified, the masks are coarse. Only after the conditional random field [51] the results become significantly better. In contrast, the raw output of our proposed algorithm is already high quality.

• **Comparison with trimap + alpha-matting methods**: In this experiment we compare with several state-of-the-art alpha matting algorithms. The visual comparison is shown Figure 4, and the performance of DCNN [7] is shown in Table 5.4. In order to make this method work, careful tuning during the trimap generation stage is required.

Figure 4 and Table 5.4 show that most alpha matting algorithms suffer from false alarms near the boundary, e.g., spectral matting [56], closed-form mating [12], learning-based matting [57] and comprehensive matting [59]. The more recent methods such as K-nearest neighbors matting [58] and DCNN [7] have equal amount of false alarm and miss. Yet, the overall performance is still worse than the proposed method and AGS. It is also worth noting that the matting approach achieves the second lowest T score (0.19), which is quite remarkable considering it is only a single-image method.



(a) Image (b) AGS [6], before (c) PDB [45], before

(d) Ours (e) AGS [6], after (f) PDB [45], after

Fig. 10. Dependency of conditional random field. (a) Input. (b) Raw output of the neural network part of AGS [6]. (c) Raw output of neural network part of PDB [45]. (d) Our result without post-processing. (e) Post-processing of AGS using conditional random field. (f) Post-processing of PDB using conditional random field. Notice the rough raw output of the deep neural network parts.

TABLE 5
Average results comparison with competing methods: AGS [6],PDB [45],MOA [50], NLVS [41], Trimap + DCNN [7], PBAS [5], ViBe [29], BSVS [19], Grabcut [23]. Higher intersection-over-union (IoU), higher Contour accuracy (F) [72], higher Structure measure (S) [73], lower MAE and lower Temporal instability (T) [72] indicate better performance.

		Unsupervised Video Segmentation				Matting	Bkgnd Subtract.		Others	
Metric	Our	AGS [6]	PDB [45]	MOA [50]	NLVS [41]	Tmap [7]	PBAS [5]	ViBe [29]	BSVS [19]	Gcut [23]
		CVPR '19	ECCV '18	ICRA '19	BMVC '14	ECCV '16	CVPRW '12	TIP '11	CVPR '16	ToG '04
IoU	**0.9321**	0.8781	0.8044	0.7391	0.5591	0.7866	0.5425	0.6351	0.8646	0.6574
MAE	**0.0058**	0.0113	0.0452	0.0323	0.0669	0.0216	0.0842	0.0556	0.0093	0.0392
F	**0.9443**	0.9112	0.8518	0.7875	0.6293	0.7679	0.6221	0.5462	0.8167	0.6116
S	**0.9672**	0.938	0.8867	0.8581	0.784	0.9113	0.7422	0.8221	0.9554	0.8235
T	**0.165**	0.1885	0.2045	0.1948	0.229	0.1852	0.328	0.2632	0.2015	0.232

• **Comparison with background subtraction methods**: Background subtraction methods PBAS [5] and ViBe [29] are not able to obtain a score higher than 0.65 for IoU. Their MAE values are also significantly larger than the proposed method. Their temporal consistency is lagging by larger than 0.25 for T measure. Qualitatively, we observe that background subtraction methods perform most badly for scenes where the foreground objects are mostly stable or only have rotational movements. This is a common drawback of background subtraction algorithms, since they learn the background model in a online fashion and will gradually include non-moving objects into the background model. Without advanced design to ensure spatial and temporal consistency, the results also show errors even when the foreground objects are moving.

• **Comparison with other methods**: Semi-supervised BSVS [19] requires ground truth key frames to learn a model. After the model is generated, the algorithm will overwrite the key frames with the estimates. When conducting this experiment, we ensure that the key frames used to generate the model are not used during testing. The result of this experiment shows that despite the key frames, BSVS [19] still performs worse than the proposed method. It is particularly weak when the background is complex where the key frames fail to form a reliable model.

The modified Grabcut [23] uses the plate image as a guide for the segmentation. However, because of the lack of additional prior models the algorithm does not perform well. This is particularly evident in images where colors are similar between foreground and background. Overall, Grabcut scores badly in most metrics, only slightly better than the background subtraction methods.

5.5 Ablation study

Since the proposed framework contains three different agents F_1, F_2 and F_3, we conduct an ablation study to verify the relative importance of the individual agents. To do so, we remove one of the three agents while keeping the other two fixed. The result is shown in Table 6. For T score, results for w/o F_1 and w/o F_3 are omitted, as their results have many small regions of false alarms rendering untrackable amount of points on the polygon contours used in calculating T measure.

The matting agent F_1 has the most impact on the performance, followed by background estimator and denoiser. The drop in performance is most significant for hard sequences such as `Book` as it contains moving background, and `Road` as it contains strong color similarity between foreground and background. On average, we observe signicant drop in IoU from 0.93 to 0.72 when the matting agent is absent. The F measure decreases from 0.94 to 0.76 as the boundaries are more erroneous without the matting agent. The structure measure also degrades from 0.97 to 0.85. The amount of error in the results also cause the T measure to become untrackable.

In this ablation study, we also observe spikes of error for some scenes when F_2 is absent. This is because, without the $\boldsymbol{\alpha}^T(1-\boldsymbol{\alpha})$ term in F_2, the result will look grayish instead of close-to-binary. This behavior leads to the error spikes. One thing worth noting is that the results obtained without F_2 do not drop significantly for S, F and T metric. This is due to the fact that IoU and MAE are pixel based metrics, whereas F, S and T are structural similarity. Therefore, even though the foreground becomes greyish without F_2, the structure of the labelled foreground is mostly intact.

For F_3, we observe that the total variation denoiser leads to the best performance for MACE. In a visual comparison shown in Figure 8, we observe that IRCNN [70] produces more detailed boundaries but fails to remove false alarms near the feet. BM3D [69] removes false alarms better but produces less detailed boundaries. TV on the other hand produces a more balanced result. As shown in Table 6, BM3D performs similarly as IRCNN scoring similar values for most metrics except that IrCNN scores 0.93 in F measure with BM3D only scoring 0.77 meaning more accurate contours. In general, even with different denoisers, the proposed method still outperforms most competing methods.

6 Limitations and Discussion

While the proposed method demonstrates superior performance than the state-of-the-art methods, it also has several limitations.

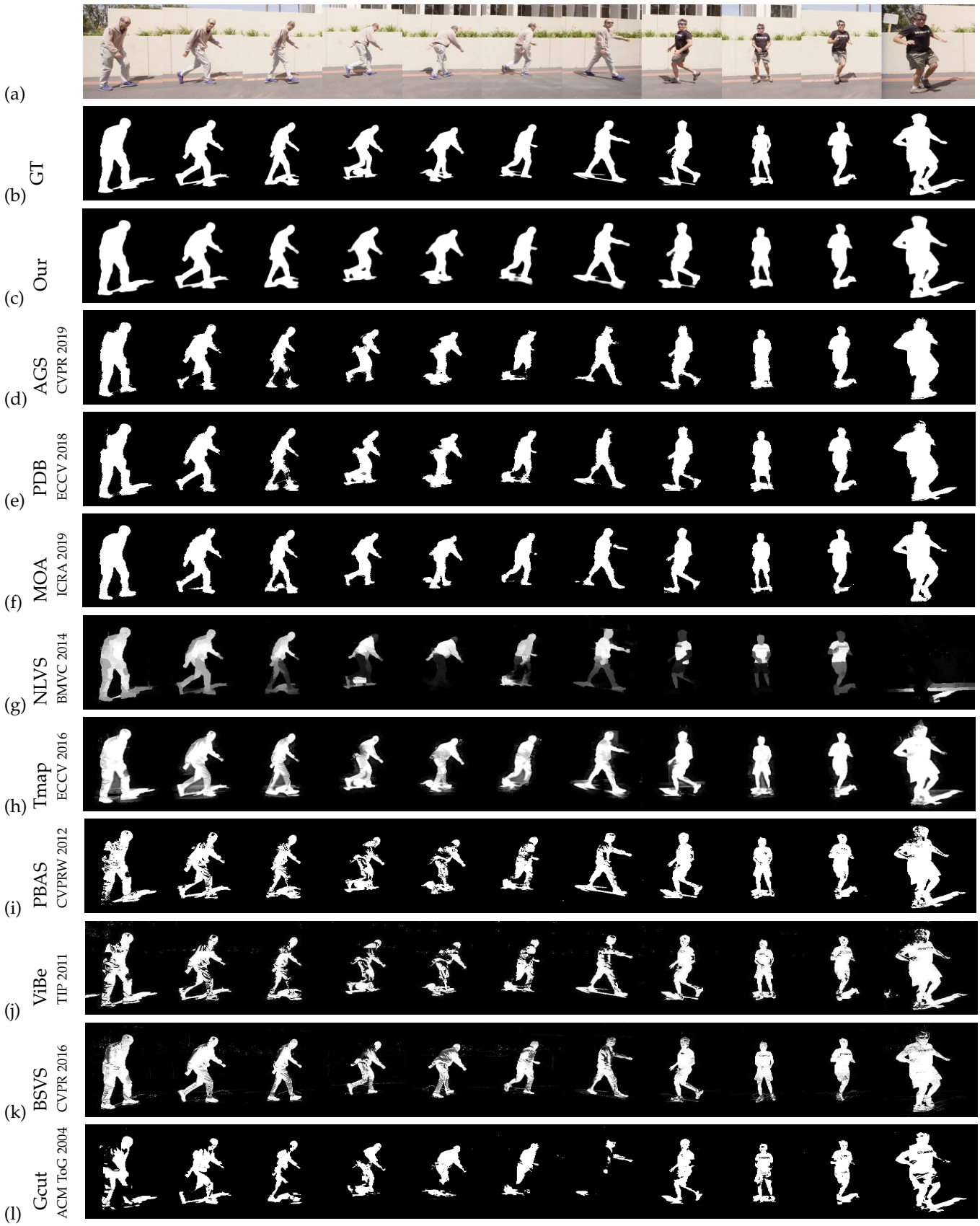


Fig. 11. Office sequence results. (a) Input. (b) Ground truth. (c) Ours. (d) AGS [6]. (e) PDB [45]. (f)MOA [50] (g)NLVS [41]. (h) Trimap + DCNN [7]. (i) PBAS [5]. (j) ViBe [29]. (k)BSVS [19]. (l)Gcut [23].

TABLE 6
Ablation study of the algorithm. We show the performance by eliminating one of the agents, and replacing the denoising agent with other denoisers. Higher intersection-over-union (IoU), higher Contour accuracy (F) [72], higher Structure measure (S) [73], lower MAE and lower Temporal instability (T) [72] indicate better performance.

Metric	Our	w/o F_1	w/o F_2	w/oF_3	BM3D	IrCNN
IoU	**0.9321**	0.7161	0.7529	0.7775	0.8533	0.8585
MAE	**0.0058**	0.0655	0.0247	0.0368	0.0128	0.0121
F	**0.9443**	0.7560	0.9166	0.6510	0.7718	0.8506
S	**0.9672**	0.8496	0.9436	0.8891	0.9334	0.9320
T	**0.165**	too large	0.1709	too large	0.1911	0.1817

- **Quality of Plate Image**. The plate assumption may not hold when the background is moving substantially. When this happens, a more complex background model that includes dynamic information is needed. However, if the background is non-stationary, additional designs are needed to handle the local error and temporal consistency.
- **Strong Shadows**. Strong shadows are sometimes treated as foreground, as shown in Figure 12. This is caused by the lack of shadow modeling in the problem formulation. The edge based initial estimate $\boldsymbol{r}_e$ can resolve the shadow issue to some extent, but not when the shadow is very strong. We tested a few off-the-shelf shadow removal algorithms [74]–[76], but generally they do not help because the shadow in our dataset can cast on the foreground object which should not be removed.

Fig. 12. Strong shadows. When shadows are strong, they are easily misclassified as foreground.

An open question here is whether our problem can be solved using deep neural networks since we have the plate. While this is certainly a feasible task because we can use the plate to replace the guided inputs (e.g., optical flow in [50] or visual attention in [6]), an appropriate training dataset is needed. In contrast, the proposed method has the advantage that it is training-free. Therefore, it is less susceptible to issues such as overfit. We should also comment that the MACE framework allows us to use deep neural network solutions. For example, one can replace F_1 with a deep neural network, and F_2 with another deep neural network. MACE is guaranteed to find a fixed point of these two agents if they do not agree.

7 Conclusion

This paper presents a new foreground extraction algorithm based on the multi-agent consensus equilibrium (MACE) framework. MACE is an information fusion framework which integrates multiple weak experts to produce a strong estimator. Equipped with three customized agents: a dual-layer closed form matting agent, a background estimation agent and a total variation denoising agent, MACE offers substantially better foreground masks than state-of-the-art algorithms. MACE is a fully automatic algorithm, meaning that human interventions are not required. This provides significant advantage over semi-supervised methods which require trimaps or scribbles. In the current form, MACE is able to handle minor variations in the background plate image, illumination changes and weak shadows. Extreme cases can still cause MACE to fail, e.g., background movement or strong shadows. However, these could potentially be overcome by improving the background and shadow models.

8 Appendix

8.1 Proof of Theorem 2

Proof. We start by writing (18) in the matrix form

$$\widetilde{J}(\boldsymbol{\alpha}^{\boldsymbol{I}}, \boldsymbol{\alpha}^{\boldsymbol{P}}, \boldsymbol{a}, \boldsymbol{b}) = \sum_{k \in \boldsymbol{I}} \left\| \begin{bmatrix} \boldsymbol{H}_k & \mathbf{1} \\ \sqrt{\eta}\boldsymbol{G}_k & \sqrt{\eta}\mathbf{1} \\ \sqrt{\epsilon}\boldsymbol{I}_{3\times 3} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{a}_k \\ b_k \end{bmatrix} - \begin{bmatrix} \boldsymbol{\alpha}_k^{\boldsymbol{I}} \\ \boldsymbol{\alpha}_k^{\boldsymbol{P}} \\ \mathbf{0} \end{bmatrix} \right\|^2$$

where

$$\boldsymbol{H}_k = \begin{bmatrix} \vdots & \vdots & \vdots \\ I_i^r & I_i^g & I_i^b \\ \vdots & \vdots & \vdots \end{bmatrix}, \quad \boldsymbol{G}_k = \begin{bmatrix} \vdots & \vdots & \vdots \\ P_i^r & P_i^g & P_i^b \\ \vdots & \vdots & \vdots \end{bmatrix},$$

$$\boldsymbol{a}_k = \begin{bmatrix} a_k^r \\ a_k^g \\ a_k^b \end{bmatrix}, \quad \boldsymbol{\alpha}_k^{\boldsymbol{I}} = \begin{bmatrix} \vdots \\ \alpha_i^{\boldsymbol{I}} \\ \vdots \end{bmatrix}, \quad \boldsymbol{\alpha}_k^{\boldsymbol{P}} = \begin{bmatrix} \vdots \\ \alpha_i^{\boldsymbol{P}} \\ \vdots \end{bmatrix},$$

and i denotes the index of the i-th pixel in the neighborhood w_k. The difference with the classic closed-form matting [12] is the new terms $\boldsymbol{G}_k$, $\mathbf{1}$ and $\boldsymbol{\alpha}_k^{\boldsymbol{P}}$ (i.e., the second row of the quadratic function above.)

Denote

$$\boldsymbol{B}_k \stackrel{\text{def}}{=} \begin{bmatrix} \boldsymbol{H}_k & \mathbf{1} \\ \sqrt{\eta}\boldsymbol{G}_k & \sqrt{\eta}\mathbf{1} \\ \sqrt{\epsilon}\boldsymbol{I}_{3\times 3} & \mathbf{0} \end{bmatrix}, \tag{37}$$

and use the fact that $\boldsymbol{\alpha}^{\boldsymbol{P}} = \mathbf{0}$, we can find out the solution of the least-squares optimization:

$$\begin{bmatrix} \boldsymbol{a}_k \\ b_k \end{bmatrix} = (\boldsymbol{B}_k^T \boldsymbol{B}_k)^{-1} \boldsymbol{B}_k^T \begin{bmatrix} \boldsymbol{\alpha}_k^{\boldsymbol{I}} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{38}$$

We now need to simplify the term $\boldsymbol{B}_k^T\boldsymbol{B}_k$. First, observe that

$$\begin{aligned}\boldsymbol{B}_k^T\boldsymbol{B}_k &= \begin{bmatrix}\boldsymbol{H}_k^T\boldsymbol{H}_k + \eta\boldsymbol{G}_k^T\boldsymbol{G}_k + \epsilon\boldsymbol{I}_{3\times3} & \boldsymbol{H}_k^T\mathbf{1} + \eta\boldsymbol{G}_k^T\mathbf{1}\\ (\boldsymbol{H}_k\mathbf{1} + \eta\boldsymbol{G}_k^T\mathbf{1})^T & n(1+\eta)\end{bmatrix}\\ &= \begin{bmatrix}\boldsymbol{\Sigma}_k & \boldsymbol{\mu}_k\\ \boldsymbol{\mu}_k^T & c\end{bmatrix}\end{aligned}$$

where we define the terms $\boldsymbol{\Sigma}_k \stackrel{\text{def}}{=} \boldsymbol{H}_k^T\boldsymbol{H}_k + \eta\boldsymbol{G}_k^T\boldsymbol{G}_k + \epsilon\boldsymbol{I}$, $\boldsymbol{\mu}_k \stackrel{\text{def}}{=} \boldsymbol{H}_k^T\mathbf{1} + \eta\boldsymbol{G}_k^T\mathbf{1}$ and $c \stackrel{\text{def}}{=} n(1+\eta)$. Then, by applying the block inverse identity, we have

$$(\boldsymbol{B}_k^T\boldsymbol{B}_k)^{-1} = \begin{bmatrix}\boldsymbol{T}_k^{-1} & -\boldsymbol{T}_k^{-1}\widehat{\boldsymbol{\mu}}_k\\ -(\boldsymbol{T}_k^{-1}\widehat{\boldsymbol{\mu}}_k)^T & \frac{1}{c} + \widehat{\boldsymbol{\mu}}_k\boldsymbol{T}_k^T\widehat{\boldsymbol{\mu}}_k\end{bmatrix} \tag{39}$$

where we further define $\boldsymbol{T}_k = \boldsymbol{\Sigma}_k - \frac{\boldsymbol{\mu}_k\boldsymbol{\mu}_k^T}{c}$ and $\widehat{\boldsymbol{\mu}}_k = \frac{\boldsymbol{\mu}_k}{c}$.

Substituting (38) back to $\widetilde{J}$, and using (39), we have

$$\begin{aligned}\widetilde{J}(\boldsymbol{\alpha}^{\boldsymbol{I}}) &= \sum_k \left|\left|(\boldsymbol{I}_{3\times3} - \boldsymbol{B}_k(\boldsymbol{B}_k^T\boldsymbol{B}_k)^{-1}\boldsymbol{B}_k^T)\begin{bmatrix}\boldsymbol{\alpha}_k^{\boldsymbol{I}}\\ \mathbf{0}\\ \mathbf{0}\end{bmatrix}\right|\right|^2\\ &= (\boldsymbol{\alpha}_k^{\boldsymbol{I}})^T\boldsymbol{L}_k\boldsymbol{\alpha}_k^{\boldsymbol{I}},\end{aligned}$$

where

$$\begin{aligned}\boldsymbol{L}_k = \boldsymbol{I}_{3\times3} - \Big(&\boldsymbol{H}_k\boldsymbol{T}_K^{-1}\boldsymbol{H}_k^T - \boldsymbol{H}_k\boldsymbol{T}_k^{-1}\widehat{\boldsymbol{\mu}}_k\mathbf{1}^T\\ &- \mathbf{1}^T(\boldsymbol{T}_k^{-1}\widehat{\boldsymbol{\mu}}_k)^T\boldsymbol{H}_k + \frac{1}{c}\mathbf{1}^T\widehat{\boldsymbol{\mu}}_k\boldsymbol{T}_k^{-1}\widehat{\boldsymbol{\mu}}_k\mathbf{1}\Big)\end{aligned} \tag{40}$$

The (i,j)-th element of $\boldsymbol{L}_k$ is therefore

$$\begin{aligned}\boldsymbol{L}_k(i,j) =&\delta_{ij} - (\boldsymbol{I}_{ki}^T\boldsymbol{T}_k^{-1}\boldsymbol{I}_{kj} - \boldsymbol{I}_{ki}^T\boldsymbol{T}_k^{-1}\widehat{\boldsymbol{\mu}}_k\\ &- \widehat{\boldsymbol{\mu}}_k^T\boldsymbol{T}_k^{-1}\boldsymbol{I}_{kj} + \frac{1}{c} + \widehat{\boldsymbol{\mu}}_k^T\boldsymbol{T}_k^{-1}\widehat{\boldsymbol{\mu}}^k)\\ =&\delta_{ij} - (\frac{1}{c} + (\boldsymbol{I}_{ki} - \widehat{\boldsymbol{\mu}}_k)^T\boldsymbol{T}_k^{-1}(\boldsymbol{I}_{kj} - \widehat{\boldsymbol{\mu}}_k))\end{aligned} \tag{41}$$

Adding terms in each w_k, we finally obtain

$$\widetilde{L}_{i,j} = \sum_{k|(i,j)\in w_k}\Big\{\delta_{ij} - (\frac{1}{c} + (\boldsymbol{I}_{ki} - \widehat{\boldsymbol{\mu}}_k)^T\boldsymbol{T}_k^{-1}(\boldsymbol{I}_{kj} - \widehat{\boldsymbol{\mu}}_k))\Big\}.$$

□

8.2 Proof: $\widetilde{\boldsymbol{L}}$ is positive definite

Proof. Recall the definition of $\widetilde{J}(\boldsymbol{\alpha}^I, \boldsymbol{\alpha}^P, \boldsymbol{a}, \boldsymbol{b})$:

$$\begin{aligned}\widetilde{J}(\boldsymbol{\alpha}^I, \boldsymbol{\alpha}^P, \boldsymbol{a}, \boldsymbol{b}) = \sum_{j\in I}\Bigg\{&\sum_{i\in w_j}\left(\alpha_i^I - \sum_c a_j^c I_i^c - b_j\right)^2\\ &+\eta\sum_{i\in w_j}\left(\alpha_i^P - \sum_c a_j^c P_i^c - b_j\right)^2 + \epsilon\sum_c (a_j^c)^2\Bigg\}\end{aligned}$$

Based on Theorem 2 we have,

$$\widetilde{J}(\boldsymbol{\alpha}) \stackrel{\text{def}}{=} \min_{\boldsymbol{a},\boldsymbol{b}}\ \widetilde{J}(\boldsymbol{\alpha}, \mathbf{0}, \boldsymbol{a}, \boldsymbol{b}) = \boldsymbol{\alpha}^T\widetilde{\boldsymbol{L}}\boldsymbol{\alpha}. \tag{42}$$

We consider two cases: (i) $a_j^c = 0\ \forall j$ and $\forall c$, (ii) there exists some j and c such that $a_j^c \neq 0$. For the second case, $\widetilde{J}$ is larger than 0. For the first case, $\widetilde{J}$ can be reduced into

$$\widetilde{J}(\boldsymbol{\alpha}, 0, \boldsymbol{a}, \boldsymbol{b}) = \sum_{j\in I}\left\{\sum_{i\in w_j}\left((\alpha_i - b_j)^2 + \eta\,(-b_j)^2\right)\right\} \tag{43}$$

For any vector $\alpha \neq 0$, there exists at least one $\alpha_i \neq 0$. Then by completing squares we can show that

$$\begin{aligned}&(\alpha_i - b_j)^2 + \eta b_j^2\\ &= \alpha_i^2 - 2\alpha_i b_j + (1+\eta)b_j^2\\ &= \left(\sqrt{\frac{1}{1+\eta}}\alpha_i - \sqrt{1+\eta}b_j\right)^2 + \frac{\eta}{1+\eta}\alpha_i^2 > 0\end{aligned}$$

Therefore, $\widetilde{J}(\boldsymbol{\alpha}, 0, \boldsymbol{a}, \boldsymbol{b}) > 0$ for any non-zero vector α. As a result, $\widetilde{J}(\boldsymbol{\alpha}, 0, \boldsymbol{a}, \boldsymbol{b}) = \boldsymbol{\alpha}^T\widetilde{\boldsymbol{L}}\boldsymbol{\alpha} > 0$ for both cases, and $\widetilde{\boldsymbol{L}}$ is positive definite. □

8.3 Proof of Proposition 1

Proof. Let $\underline{\boldsymbol{x}} \in \mathbb{R}^{nN}$ and $\underline{\boldsymbol{y}} \in \mathbb{R}^{nN}$ be two super-vectors.

(i). If the F_i's are non-expansive, then

$$\begin{aligned}&\|\mathcal{F}(\underline{\boldsymbol{x}}) - \mathcal{F}(\underline{\boldsymbol{y}})\|^2 + \|\underline{\boldsymbol{x}} - \underline{\boldsymbol{y}} - (\mathcal{F}(\underline{\boldsymbol{x}}) - \mathcal{F}(\underline{\boldsymbol{y}}))\|^2\\ &= \sum_{i=1}^N \left(\|F_i(\boldsymbol{x}_i) - F_i(\boldsymbol{x}_i)\|^2 + \|\boldsymbol{x}_i - \boldsymbol{y}_i - (F_i(\boldsymbol{x}_i) - F_i(\boldsymbol{y}_i))\|^2\right)\\ &\stackrel{(c)}{\leq} \sum_{i=1}^N \|\boldsymbol{x}_i - \boldsymbol{y}_i\|^2 = \|\underline{\boldsymbol{x}} - \underline{\boldsymbol{y}}\|^2\end{aligned}$$

where (c) holds because each F_i is firmly non-expansive. As a result, $\mathcal{F}$ is also firmly non-expansive.

(ii). To prove that $\mathcal{G}$ is firmly non-expansive, we recall from Theorem 1 that $2\mathcal{G} - \mathcal{I}$ is self-inverse. Since $\mathcal{G}$ is linear, it has a matrix representation. Thus, $\|(2\mathcal{G} - \mathcal{I})\boldsymbol{x}\|^2 = \boldsymbol{x}^T(2\mathcal{G} - \mathcal{I})^T(2\mathcal{G} - \mathcal{I})\boldsymbol{x}$. Because $\mathcal{G}$ is an averaging operator, it has to be symmetric, and hence $\mathcal{G}^T = \mathcal{G}$. As a result, we have $\|(2\mathcal{G} - \mathcal{I})\boldsymbol{x}\|^2 = \|\boldsymbol{x}\|^2$ for any $\boldsymbol{x}$, which implies non-expansiveness.

(iii). If $\mathcal{F}$ and $\mathcal{G}$ are both firmly non-expansive, we have

$$\begin{aligned}&\|(2\mathcal{G} - \mathcal{I})[(2\mathcal{F} - \mathcal{I})(\underline{\boldsymbol{x}})] - (2\mathcal{G} - \mathcal{I})[(2\mathcal{F} - \mathcal{I})(\underline{\boldsymbol{y}})]\|^2\\ &\stackrel{(a)}{\leq} \|(2\mathcal{F} - \mathcal{I})(\underline{\boldsymbol{x}}) - (2\mathcal{F} - \mathcal{I})(\underline{\boldsymbol{y}})\|^2 \stackrel{(b)}{\leq} \|\underline{\boldsymbol{x}} - \underline{\boldsymbol{y}}\|^2\end{aligned}$$

where (a) is true due to the firmly non-expansiveness of $\mathcal{G}$ and (b) is true due to the non-expansiveness of $\mathcal{G}$. Thus, $\mathcal{T} \stackrel{\text{def}}{=} (2\mathcal{G} - \mathcal{I})(2\mathcal{F} - \mathcal{I})$ is non-expansive. This result also implies convergence of the MACE algorithm, due to [60]. □

REFERENCES

[1] B. Garcia-Garcia and A. Silva T. Bouwmans, "Background subtraction in real applications: Challenges, current models and future directions," *Computer Science Review*, vol. 35, pp. 1–42, Feb. 2020.

[2] T. Bouwmans, F. Porikli, B. Hoferlin, and A. Vacavant, *Background Modeling and Foreground Detection for Video Surveillance*, Chapman and Hall, CRC Press, 2014.

[3] Y. Wang, Z. Luo, and P. Jodoin, "Interactive deep learning method for segmenting moving objects," *Pattern Recognition Letters*, vol. 96, pp. 66–75, Sep. 2017.

[4] S. Shimoda, M. Hayashi, and Y. Kanatsugu, "New chroma-key imagining technique with hi-vision background," *IEEE Trans. on Broadcasting*, vol. 35, no. 4, pp. 357–361, Dec. 1989.

[5] M. Hofmann, P. Tiefenbacher, and G. Rigoll, "Background segmentation with feedback: The pixel-based adaptive segmenter," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2012, pp. 38–43.

[6] W. Wang, H. Song, S. Zhao, J. Shen, S. Zhao, S. Hoi, and H. Ling, "Learning unsupervised video object segmentation through visual attention," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019, pp. 3064–3074.

[7] D. Cho, Y. Tai, and I. Kweon, "Natural image matting using deep convolutioonal neural networks," in *European Conference on Computer Vision (ECCV)*, Oct. 2016, pp. 626–643.

[8] T. Bouwmans, "Recent advanced statistical background modeling for foreground detection: A systematic survey," *Recent Patents on Computer Science*, vol. 4, no. 3, pp. 147–176, Sep. 2011.

[9] S. Javed, P. Narayanamurthy, T. Bouwmans, and N. Vaswani, "Robust PCA and robust subspace tracking: A comparative evaluation," in *Proc. IEEE Statistical Signal Processing Workshop*, 2018, pp. 836–840.

[10] T. Bouwmans, M. Sultana S. Javed, and S. K. Jung, "Deep neural network concepts in background subtraction: A systematic review and a comparative evaluation," *Neural Networks*, vol. 117, pp. 8–66, Sep. 2019.

[11] J. Sun, J. Jia, C. K. Tang, and H. Shum, "Poisson matting," *ACM Trans. on Graphics (ToG)*, vol. 23, no. 3, pp. 315–321, Aug. 2004.

[12] A. Levin, D. Lischinski, and Y. Weiss, "A closed-form solution to natural image matting," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 228–242, Feb. 2008.

[13] E. Gastal and M. Oliveira, "Shared sampling for real-time alpha matting," *Euro Graphics*, vol. 29, no. 2, pp. 575–584, 2010.

[14] Y. Y. Chuang, B. Curless, D. H. Salesin, and R. Szeliski, "A Bayesian approach to digital matting," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Dec 2001, vol. 2, pp. 11–18.

[15] J. Wang and M. F. Cohen, "Optimized color sampling for robust matting," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2007, pp. 1–8.

[16] N. Xu, B. Price, and T. Huang, "Deep image matting," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, pp. 311–320.

[17] J. Tang, Y. Aksoy, C. Oztireli, M Gross, and T. Aydin, "Learning-based sampling for natural image matting," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3055–3065.

[18] W. Xi, J. Chen, L. Qian, and J. Allebach, "High-accuracy automatic person segmentation with novel spatial saliency map," in *Proc. IEEE International Conference on Image Processing*, Sept. 2019.

[19] N. Marki, F. Perazzi, O. Wang, and A. Sorkine-Hornung, "Bilateral space video segmentation," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 743–751.

[20] S. A. Ramakanth and R. V Babu, "Seamseg: Video segmentation using patch seams," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2014, vol. 2, pp. 376–383.

[21] S. D. Jain and K. Grauman, "Supervoxel-consistent foreground propagation in video," in *Proc. European Conference on Computer Vision (ECCV)*. Sep. 2014, pp. 656–671, Springer.

[22] C. Hsieh and M. Lee, "Automatic trimap generation for digital image matting," in *Proc. IEEE Signal and Information Processing Association Annual Summit and Conference*, Oct. 2013, pp. 1–5.

[23] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interatcive foreground extraction using iterated graph cuts," *ACM Trans. on Graphics (ToG)*, vol. 23, no. 3, pp. 309–314, Aug. 2004.

[24] J. Cho, T. Yamasaki, K. Aizawa, and K. H. Lee, "Depth video camera based temporal alpha matting for natural 3d scene generation," in *3DTV Conference: The True Vision-Capture, Transmission and Display of 3D Video*, May 2011, pp. 1–4.

[25] O. Wang, J. Finger, Q. Yang, J. Davis, and R. Yang, "Automatic natural video matting with depth," in *15th Pacific Conference On Computer Graphics and Applications*, Oct. 2007, pp. 469–472.

[26] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction," in *Proc. IEEE International Conference on Pattern Recognition*, Aug. 2004, vol. 2, pp. 28–31.

[27] J. W. Davis and V. Sharma, "Fushion-based background-subtraction using contour saliency," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2005, pp. 11–19.

[28] V. Mahadevan and N. Vasconcelos, "Background subtraction in highly dynamic scenes," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2008, pp. 1–6.

[29] O. Barnich and V. D. Marc, "ViBe: A universal background subtraction algorithm for video sequences," *IEEE Trans. on Image Processing*, vol. 20, no. 6, pp. 1709–1724, Jun. 2011.

[30] T. Bouwmans, L. Maddalena, and A. Petrosino, "Scene background initialization: a taxonomy," *Pattern Recognition Letters*, vol. 96, pp. 3–11, Sep. 2017.

[31] A. Colombari, A. Fusiello, and V. Murino, "Background initialization in cluttered sequences," in *Workshop on Perceptual Organization in Computer Vision*, 2006, pp. 197–197.

[32] S. Javed, A. Mahmood, T. Bouwmans, and S. K. Jung, "Spatiotemporal low-rank modeling for complex scene background initialization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 6, pp. 1315–1329, Jun. 2018.

[33] S. Javed, T. Bouwmans, and S. K. Jung, "SBMI-LTD: Stationary background model initialization based on low-rank tensor decomposition," in *Proceedings of the Symposium on Applied Computing*, Apr. 2017, p. 195200.

[34] M. Sultana, A. Mahmood, S. Javed, and S. K. Jung, "Unsupervised deep context prediction for background estimation and foreground segmentation," *Machine Vision and Applications*, vol. 30, pp. 375395, Apr. 2019.

[35] B. Laugraud, S. Pierard, and M. V. Droogenbroeck, "LaBGen-P-Semantic: A first step for leveraging semantic segmentation in background generation," *MDPI Journal of Imaging*, vol. 4, no. 7, pp. 1–22, 2018.

[36] S. Pierard B. Laugraud and M. V. Droogenbroeck, "LaBGen: A method based on motion detection for generating the background of a scene," *Pattern Recognition Letters*, vol. 96, pp. 12–21, Sep. 2017.

[37] Y. Lee and K. Grauman, "Key-segments for video object segmentation," in *Proc. IEEE International Conference on Computer Vision (ICCV)*, Nov. 2011, pp. 1995–2002.

[38] T. Ma and L. Latecki, "Maximum weight cliques with mutex constraints for video object segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2012, pp. 670–677.

[39] F. Perazzi, P. Krahenbuhl, Y. Pritch, and A. Mornung, "Saliency filters: Contrast based filtering for salient region detection," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2012, pp. 733–740.

[40] A. Papazouglou and V. Ferrari, "Fast object segmentation in unconstrained video," in *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2013, pp. 1777–1784.

[41] A. Faktor and M. Irani, "Video segmentation by non-local consensus voting," in *Proc. British Machine Vision Association (BMVC)*, Jun. 2014, vol. 2, p. 8.

[42] W. Wang, J. Shen, X. Li, and F. Porikli, "Robust video object cosegmentation," *IEEE Trans. on Image Processing*, vol. 24, no. 10, pp. 3137–3148, June 2015.

[43] W. Jang, C. Lee, and C. Kim, "Primary object segmentation in videos via alternate convex optimization of foreground and background distributions," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 696–704.

[44] Q. Hou, M. Cheng, X. Hu, A. Borji, Z. Tu, and PH. Torr, "Deeply supervised salient object detection with short connections," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3203–3212.

[45] H. Song, W. Wang, S. Zhao, J. Shen, and K. Lam, "Pyramid dilated deeper convlstm for video salient object detection," in *InProceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[46] W. Wang, J. Shen, F. Porikli, and R. Yang, "Semi-supervised video object segmentation with super-trajectories," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 41, no. 4, pp. 985–998, Mar. 2018.

[47] S. Dong, Z. Gao, S. Sun, X. Wang, M. Li, H. Zhang, G. Yang, H. Liu, and S. Li, "Holistic and deep feature pyramids for saliency detection," in *Proc. British Machine Vision Conference (BMVC)*, 2018, pp. 1–13.

[48] S. Dong, Z. Gao, S. Pirbhulal, G. Bian, H. Zhang, W. Wu, , and S. Li, "Iot-based 3d convolution for video salient object detection," *Neural Computing and Applications*, vol. 32, pp. 735–746, 2020.

[49] W. Wang, J. Shen, J. Xie, M. Cheng, H. Ling, and A. Borji, "Revisiting video saliency prediction in the deep leanring era," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Apr. 2019, available on arXiv: https://arxiv.org/pdf/1904.09146.pdf.

[50] M. Dehghan, Z. Zhang, M. Siam, J. Jin, L. Petrich, and M. Jagersand, "Online tool and task learning via human robot interaction," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, Available online https://arxiv.org/pdf/1809.08722.pdf.

[51] P. Krahenbuhl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," in *Advances in Neural Information Processing Systems*, 2011, pp. 109–117.

[52] H. Wang and D. Suter, "Background subtraction based on a robust consensus method," in *International Conference on Pattern Recognition*, 2006, pp. 223–226.

[53] G. Han, X. Cai, and J. Wang, "Object detection based on combination of visible and thermal videos using a joint sample consensus background model," *Journal of Software*, vol. 8, no. 4, pp. 987–994, Apr. 2013.

[54] P. St-Charles, G. Bilodeau, and R. Bergevin, "Universal background subtraction using word consensus models," *IEEE Transactions on Image Processing*, vol. 25, no. 10, pp. 4768–4781, Oct. 2016.

[55] M. Camplani, L. Maddalena, G. M. Alcover, A. Petrosino, and L. Salgado, "A benchmarking framework for background subtraction in RGBD videos," in *New Trends in Image Analysis and Processing-ICIAP 2017 Workshops*. Sep. 2017, pp. 219–229, Springer.

[56] A. Levin, A. Rav-Acha, and D. Lischinski, "Spectral matting," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 30, no. 10, pp. 1699–1712, Oct. 2008.

[57] Y. Zheng and C. Kambhamettu, "Learning based digital matting," in *Proc. IEEE International Conference on Computer Vision (ICCV)*, Sep. 2009, pp. 889–896.

[58] Q. Chen, D. Li, and C. Tang, "KNN matting," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 35, no. 9, pp. 2175–2188, Sep. 2013.

[59] E. Shahrian, D. Rajan, B. Price, and S. Cohen, "Improving image matting using comprehensive sampling sets," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2013, pp. 636–643.

[60] G. Buzzard, S. H. Chan, S. Sreehari, and C. A. Bouman, "Plug-and-play unplugged: optimization free reconstruction ufing consensus equilibrium," *SIAM Imaging Science*, vol. 11, no. 3, pp. 2001–2020, Sept. 2018.

[61] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine learning*, vol. 3, no. 1, pp. 1–22, Jul. 2011.

[62] S. Sreehari, S. V. Venkatakrishnan, B. Wohlberg, G. T. Buzzard, L. F. Drummy, J. P. Simmons, and C. A. Bouman, "Plug-and-play priors for bright field electron tomography and sparse interpolation," *IEEE Trans. on Computational Imaging*, vol. 2, no. 4, pp. 408–423, Dec. 2016.

[63] S. H. Chan, X. Wang, and O. A. Elgendy, "Plug-and-play ADMM for image restoration," *IEEE Trans. on Computational Imaging*, vol. 3, no. 1, pp. 84–98, Mar. 2017.

[64] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 127–239, Jan. 2014.

[65] S. Venkatakrishnan, C. Bouman, and B. Wohlberg, "Plug-and-play priors for model based reconstruction," in *Proc. IEEE Global Conference on Signal and Information Processing*, Dec. 2013, pp. 945–948.

[66] X. Wang and S. H. Chan, "Parameter-free-plug-and-play ADMM for image restoration," in *Proc. IEEE International Conference on Acoustic, Speech, and Signal Processing*, Mar. 2017, pp. 1323–1327.

[67] S. V. Burtsev and Y. P. Kuzmin, "An efficient flood-fill algorithm," *Computers & Graphics*, vol. 17, no. 5, pp. 549–561, Sept 1993.

[68] S. H. Chan, R. Khoshabeh, K. B. Gibson, P. E. Gill, and T. Q. Nguyen, "An augmented Lagrangian method for total variation video restoration," *IEEE Trans. on Image Processing*, vol. 20, no. 11, pp. 3097–3111, May 2011.

[69] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3D transform-domain collaborative filtering," *IEEE Trans. on Image Processing*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

[70] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep cnn denoiser for image restoration," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, pp. 2808–2817.

[71] D. Cho, S. Kim, and Y. W. Tai, "Automatic trimap generation and consistent matting for light-field images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 39, no. 8, pp. 1504–1517, Aug. 2017.

[72] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van, M. Gross, and A. Sorkine-Hornung, "A benchmark dataset and evaluation methodology for video object segmentation," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 724–732.

[73] D. Fan, M. Chen, Y. Liu, T. Li, and A. Borji, "Structure-measure: A new way to evaluate foreground maps," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4548–4557.

[74] R. Guo, Q. Dai, and D. Hoiem, "Single-image shadow detection and removal using paired regions," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2011, pp. 2033–2040.

[75] E. Arbel and H. Hel-Or, "Shadow removal using intensity surfaces and texture anchor points," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 33, no. 6, pp. 1202–1216, Jun. 2011.

[76] F. Liu and M. Gleicher, "Texture-consistent shadow removal," in *Proc. European Conference on Computer Vision (ECCV)*. Oct. 2008, pp. 437–450, Springer.