N° d'ordre NNT : 041 2018

HABILITATION À DIRIGER DES RECHERCHES DE L'UNIVERSITÉ DE LYON
opérée au sein de
l'Université Claude Bernard Lyon 1

École Doctorale ED01
École Doctorale InfoMaths

Soutenue publiquement le 09/11/2018, par :
Nicolas Bonneel

Optimal Transport for Computer Graphics and Temporal Coherence of Image Processing Algorithms

Remerciements

Je remercie d'abord les rapporteurs et examinateurs de mon jury pour leur lecture attentive du manuscrit, les questions pertinentes lors de ma soutenance, mais aussi les riches discussions qui s'en sont suivies par email.

Je remercie mes co-auteurs (j'ai pu en dénombrer une soixantaine que je ne listerai pas ici!), étudiants et collègues du LIRIS. En particulier, je ne serais pas arrivé au CNRS sans l'accueil qui m'a été fourni auparavant en master, thèse et postdoctorats, par Matthias Paulin, George Drettakis, Michiel van de Panne, Bruno Lévy et Hanspeter Pfister.

Je remercie finalement mon père Stéphane pour son intérêt continuel, et ma femme Lucie pour m'avoir soutenu et accompagné lors de plusieurs années de vadrouille post-doctoresques en attente du Graal du poste permanent. Et je dédie ce manuscrit à Evan, notre fils, né le 19 Novembre 2017, presqu'un an avant cette soutenance.

Contents

Introduction (English)

This document summarizes my main contributions to computer graphics in the last nine years, since I defended my PhD in 2009. These contributions have spanned two distinct areas of computer graphics – Optimal Transport and Video Processing – but linked with the desire to provide numerical tools to the computer graphics community. Optimal transport is a trending tool with applications in computer graphics, including video processing, which I explore in a first chapter. As I explored different solutions to video processing problems that led to high impact contributions, I decided to expose these contributions in a separate chapter.

Optimal Transport. The Optimal Transport problem seeks a way to *warp* one function (or more precisely a probability measure) towards another, by minimizing a certain cost function modeling the effort one would require to move the function as if it were a pile of sand. This framework is particularly attractive to computer graphics as it produces visually appealing deformations. I developed computationally efficient optimal transport algorithms (Sec. 2.1 and 2.2), formulated inverse problems making use of the optimal transport theory (Sec. 2.3) and applied these algorithms to computer graphics problems throughout my work. I became aware of optimal transport theory at the end of my PhD thesis, while working on a photograph-based reflectance acquisition method for hair, that made use of the so-called *Earth Mover's Distance* [27]. I pursued the study of this fascinating theory during my post-docs and until now.

Video Processing. The second problem I addressed in this manuscript is that of making image processing algorithms stable when applied to video frames. This came from the observation that most image processing algorithms, when applied to videos, tend to produce flickering artifacts. My first attempts at solving this problem were focusing on two specific but well studied image processing problems: color grading (Sec. 3.1) and intrinsic decomposition (Sec. 3.2). Faced to the colossal project of adapting each possible image processing algorithm to videos, I then developed an all-purpose *blind* solution that handles most image processing algorithms, and further extended it to the case of videos taken from multiple cameras (Sec. 3.3). I entered the area of video processing during my post-doc at Harvard University funded by an NSF grant on video processing, and continued this line of work afterwards thanks to a long-lasting collaboration with Adobe.

Introduction (Français)

Ce document résume mes principales contributions à l'informatique graphique ces neuf dernières années, depuis que j'ai soutenu ma thèse en 2009. Ces contributions englobent deux pans distincts de l'informatique graphique – le transport optimal et le traitement de vidéos – mais reliés par le désir de fournir des outils mathématiques pour la communauté graphique. Le transport optimal est un outil en vogue avec des applications en informatique graphique, incluant le traitement vidéo, que j'explore dans un premier chapitre. En explorant diverses solutions liées aux problèmes de traitements vidéos ayant données des contributions de fort impact, j'ai décidé d'exposer ces contributions dans un chapitre distinct.

Transport Optimal. Le problème du transport optimal est celui de chercher une manière de *déformer* une fonction (ou plus précisément une mesure de probabilité) vers une autre, tout en minimisant une certaine fonction coût qui modélise l'effort qu'il faudrait fournir pour déplacer cette fonction comme s'il s'agissait d'un tas de sable. Ce cadre est particulièrement attractif en informatique graphique car il produit des déformations visuellement plaisantes. J'ai développé des algorithmes efficaces de transport optimal (Sec. 2.1 et 2.2), formulé des problèmes inverses utilisant la théorie du transport optimal (Sec. 2.3) et ai appliqué ces algorithmes à des problèmes d'informatique graphique tout au long de mon travail. Je pris connaissance de la théorie du transport optimal à la fin de ma thèse, lorsque je travaillais sur une méthode d'acquisition de la réflectance des cheveux à partir d'une photographie, qui utilisait ce qu'on appelle la *Distance du cantonnier* [27]. J'ai poursuivi cette fascinante théorie lors de mes post-doctorats et jusque maintenant.

Traitements Vidéos. Le second problème que j'expose dans ce manuscrit est celui de rendre des traitements d'images temporellement stables lorsqu'ils sont appliqués à chaque image de vidéos. Cela vient de l'observation que la plupart des algorithmes de traitement d'images, lorsqu'ils sont appliqués à des vidéos, ont tendance à produire des artefacts de scintillement. Mes première tentatives pour résoudre ce problème se sont focalisés sur deux problèmes spécifiques mais bien étudiés en traitements d'images : l'étalonnage de couleurs (Sec. 3.1), et la décomposition intrinsèque (Sec. 3.2). Face au projet colossal d'adapter chaque traitement d'image possible et imaginable aux vidéos, j'ai ensuite développé une solution générique *aveugle* qui supporte la plupart des algorithmes de traitements d'images, et l'ai étendu pour gérer le cas de vidéos prises par plusieurs caméras en même temps (Sec. 3.3). Je suis entré dans le domaine du traitement vidéo pendant mon post-doctorat à l'Université d'Harvard, financé par une subvention de la NSF sur le traitement de vidéos, et ai continué cette ligne de recherche par la suite, grâce à ma continuelle collaboration avec Adobe.

1 Background

This chapter is dedicated to the introduction of the two problems being tackled, numerical techniques used and related work. While it aims at defining the introduced problem, more context will be presented in subsection 1.1.7 and subsection 1.1.8.

1.1 The Optimal Transport Problem

This section is intended to be a well-illustrated but concise exposition of optimal transport, its history and recent numerical techniques for solving it, along with concurrent approaches. Most illustration are one-dimensional for clarity but extend to higher dimensions. The interested reader can find an in-depth coverage of optimal transport in Cedric Villani's monograph [186], numerical tools in the recent books of Peyré and Cuturi [136] and Filippo Santambrogio [158], and its historical context in the articles of Vershik [184] and Schrijver [162].

1.1.1 The Monge Problem

In a memoir published in 1781 [120], Gaspard Monge, a French mathematician, formulated the earliest known instance of a linear programming problem precursor. Monge was interested in solving practical problems of operations research, and first described the *Optimal Transport* problem. This problem aims at finding the most efficient way to move a pile of sand to a hole in the ground, while minimizing the total distance traveled by all particles of sand from their location in the pile to their destination in the hole (see Fig. 1.1 for an illustration in 1D, though the problem works in higher dimension too).

The above problem is known as the *Monge Problem.* In modern mathematical language, and with some generalization, it consists in finding a mass preserving map T between probability measures μ_0 and μ_1 minimizing:

$$W(\mu_0, \mu_1) = \inf_T \int_{\mathcal{M}} c(x, T(x)) \mathrm{d}\mu_0(x). \tag{1.1}$$

Here, $\mathcal{M}$ denotes a Riemannian manifold along which the sand is transported, since the Earth is not necessary flat (though in most cases, we will use the Euclidean space). $c(x, T(x))$ is called the *ground cost* and denotes the cost of transporting one particle of sand from location x in the manifold to its target location $T(x)$. Monge originally used $c(x, T(x)) = |x - T(x)|$, the Euclidean

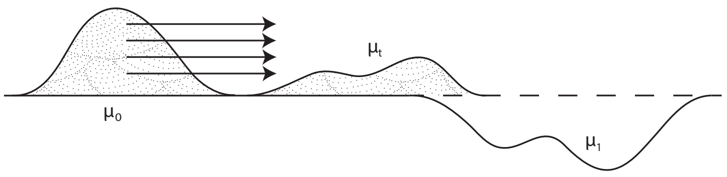


Figure 1.1: The mass transportation problem consists in optimally moving a probability measure μ_0 represented by a pile of sand, towards a probability measure μ_1 making a hole. At an intermediate time $t \in [0, 1]$, an interpolated probability measure μ_t, the *displacement interpolation*, is obtained. A *Wasserstein barycenter* generalizes this notion to more than 2 input probability measures.

distance, but most optimal transport research has focused on the simpler case $c(x, T(x)) = |x - T(x)|^2$ (called the *quadratic cost*) or $c(x, T(x)) = d(x, T(x))^2$ where $d(x, T(x))$ is the geodesic distance from x to $T(x)$ along the manifold. We recall a probability measure μ is a positive measure with $\mu(\mathcal{M}) = 1$. The "mass preservation" constraint is imposed on T and states that once T is applied to μ_0, it indeed reconstructs μ_1. This imposes that the measure of all subsets be preserved by T, or said differently, that the following constraint be enforced:

$$\text{s.t.} \qquad \forall \Omega \in \mathcal{M}, \mu_0(T^{-1}(\Omega)) = \mu_1(\Omega) \tag{1.2}$$

In that case, μ_1 is also called the pushforward measure of μ_0 by T, and Eq. 1.2 can be noted $\mu_1 = T\#\mu_0$.

In many cases, we will deal with absolutely continuous measures. This means that μ_0 and μ_1 can be represented by their respective probability density functions f_0 and f_1 and the Lebesgue measure as $\mathrm{d}\mu_0 = f_0\,\mathrm{d}x$ and $\mathrm{d}\mu_1 = f_1\,\mathrm{d}x$. This essentially precludes measures consisting of Dirac distributions. In that case, Eq. 1.1 and 1.2 can be simplified to:

$$W(f_0, f_1) = \inf_T \int_{\mathcal{M}} c(x, T(x)) f_0(x) \mathrm{d}x \tag{1.3}$$

$$\text{s.t.} \qquad f_0(x) = f_1(T(x))\, |\det \mathrm{Jac}\, T(x)|, \tag{1.4}$$

where $\det \mathrm{Jac}\, T(x)$ is the determinant of the Jacobian of T (one can recognize in Eq. 1.4 the standard change of variable formula for integration). Existence of Monge solutions is an active topic in mathematics [5]. When solutions exist, and when the ground cost is quadratic, Brenier showed that the optimal T is the gradient of a convex function [36]. In 1-D, this simply means that T is non-decreasing (see Fig. 1.2)) While Monge successfully cast and studied this problem, he did not manage to find a solution. The Jacobian determinant makes this problem highly non-linear, and his focus on the Euclidean distance as a ground cost rendered the problem particularly complex to solve.

$W(\mu_0, \mu_1)$ defines the overall cost of transporting the pile of sand μ_0 into the hole μ_1. In fact, when the ground cost can be expressed as $c(x, T(x)) = d(x, T(x))^p$ with $p > 1$ and d the geodesic distance, it can be shown that $W(\mu_0, \mu_1)^{\frac{1}{p}}$ defines a mathematical distance between μ_0 and μ_1. This distance is commonly refered to as the p^{th} **Wasserstein distance** and is often denoted $W_p(\mu_0, \mu_1)$. However, when $0 < p < 1$ (or more generally, for concave cost functions), there is in general no solution if μ_0 and μ_1 overlap [185]. For $p = 1$, solutions exist if measures are absolutely continuous, but are in general non unique.

The facts that optimal transport defines a distance W_p between probability measures and this distance accounts for the motion of mass are the main motivations for its adoption in computer graphics, and for the work presented in this manuscript.

In practice, computers deal with discrete representations of probability density functions, such as histograms or signatures [153], and we will expose recent numerical techniques for optimal transport in Sec. 1.2.

1.1.2 Kantorovich Reformulation

Leonid Vitaliyevich Kantorovich is a Russian mathematician, who formulated the first solution of the optimal transport problem in 1942 [88]. While Kantorovich was busy inventing linear programming, the celebration of the bicentenary of Monge's birthday led the Commission on History of Mathematical and Physical Sciences of the USSR Academy to hold a public session in 1947 in Monge's honour [184]. The proceedings of this session ended up in the hands of Kantorovich, who drew connections to his work [89]. He finally got awarded the Nobel Prize in economics in 1975, along with Tjalling C. Koopmans, for his work on the theory of optimum allocation of resources.

Contrary to his previous articles on finite dimensional linear programming, his 1942 paper focused on the following infinite dimensional linear programming problem, now called the *Monge-Kantorovich* problem:

$$W(\mu_0, \mu_1) = \inf_{\pi \in \Gamma(\mu_0,\mu_1)} \int_{\mathcal{M}} \int_{\mathcal{M}} c(x, y) \mathrm{d}\pi(x, y), \tag{1.5}$$

where

$$\Gamma(\mu_0, \mu_1) = \{\pi \in Pr(\mathcal{M} \times \mathcal{M}) \mid \forall \Omega \in \mathcal{M}, \pi(\mathcal{M} \times \Omega) = \mu_1(\Omega), \pi(\Omega \times \mathcal{M}) = \mu_0(\Omega)\},$$

and Pr the set of probability measures. Γ hence represents the set of probability measures whose projections (or *marginals*) are μ_0 and μ_1. As this amounts to a linear program, a dual problem can be expressed:

$$\begin{aligned} W(\mu_0, \mu_1) &= \sup_{\varphi_0,\varphi_1} \int_{\mathcal{M}} \varphi_0(x)\mathrm{d}\mu_0(x) + \int_{\mathcal{M}} \varphi_1(y)\mathrm{d}\mu_1(y) \\ \text{s.t.} \quad & \varphi_0(x) + \varphi_1(y) \leq c(x, y). \end{aligned}$$

Solutions to this problem always exist since the plan obtained by the product $\pi = \mu_0 \otimes \mu_1$ is always admissible (unless $c \equiv \infty$). The interesting thing is that, when solutions of the problem 1.1 exist, they correspond to solutions of problem 1.5 and produce the same transportation cost [69, 142]. These solutions are given by $\pi(\mathrm{dx}, \mathrm{dy}) = \delta(y - T(x))\mathrm{dx}$. Said differently, when a solution T of the Monge problem exists, a solution π of the Monge-Kantorovich problem exists and the

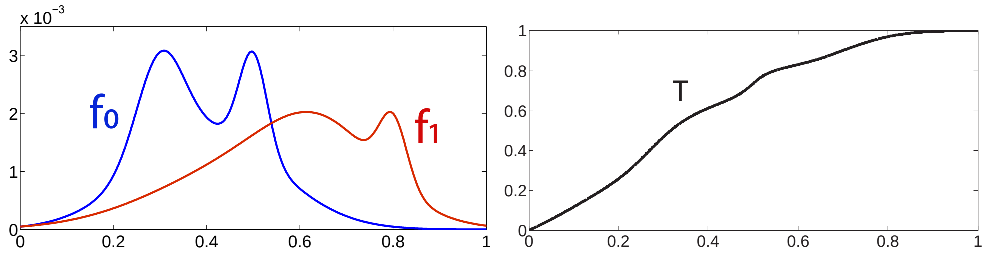


Figure 1.2: Given probability density functions f_0 and f_1 (left) the Monge problem uncovers an optimal map T (right) from the domain of f_0 to the domain of f_1 (here, both live in the interval $[0, 1]$. In 1-D, T is necessarily non-decreasing for any convex ground cost.

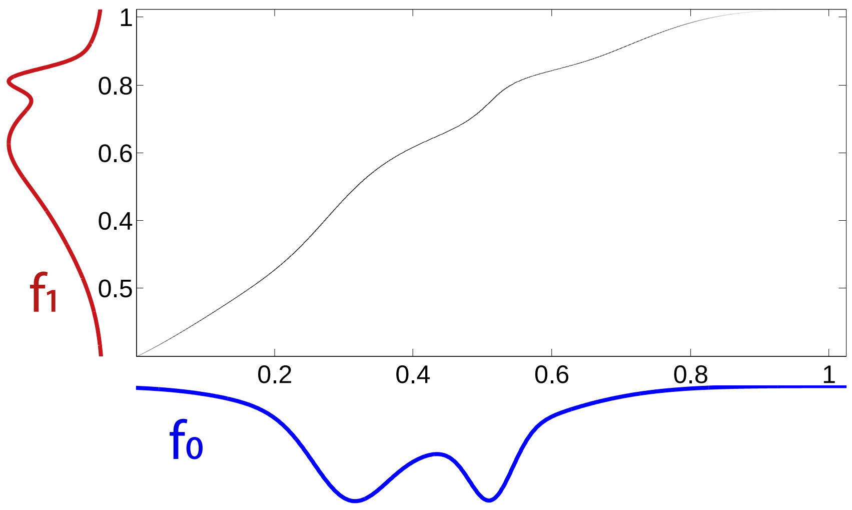


Figure 1.3: Similarly to Fig. 1.2, we transport f_0 to f_1 shown in blue and red but using Kantorovich's formulation. This amount to computing a large but sparse matrix (here, zeros are shown in white) called the transport plan, whose entry (i, j) indicates the amount of mass going from bin i to bin j. It can be shown that when a Monge solution exist, Kantorovich's transport plan corresponds to the graph of Monge's transport map, as can be seen here in black.

mass of the measure π is concentrated on the graph of T (see Fig. 1.3). Both problems are thus essentially the same. It is also known that the set of solutions to 1.5 is broader than that of 1.1 [176]. For instance, it is easy to build a mass transport problem between a single Dirac ($\mu_0 = \delta_0$) and two half-Diracs ($\mu_1 = \frac{1}{2}\delta_{-1} + \frac{1}{2}\delta_1$), and in this case, Monge solutions do not exist as a map cannot warp a single Dirac into two, but Kantorovich solutions exist and allow for the splitting of mass. It is shown that when μ_0 does not give mass to small sets (e.g., no Diracs, nor mass supported on submanifolds), then Kantorovich solutions always exist. When the ground cost is $c(x, y) = |x - y|^p$ with $p > 1$, and μ_0 and μ_1 are absolutely continuous, then solutions to the Kantorovich problems exist and are unique.

Discretizing 1.5 may help to understand the problem from a computer scientist perspective. Assuming μ_0 is discretized as a histogram called h_0 over m bins, and μ_1 as a histogram h_1 over n bins, the discretized Eq. 1.5 amounts to the following finite-dimensional linear programming problem:

$$\inf_\pi \sum_{i,j} c(x_i, y_j)\pi_{i,j} \tag{1.6}$$

$$\text{s.t.} \qquad \pi_{i,j} \geq 0 \qquad \forall i, j \tag{1.7}$$

$$\sum_{i=1}^{n} \pi_{i,j} = h_1(x_j) \qquad \forall j \tag{1.8}$$

$$\sum_{j=1}^{m} \pi_{i,j} = h_0(x_i) \qquad \forall i \tag{1.9}$$

$$\tag{1.10}$$

where now, π is represented by a $m \times n$ matrix. It is a linear program with mn unknowns, mn inequality constraints and $m + n$ equality constraints. It can be shown that the solution of such a linear program is a matrix with at most $m + n - 1$ non-zero elements [64], and can be obtained via linear programming solvers such as the simplex method [84] though at a fairly high computational cost (typically, in roughly $O(n^3)$ in practice or worst case exponential). When coefficients are integers, solutions also are. The particular case where $m = n$ and the only possible values for h_0 and h_1 are 0 or 1 amounts to an assignment problem. In this case, the solution π can only take values 0 or 1.

In the literature, the optimal transport problem has then arised with various names, giving rise to the *Earth Mover's Distance* (or EMD, or as I found, in French, “la distance du cantonnier”), 1st Mallow's distance, the Kantorovich-Rubinstein problem, the Monge-Kantorovich problem, the Wasserstein distance, the Hitchcock-Koopmans problem...

1.1.3 Displacement Interpolation and Wasserstein Barycenters

Since optimal transport defines a distance W_p between probability measures μ_0 and μ_1, it is tempting to construct probability measures μ_t ($0 \le t \le 1$) that interpolate this distance. This would correspond to finding measures *in-between* μ_0 and μ_1 in the sense of the W_p distance. For the case W_2, this notion has been introduced by Robert McCann in 1997 [117] that he called *displacement interpolation.* Using Monge's framework, this finds an optimal transport map T between μ_0 and μ_1, but uses it to move the pile of sand μ_0 only partway towards μ_1. This corresponds to applying to μ_0 a linearly interpolated map T_t between the identity map and T : $T_t = (1-t)\mathrm{Id} + t\,T$. We obtain the displacement interpolated measure $\mu_t = T_t\#\mu_0$ (or using densities, $f_t(T_t(x))\,|\det \mathrm{Jac}\,T_t| = f_0(x)$). This interpolation smoothly advects one measure towards the other, and this motion makes it attractive for visual effects studied in computer graphics (e.g., Figure 2.1 or Figure 2.9).

Once a notion of interpolation between two measures has been defined, one could ask how to generalize this interpolation to more than two measures. This type of interpolation defines a barycenter of measures, and Agueh and Carlier have introduced this concept as that of *Wasserstein barycenter* [1]. By analogy with Euclidean barycenters, they introduced Wasserstein barycenters using Fréchet mean. Given measures along with barycentric weights $\{(\mu_i, \lambda_i)\}_{i=1..N}$, the Wasserstein barycenter $\overline{\mu}$ is defined as the minimizer:

$$\overline{\mu} = \mathrm{argmin}_{\mu} \lambda_i W_2^2(\mu, \mu_i) \qquad (1.11)$$

Using the Monge-Kantorovich formulation of optimal transport, they show that the minimizer exists and is unique if at least one of the μ_i does not give mass to small sets. **The notion of Wasserstein barycenter is the second pilar of my work.**

1.1.4 Riemannian Geometry

It is tempting to use optimal transport between two infinitesimally close probability distributions. In fact, optimal transport with a quadratic ground cost defines a Riemannian metric, and the displacement interpolation μ_t between two probability measures μ_0 and μ_1 corresponds to the geodesic between μ_0 and μ_1 in the infinite-dimensional manifold of probability measures endowed with this Riemannian metric. The cost W_2 is the geodesic distance.

To define this manifold, one need to characterize its tangent space. A tangent vector $\frac{\partial\rho}{\partial t}$ at a measure with density ρ (see the tangent vector as an infinitesimal variation of the density ρ, see Fig. 1.4) is such that $\frac{\partial\rho}{\partial t} = -\nabla\cdot(\rho\nabla u)$ with ∇u a velocity field [186]. The scalar product of two tangent vectors (i.e., the Riemannian metric) can then be defined as $\langle\frac{\partial\rho}{\partial t_1}, \frac{\partial\rho}{\partial t_2}\rangle_\rho = \int \rho\langle\nabla u_1, \nabla u_2\rangle$, with ∇u_1 and ∇u_2 both velocity fields characterizing each tangent vectors. Minimal length geodesics using this Riemannian metric indeed correspond to displacement interpolation.

I will make use of the Riemannian geometry of optimal transport in Sec. 3.1 to compute

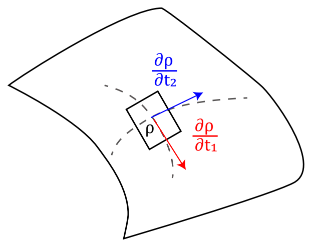


Figure 1.4: In the infinite dimensional Riemannian manifold of probability distributions, each probability distribution (here with a density ρ) is a point, and tangent vectors can be defined using infinitesimal variations of probability measures. Given two tangent vectors $\frac{\partial\rho}{\partial t_1}$ and $\frac{\partial\rho}{\partial t_2}$, a scalar product between them defines the Wasserstein Riemannian metric.

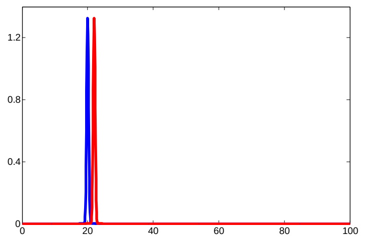


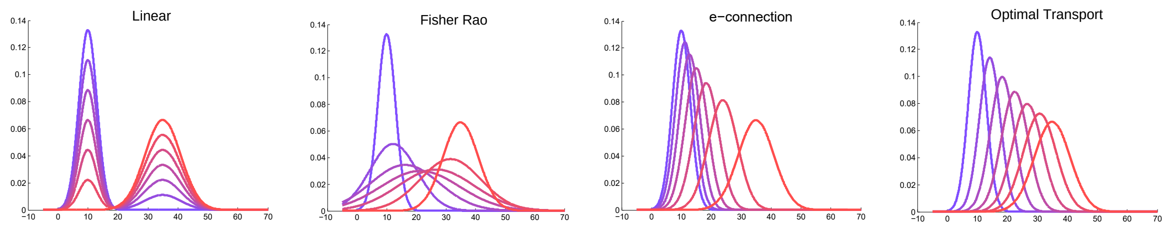
Figure 1.5: Two high-peaked distributions of disjoint support always have an Euclidean distance of 2. In contrast, optimal transport accounts for their motion, here resulting in a small distance.

the curvature of a curve of Gaussian densities within this manifold (see also Sec. 1.1.6 for the particular case of Gaussian densities) for video processing purposes.

1.1.5 Other metrics

Other metrics and ways of interpolating between probability distributions have been proposed. The simplest way is the Euclidean geometry along with the linear interpolation of probability measures: $(1-t)\mu_0+\mu_1$. This is the appropriate interpolation in many applications. For instance, building the age pyramid of a country without accounting for gender can be performed by taking the sum of the age pyramid of women and that of men. More generally, drawing m male and n female individuals from a population results in a distribution that is a linear combination of male and female distributions with weights $\frac{m}{m+n}$ and $\frac{n}{m+n}$ respectively. However, as it does not model any horizontal motion, it will not behave well for instance when trying to model the evolution in time of age pyramids (e.g., see my vulgarization article, in French but illustrated with animated gifs, on optimal transport [25]). In general, computer graphic applications often rely on horizontal motions as we shall see in Sec. 1.1.7. In term of distance measure, the Euclidean distance between two probability distributions measures their overlap, such that two distributions of disjoint support will always have the same distance equal to 2 ; in particular, when two near-Dirac distributions are compared, the Euclidean geometry is rarely useful (see Fig. 1.5).

Information Geometry is a field interested in studying the differential geometry of probability distributions – I will refer to the excellent book of Amari for a clear and fascinating introduction to the subject [4]. However, the Wasserstein metric has essentially been left out of this field in favor of the Fisher metric. The Fisher metric provides an alternative Riemannian metric to manipulate, interpolate and compute distances between probability distributions. In addition, when there is no need for an actual distance to be computed, information geometry provides other tools to manipulate probability distributions. In particular, parallel transport is a concept



Linear Fisher Rao e-connection Optimal Transport

Figure 1.6: Various metrics or ways to interpolate two simple Gaussian distributions.

of differential geometry that allows to *move* points (or, in this context, probability distributions) without the need for a Riemannian metric but only a *connection*. A connection is essentially a way of moving on a manifold by describing how tangent spaces vary from a point to a neighboring point. While they do not provide a way to compare probability distributions, these connections, called α-connection in the context of information geometry, allow for powerful applications such as statistical inference and have $\alpha \in [-1, 1]$ as a degree of freedom [1].

I will not discuss the generality of connections further. Instead, I will illustrate in Fig. 1.6 how Euclidean metric, the Fisher metric, the e-connection (which is the α-connection with $\alpha = 1$, which is most suited to distributions of the exponential family such as Gaussians) and optimal transport behave when simply interpolating between two Gaussian distributions. Formulas to interpolate Gaussian distributions according to the Fisher metric are available in a paper by Xia et al. [191], while formulas for the e-connection can be found in Amari's book [4].

1.1.6 Particular Cases

Optimal Transport in 1D When probability measures are supported on the real line, optimal transport becomes much easier [186]. This section will further assume the ground cost is a strictly convex function of the distance (e.g., $c(x, y) = |x - y|^n$ with $n > 1$). In that case, the optimal transport map between probability measures μ_0 and μ_1 is given by:

$$T = M_1^{-1} \circ M_0$$

where M_0 (resp. M_1) is the cumulative distribution function $M_0 = \int_{-\infty}^{x} d\mu_0 = \mu_0[(-\infty, x)]$ and the generalized inverse M_1^{-1} is defined by $M_1^{-1}(t) = \inf\{x \in \mathbb{R}; M_1(x) > t\}$. This amounts to progressively accumulating mass from μ_1 until it matches that of μ_0. A discrete version of this procedure is expressed in a short Matlab function of linear time complexity working on histograms h_0 and h_1:

```
function T = transport1d(h0, h1)
    cdf1 = cumsum(h0);
    cdf2 = cumsum(h1);
    j=1;
    for i=1:size(h0, 1)
        while(cdf2(j)<cdf1(i)) j = j+1; end;
        T(i) = j;
    end
```

[1]When a Riemannian metric is available, such as the Fisher or Wasserstein metric, one can use the Levi-Civita connection obtained from the metric. In the case of the Wasserstein metric, this recovers the usual theory of the differential geometry of optimal transport, including displacement interpolation as minimizing geodesics. The α-connection with $\alpha = 0$ is the Levy-Civita connection of the Fisher metric, but when $\alpha \neq 0$, they are not derived from a metric.

The optimal transport cost can be obtained using $W(\mu_0, \mu_1) = \int_0^1 c(M_0^{-1}(t), M_1^{-1}(t))\mathrm{d}t$, or, in the specific case of $c(x, y) = |x-y|$, this can be simplified to $W(\mu_0, \mu_1) = \int_{\mathbb{R}} |M_0(x) - M_1(x)|\mathrm{d}x$.

Now, using a quadratic ground cost, the displacement interpolation can then be obtained using $\mu_t = ((1-t)\mathrm{Id} + t\,T)\#\mu_0$, but a computationally more interesting option directly provides the Wasserstein barycenter between any number of probability measures associated with their weights $\{(\mu_i, \lambda_i)\}_{i=1..N}$:

$$\overline{M}^{-1} = \sum_{i=1}^{N} \lambda_i M_i^{-1} \tag{1.12}$$

where again $\overline{M}^{-1}$ denotes the generalized inverse of the cumulative distribution function of the desired barycenter $\overline{\mu}$. As can be seen by the expression of the Wasserstein distance for quadratic costs, the space $W_2(\mathbb{R})$ has zero curvature – more details in the geometric study of Kloeckner [92].

When dealing with 1D discrete measures consisting of a sum of Diracs with uniform weights and with the same number of atoms, the optimal transport map between two such measures is easily obtained by sorting the positions of each Dirac in both measures, and progressively pairing them from left to right. I will make use of 1D optimal transport to define Radon barycenters in Sec. 2.2.

Optimal Transport of Gaussian Measures

A second case for which optimal transport can be expressed in closed form is that of Gaussian measures with quadratic ground cost [56, 1, 62]. Given two Gaussian $\mu_0 = \mathcal{N}(\Sigma_0, m_0)$ and $\mu_1 = \mathcal{N}(\Sigma_1, m_1)$, the optimal transport cost is then given by:

$$W(\mu_0, \mu_1)^2 = \mathrm{tr}(\Sigma_0 + \Sigma_1 - 2\Sigma_{0,1}) + \|m_0 - m_1\|^2 \tag{1.13}$$

with $\Sigma_{0,1} = (\Sigma_0^{1/2}\Sigma_1\Sigma_0^{1/2})^{1/2}$.

The displacement interpolation is also a Gaussian $\mu_t = \mathcal{N}(\Sigma_t, (1-t)m_0 + t\,m_1)$, with

$$\Sigma_t = [(1-t)\mathrm{Id} + t\,P]\Sigma_0[(1-t)\mathrm{Id} + t\,P] \tag{1.14}$$

and $P = \Sigma_1^{1/2}\Sigma_{0,1}^{+}\Sigma_1^{1/2}$, denoting $\Sigma_{0,1}^{+}$ the Moore-Penrose pseudo-inverse of $\Sigma_{0,1}$. Here P is the Monge transport map between the two centered Gaussians.

The Wasserstein barycenter of multiple Gaussians $\{\mu_i = \mathcal{N}(\Sigma_i, m_i)\}_{i=1..N}$ with weights $\{\lambda_i\}_{i=1..N}$ is also a Gaussian $\overline{\mu} = \mathcal{N}(\Sigma, \sum_{i=1}^{N} \lambda_i m_i)$ with Σ obtained by solving a fixed point equation, by taking the limit of the iterates:

$$\Sigma^{(n+1)} = \sum_{i=1}^{N} \lambda_i (\Sigma^{(n+1)^{1/2}} \Sigma_i \Sigma^{(n+1)^{1/2}})^{1/2}$$

As can be seen above, the space of Gaussian measures is a totally geodesic submanifold of W^2, meaning that any geodesic in this submanifold is also a geodesic in W^2 (since the displacement interpolation between Gaussians is always a Gaussian). Takatsu has thus derived the expression of the Riemannian metric in this submanifold [178]. The space of Gaussian measures is the product of the space of 0-mean Gaussian measures and the Euclidean space of translations. So, in the rest of this section, we will only consider 0-mean Gaussian measures, and we will identify it with the space of their covariance matrices (a space of symmetric positive definite matrices).

Let Σ be a covariance matrix at which the metric is desired. Takatsu's method makes use of the Riemannian submersion $\Pi(A) = A^\top A = \Sigma$ with A in the space of general real matrices.

Then, given two matrices U and V in the tangent space of Σ (the tangent space of symmetric positive matrices is the space of symmetric matrices), we solve for symmetric matrices X and Y such that $U = \Sigma X + X\Sigma$ and $V = \Sigma Y + Y\Sigma$ (a so-called Lyapunov or Sylvester equation). Then the Riemannian metric can be expressed as $g_\Sigma(U, V) = \mathrm{tr}(X\Sigma Y)$.

I will make extensive use of Gaussian optimal transport in Sec. 3.1 for video processing purpose.

1.1.7 Optimal Transport in Computer Graphics

Optimal transport has attracted much attention in recent years in the computer graphics and vision communities for several reasons:

- Optimal transport deals with histograms and probability distributions, which are frequently encountered in computer graphics (color histograms, reflectance functions, several histogram-based image descriptors). It has clear benefits over other metrics as we have seen in Sec. 1.1.5.
- Optimal transport also deals with mass being transported, which is attractive for other applications such as geometry processing, and models a perceptually meaningful motion (e.g., the motion of highlights in reflectance functions, the warping of shapes and images)
- Optimal transport defines relevant metrics to compare histograms
- Optimal transport has recently seen practical numerical methods
- Optimal transport has a beautiful underlying theory

Aside from my own (and co-authors') contributions that will be described in the next chapter, it is thus not surprising to find numerous applications in computer graphics. Notably for retrieving images in a collection based on their color histogram [153], for altering colors in an image to match those of another image [140, 143, 174], to interpolate between two tetrahedralized meshes [107] or produce barycenters of several voxelized 3d objects [174], as a metric to fit shapes and meshes to point clouds [47, 51], for mesh parameterization [53] and retrieval [144, 110], for registering and warping images [79, 119, 129], to interpolate between simulated caustics [77], and to interpolate between multiple reflectance functions [174]. Other intriguing applications include the computation of geodesics on meshes [175] and blue noise sampling patterns [46], the design of transparent or reflecting caustic-producing shapes [163, 6], and fluid dynamics simulation [68]. Applications in image processing are detailed in Nicolas Padadakis' habilitation [128].

It is interesting to note that the vast majority of these applications are less than 10 years old. This is mostly because until recently optimal transport was too computationally costly for being used in practice. The situation has evolved, and the section 1.2 will illustrate the two major approaches.

1.1.8 Optimal Transport in Machine Learning

Optimal transport provides a loss function suitable for comparing histogram data. It also provides ways of interpolating and a whole Riemannian geometry. This makes the theory ideal for replacing Euclidean concepts often encountered in machine learning by the Wasserstein geometry.

Several recent papers have investigated the use of optimal transport distances as fitting losses that have desirable properties that KL or Euclidean distances cannot offer. For instance, Principal Component Analysis (PCA) has been generalized to the set of probability measures via the use of optimal transport distances [23, 165]. Sandler and Lindenbaum first considered the Non-Negative Matrix Factorization (NMF) problem with a Wasserstein loss [157]. Their computational approach was, however, of limited practical use. More scalable algorithms for Wasserstein NMF and (linear) dictionary learning were subsequently proposed [151]. The Wasserstein distance was also used as a loss function with desirable robustness properties to address multilabel supervised learning problems [66].

Using the Wasserstein distance to quantify the fit between data (an empirical measure) and a parametric family of densities, or a generative model defined using a parameterized push-forward map of a base measure, has also received ample attention in the recent literature. Theoretical properties of such estimators were established by Bassetti, Bodini, and Regazzini [14] and Bassetti and Regazzini [15], and additional results by Bernton et al. [19]. Entropic smoothing (see subsection 1.2.1) has facilitated the translation of these ideas into practical algorithms, as illustrated in the work by Montavon, Müller, and Cuturi, who proposed to estimate the parameters of restricted Boltzmann machines using the Wasserstein distance instead of the KL divergence [121]. Purely generative models, namely, degenerate probability measures defined as the push-forward of a measure supported on a low-dimensional space into a high-dimensional space using a parameterized function, have also been fitted to observations using a Wasserstein loss [19], allowing for density fitting without having to choose summary statistics (as is often the case with usual methods). The Wasserstein distance has also been used in the context of generative adversarial networks (GANs) [7]. In that work, the authors use a proxy to approximate the 1-Wasserstein distance. Instead of computing the 1-Wasserstein distance using 1-Lipschitz functions, a classical result from Kantorovich's dual formulation of optimal transport (see Theorem 1.14 in Villani's book [185]), the authors restrict that set to multilayer networks with rectified linear units and boundedness constraints on weights, which allows them to enforce some form of Lipschitzness of their networks. Unlike entropic smoothing, that approximation requires solving a nonconvex problem whose optimum, even if attained, would be arbitrarily far from the true Wassertein distance. More recently, Genevay, Peyré, and Cuturi introduced a general scheme for using optimal transport distances as the loss in generative models [73], which relies on both the entropic penalty and automatic differentiation of the Sinkhorn algorithm.

Our work will exploit the geometry of optimal transport for machine learning – such as Wasserstein Barycentric Coordinates and Wasserstein Dictionary Learning (section 2.3).

1.2 Numerical Optimal Transport

Optimal transport was, until recently, very costly, and mostly relying on the resolution of the linear programming problem via classical operational research approaches (the Hungarian method [99], the auction algorithm [20], the transportation simplex [83] or network simplex [42]), or the resolution of partial differential equations [17, 79, 53, 129]. Optimal transport has seen some breakthrough in the last decade, mostly coming from two different approaches: entropy regularized optimal transport, and semi-discrete optimal transport. I will expose both approaches in this section. In Sec 2.2, I will expose a third even faster algorithm which I developed in the same period, based on Radon transforms, but that only solves an approximate transport problem. For completeness, this section will also briefly introduce an approach that solves a similar approximation, called *Sliced optimal transport*.

```
function SINKHORN($\mu_0, \mu_1$)
    $a, b \leftarrow \mathbb{1}$
    for $i = 1, 2, \ldots, L$
        $a \leftarrow \frac{p_0}{Kb}$
        $b \leftarrow \frac{p_1}{K^T a}$
    return $\pi_\gamma \leftarrow diag(b) K diag(a)$
```

Algorithm 1: Given two histograms p_0 and p_1, this function computes the entropy-regularized transport plan π_γ.

1.2.1 The Sinkhorn Algorithm

In a 2013 seminal paper [43], Cuturi has brought the Sinkhorn algorithm for optimal transport [171] to light. This algorithm considers a slightly modified functional, that adds an entropic regularization term (originally, it looked for solutions with bounded entropy). Doing so, transport plans become slightly blurred, but the method becomes trivial to implement and very fast. Its convergence speed depends on the regularization parameter, and, as the transport plan becomes more blurry, it converges faster. The new functional reads:

$$W_\gamma(\mu_0, \mu_1) = \inf_{\pi \in \Gamma(\mu_0, \mu_1)} \int_{\mathcal{M}} \int_{\mathcal{M}} c(x, y) \mathrm{d}\pi(x, y) - \gamma H(\pi) \tag{1.15}$$

where γ is the regularization factor, and $H(\pi) = -\int_{\mathcal{M}} \int_{\mathcal{M}} \pi(x, y) \log \pi(x, y) \mathrm{dxdy}$ the entropy (with $H(\pi) = -\infty$ when π is not absolutely continuous). This functional is often rewritten in vector form as follows. For two matrices A, B of the same size, denoting $\langle A, B\rangle = \mathrm{tr}(A^T B)$ their usual inner-product, where A^T is the transpose of A, the entropy regularized distance between two histograms p and q discretized over N bins can be written:

$$W_\gamma(p, q) \overset{\text{def.}}{=} \min_{T \in \mathbb{R}_+^{N \times N}} \left\{ \langle T, C \rangle - \gamma H(T) \; ; T\mathbb{1} = p, T^T \mathbb{1} = q \right\}, \tag{1.16}$$

where the matrix $C = (c_{i,j})_{i,j}$ quantifies the cost of transporting mass between histogram bins. This functional can be minimized via iterative Bregman projections, leading to the so-called Sinkhorn algorithm (see Alg. 1). All multiplications and divisions in this iterative algorithm are performed component-wise on vectors. This algorithm makes uses of a kernel matrix K defined as $K = \exp(-C/\gamma)$, where exp is again a component-wise exponential on matrix values.

This also extends the definition of Wasserstein barycenters to that of regularized barycenters using the resulting smoothed cost W_γ, leading to a fast algorithm (Alg. 2) [18]. Iterations of this algorithm correspond to projections for the KL divergence on a set of affine constraints.

Definition 1. Given a family of S normalized input histograms $(p_s)_s$, the barycentric map P associates to a vector $\lambda \in [0, 1]^S$ with $\sum_s \lambda_s = 1$, the barycenter of $(p_s)_s$ with weights λ, uniquely defined as

$$P : \lambda \mapsto P(\lambda) \overset{\text{def.}}{=} \underset{p \in \Sigma_N}{\operatorname{argmin}} \sum_s \lambda_s W_\gamma(p, p_s). \tag{1.17}$$

where Σ_N denotes the simplex of normalized histograms of N bins. The uniqueness of $P(\lambda)$ comes from the strong convexity (as a function of p) of the energy defined on the right-hand side of Eq. (1.17), itself inherited from the regularization term in Eq. (1.16).

In particular cases, the operator $K = exp(-c/\gamma)$ is translation invariant, and applying K on a vector can be implemented via convolutions instead of matrix-vector products [174]. In particular, the quadratic ground cost $c(x, y) = ||x - y||^2$ makes the kernel K a Gaussian convolution kernel,

```
function SINKHORN-BARYCENTER($(p_s)_{s=1}^S, q, \lambda$)
    $\forall\, s, b_s^{(0)} \leftarrow \mathbb{1}$
    $(w, r) \leftarrow (0^S, 0^{S\times N})$
    for $\ell = 1, 2, \ldots, L$
        $\forall\, s, \varphi_s^{(\ell)} \leftarrow K^\top \frac{p_s}{K b_s^{(\ell-1)}}$
        $p \leftarrow \prod_s \left(\varphi_s^{(\ell)}\right)^{\lambda_s}$
        $\forall\, s, b_s^{(\ell)} \leftarrow \frac{p}{\varphi_s^{(\ell)}}$
    return $P^{(L)}(\lambda) \leftarrow p$
```

Algorithm 2: Given a database of histograms $(p_s)_{s=1}^S$ and weights λ, the function computes the entropy regularized Wasserstein barycenter $P^{(L)}(\lambda)$

that can be efficiently implemented [2]. However, this method is particularly sensitive to the choice of γ, as too small values tend to make the the iterations diverge due to divisions by increasingly small numbers. More robust yet computationally more complex approaches perform Sinkhorn iterations in the log domain [161, 40, 115].

1.2.2 Semi-Discrete Formulation

This sections briefly exposes a formulation that I do not make use of within the contributions detailed in this manuscript (though my current work investigates this formulation as discussed in section 4.1). It is exposed nevertheless for completeness, but the hasty reader might choose to skip this section.

A special case of optimal transport arises when considering the input distribution μ_0 with a density, and a target distribution μ_1 being discrete (that is, μ_1 is a weighted sum of Dirac measures with weights $\{p_i\}_i$). Aurenhamer showed that in this case, the transport map can be expressed as a power diagram [9], that is, a weighted Voronoi diagram with weights $\{\varphi_i\}_i$ (Figure 1.7).

Definition Given a domain X, a finite set of weighted samples $(x_i)_{1\leq i\leq n} \subset X$ coupled with weights $(\varphi_i)_{1\leq i\leq n} \subset \mathbb{R}$, a power diagram (or Laguerre's diagram) is a partition of the space into cells $\{Pow(x_i)\}_i$ given by

$$Pow(x_i) \overset{\text{def.}}{=} \{x \in X \mid \|x - x_i\|^2 - \varphi_i \leq \|x - x_j\|^2 - \varphi_j\}$$

The transport map will transport all the mass within a power cell $Pow(x_i)$ of mass $\mu_0(Pow(x_i))$ towards its corresponding seed located at x_i in μ_1. Finding the transport map thus amounts to finding the power weight φ_i associated to each sample x_i such that the mass of each power cell $\mu_0(Pow(x_i))$ equals the mass of its corresponding sample $\mu_1(x_i)$. This can be achieved by designing a loss function such that the optimum satisfies this constraint. The following loss has been proposed [9] :

$$\Phi((x_i), (\varphi_i)) = \sum_{i=1}^n \int_{x \in Lag^\varphi(x_i)} (\varphi_i - \|x - x_i\|^2)\mu_0(x)dx - \sum_{i=1}^n \varphi_i p_i \tag{1.18}$$

[2]From personal experience, however, even relatively accurate approximations such as recursive filters produce significant artifacts ; truncated kernels prevent mass from moving further than the kernel support (as this corresponds to setting $c(x, y) = \infty$ outside) ; FFT-based solutions resulted in ringing artifacts. In practice, I could only benefit from the kernel separability.

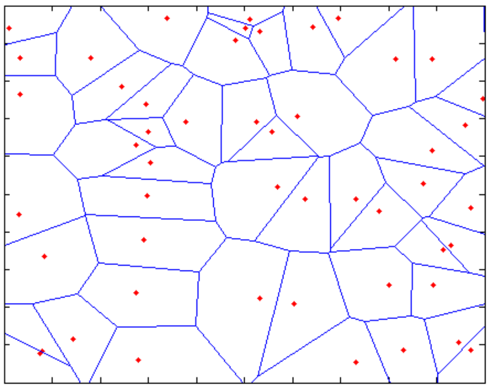


Figure 1.7: Power diagram (blue) of a set of samples (red), associated to random weights.

The construction of Power diagrams in dimension d can be obtained by *lifting* a Voronoi diagram, i.e., by considering a Voronoi diagram in dimension $d+1$ for which the additional dimension is given by $\sqrt{(}M - \varphi_i)$ where M is any sufficiently large constant, and intersecting it with the original d-dimensional hyperplane passing through the origin. This leads to a geometric interpretation of the semi-discrete optimal transport, best illustrated in 1D in Fig. 1.8.

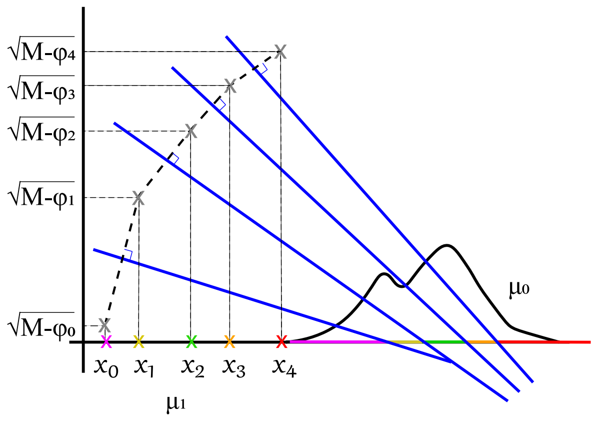


Figure 1.8: Illustration of 1D semi-discrete transportation. The transportation problem between a continuous density μ_0 and Diracs supported on the 5 colored crosses at $\{x_i\}_i$, amounts to finding weights $\{\varphi_i\}_i$. These weights are such that the measure according to μ_0 of each 1D cell (shown as colored line segments) formed by the intersection of the 2D Voronoi Diagram of $\{(x_i, \sqrt{M-\varphi_i})\}_i$ and the line $(x, 0)$ is equal to the corresponding Dirac mass. These 1D intersections define a power diagram.

Loss 1.18 can be minimized using a gradient descent approach [9] since the gradient can be expressed simply by the difference between the mass of the power cell and the corresponding Dirac mass (it can trivially be seen that, at the optimum the gradient is null and the mass preservation criterion is met). A multiscale approach has been proposed and implemented in 2D [119], and a (non-multiscale) version for tetrahedral meshes in 3D has been implemented [107]. Recent methods consider second order optimization routines making use of the Hessian [91].

1.2.3 Sliced Optimal Transport

For completeness, this section describes an approach that does not directly compute an optimal transport map, but a transport map that is optimal per 1D slice. Its benefit is its very high computational speed and ease of implementation, and has been pioneered by Pitié [140] and extended to barycenters by Rabin et al. [145]. We refined the notion of Sliced Wasserstein Barycenter in our work [29], and provided a similar barycenter construction on regular grids via Radon transforms in section 2.2.

Realizing that 1D optimal transport between two discrete measures of uniform weights can be achieved by a mere sort of the atom positions (see subsection 1.1.6), the idea of sliced optimal transport is to project two n-dimensional discrete measures with uniform weights onto a random 1D line of direction ω, sorting the atoms, pairing them, and translating the atoms of the first distribution along ω so that their projections match those of their paired atoms in the second measure, and iteratively repeat the procedure with another random 1D line. This progressively makes the first distribution closer and closer to the second, by iteratively moving its atoms closer to those of the second distribution. The resulting advection is not optimal with respect to the n-dimensional Monge functional, but is optimal with respect to a simpler functional that operates per slice (see section 2.2 – both the Radon and Sliced approaches minimize the same functional). This approach has initially been used for matching the color distribution of one image to that of a second image – a process called color transfer – and is, I believe, still a near state-of-the-art method for color transfer despite it being introduced in 2005.

1.3 Temporal Coherence

Chapter 3 will focus on my second line of work: making image processing algorithms temporally consistent. In fact, most non-trivial image processing algorithms, when applied independently to each frame of a video, produce large flickering artifacts. This section reviews the litterature on temporal consistency of both application specific approaches and more general ones. While other notions of temporal consistency exist (e.g., video stabilization [116], restoring flickering from old footages [138, 48], enforcing temporal regularity of brush strokes or inpainting [146]), we mostly focus on algorithms applicable to the regularization in time of image processing filters assuming a clean input video.

1.3.1 Application specific consistency

Various image processing operations have been extended to work on videos. Notably, as we shall see in section 3.2, numerous attempts investigate the temporal regularization of the intrinsic decomposition problem – the problem of separating an image (or video frame) into reflectance and illumination components – appeared concurrently to our work in 2014 [193, 94, 167], and were later extended to light field images [70] or made real-time [118].

Color processing has also been studied for videos. Chang et al. [38] use anisotropic diffusion to ensure that subtle color transformations are temporally coherent, while Wang et al. use B-splines to interpolate color transforms in time [187]. Farbman and Lischinski [60] stabilize tonal fluctuations in videos by using optical flow estimated at sparse locations to propagate tonal differences across frames. Oskam et al. [127] employ spatio-temporally consistent radial basis functions to color balance sequences for augmented reality.

Other filters made temporally consistent include tone mapping [90, 11], matting [188], colorization [192], painterly rendering [113], or more generally neural style transfer approaches [154].

Each approach introduced in this section would not readily work to tackle other filtering problems. While we will first adress video color grading in section 3.1 and the intrinsic decomposition problem in section 3.2, we aim for an all-purpose strategy that will be exposed in section 3.3.

1.3.2 General-purpose consistency

Regularization methods have been proposed, which amount to low-pass filtering pixel values. In particular, Paris [130] extends the Gaussian kernel to the time domain and uses this result to adapt applications such as bilateral filtering and mean-shift clustering to videos. Lang et al. [102] also extend the notion of smoothing to the time domain by exploiting optical flow and revisit optimization-based techniques like motion estimation and colorization. For other techniques that do not optimize an energy, they resort to temporal low-pass filtering.

Concurrent to our work, Dong et al. [55] present a technique to stabilize video frames processed by an unknown image filter that can be expressed as nonlinear curves applied to regions of the original video frames. In contrast, our technique described in section 3.3 is not restricted to a specific formulation and can handle applications like intrinsic images or depth prediction that violate this assumption.

2

Optimal Transport for Computer Graphics

This chapter details my contributions relating to optimal transport. The following section describes my first solver for optimal transport based on linear programming and radial basis functions. I will then expose a faster albeit approximate solver based on Radon transforms in section 2.2, and finally expose definitions of inverse problems as well as efficient algorithms in section 2.3.

2.1 Lagrangian Displacement Interpolation

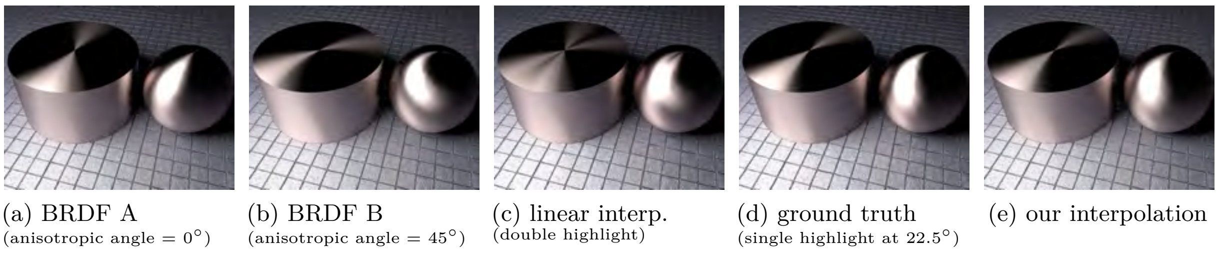


Figure 2.1: Examples of interpolation between two anisotropic sampled BRDFs (a,b). Naive linear interpolation generates an unrealistic double highlight (c). Since we used a parametric BRDF model, we can compute the ground-truth in-between BRDF by interpolating the parameters (d). Our approach has no knowledge of the parametric BRDF representation but is nonetheless able to produce a similar output (e).

After preliminary exposure to optimal transport during my Ph.D [27], I wanted to investigate the use of optimal transport for interpolating functions. I re-discovered the notion of displacement interpolation, and after more readings, proposed a first framework published in 2011 [34], exposed in this section, with applications to computer graphics.

Challenges: First, optimal methods working on regular grids [17, 79, 147] do not scale well in high dimensions. We hence want to cast optimal transport solutions based on min-cost flow on graph [83] to continuous domains, allowing to generate continuous interpolated distributions from a pair of continuous input distributions. Second, in the discrete setting, it is unclear which mass transport solvers are best suited to the task at hand, so we investigate a number of solutions. Third, we wish to extend advection-style interpolation so that it can be applied to arbitrary functions, rather than being restricted to probability distributions. These produce unintuitive

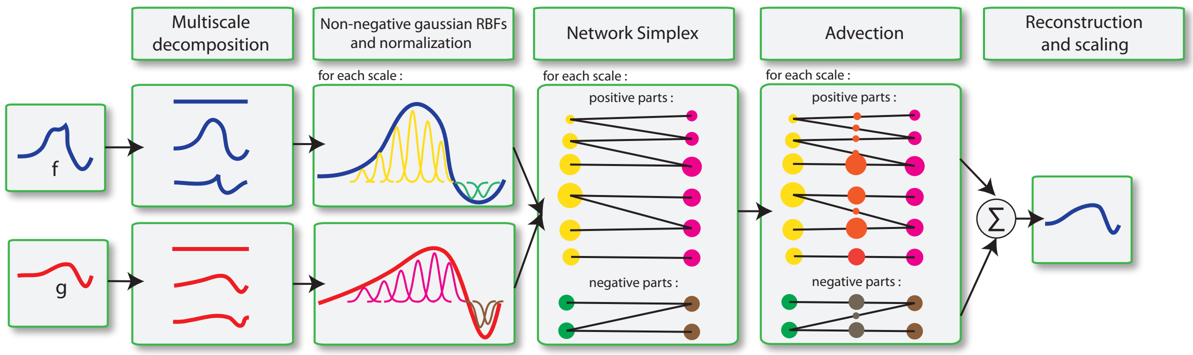


Figure 2.2: Overview of our pipeline.

artifacts if simply treated as a distribution.

2.1.1 Displacement Interpolation of Continuous Distributions

Our interpolation follows a three-step process. First, the source and target distributions are decomposed into a sum of Gaussians. Each Gaussian in the source distribution is then paired to one or more Gaussians of the target distributions by solving a mass transport problem. In the last step, an interpolated distribution is constructed by summing the Gaussians after a partial advection to the target locations. When appropriate, this whole procedure is repeated independently at different frequency scales to achieve a multiscale decomposition and reconstruction. Figure 2.2 summarizes the complete pipeline.

RBF decomposition

We assume that the source and target functions f and g are given as a set of samples at locations x_i and y_j, that is, we know a set of values $f(x_i)$ and $g(y_j)$. We do not require that the x_i and y_j points be the same or that they be regularly spaced. To approximate smooth functions, we associate a smooth Gaussian radial basis function (RBF) kernel with each sample point that will also serve as a particle of mass within the mass transport problem. Each particle is represented by $w\,G_\sigma$, where w is the mass associated with the particle, and G_σ is a normalized Gaussian kernel with variance σ^2 and centered at the sample point.

To represent a non-negative function f as an appropriate sum-of-Gaussians, we seek to approximate $f(x)$ by $\sum_i w_i\, G_{\sigma_i}(d(x_i, x))$, where $d(x_i, x)$ is the geodesic distance between point x and the center of a particle x_i, and w_i and σ_i are the unknowns to be determined. The kernel width σ_i controls the smoothness of particle representations, and is chosen as the distance d_i^N between x_i and its N^{th} nearest neighbor (typically $1 \le N \le 10$). Given σ_i, we can estimate the w_i weights using a non-negative least-squares formulation:

$$\min_{w_i} \sum_k \left[f(x_k) - \textstyle\sum_i w_i\, G_{\sigma_i}(d(x_i, x_k))\right]^2 \quad \text{with } w_i \ge 0 \tag{2.1}$$

For a function with positive and negative values, we decompose it into its positive and negative components, $f^+ = \max(0, f)$ and $f^- = \max(0, -f)$ (the same applies to g), interpolate each component separately, and recombine the result $h = h^+ - h^-$.

We use a dense non-negative least-squares solver based on QR factorization [103] which is more robust and faster than solvers based on sparse matrices and the normal equations [37] due to the poor conditionning of RBF matrices.

Mass transport

The core of the algorithm resides in the mass transport scheme [64]. The goal is to pair each particle of the source distribution f to one or more Gaussian particles of the target distribution g. First, we normalize particles weights w_i and w_j by $s_f = \sum_i w_i$ and $s_g = \sum_j w_j$ respectively, yielding normalized weights, $\bar{w}$. Since the Gaussian kernels G are normalized, the mass of a particle $\bar{w}\,G$ is simply $\bar{w}$. We assumes that the mass of particle is concentrated at its center, which further motivates our use of small kernels (§ 2.1.1), and we directly apply Equation 1.6.

The transport problem is solved using a network simplex method (see subsection 2.1.2). The output of this algorithm is a coupling between the source and target particles. A particle may be associated to several particles in the other distribution. In this case, it is split into as many particles as pairs in which it is involved, each with an associated weight that meets the mass transported requirement along the links and summing up to the mass of the original particle. As a result, we have new source and target particles, denoted by $\hat{\imath}$ and $\hat{\jmath}$. The notation $\hat{\jmath}(\hat{\imath})$ is used for the index of the target particle paired with the source particle $\hat{\imath}$. The new set of particles is such that there are equal numbers of source and target particles, and that the paired particles have the same weight, i.e. $\bar{w}_{\hat{\imath}} = \bar{w}_{\hat{\jmath}(\hat{\imath})}$.

Advecting the particles

To build the interpolated function h corresponding to the parameter $t \in [0, 1]$, we advect the particles to the position $\gamma(t)$ along the geodesic path γ that links their source location to their target destination. We use z_ℓ to denote the new position of each particle. We linearly interpolate the size of each particle: $\sigma_\ell^2 = (1-t)\sigma_{\hat{\imath}}^2 + t\,\sigma_{\hat{\jmath}(\hat{\imath})}^2$ and the total mass of the function $s_h = (1-t)s_f + t\,s_g$. Since the weight is constant in a pair, it remains the same, i.e., $\bar{w}_\ell = \bar{w}_{\hat{\imath}} = \bar{w}_{\hat{\jmath}(\hat{\imath})}$. With these values, we construct the interpolated function h:

$$h(x) = s_h \sum_\ell \bar{w}_\ell\, G_{\sigma_\ell}(d(z_\ell, x)). \tag{2.2}$$

Multiresolution interpolation

The Lagrangian mass transport method described thus far results in intuitive advection-like behavior in many settings. For example, interpolating between two bumps generates a translating bump. However, the intuitive behavior is lost when a constant offset or low frequency is added. In the example shown in Figure 2.3, the displacement interpolation solution (top two rows) does not establish correspondences between the peaks of the two distributions, nor for the valleys. Instead, the solution resembles a linear blending solution because of particles representing the peaks and valleys finding correspondences with particles that represent the constant offset associated with the function.

To remedy this problem, we propose a multi-resolution scheme that interpolates different band-passed versions of the functions separately. The lowest frequency bands are constant functions that are blended linearly. The other frequency bands are interpolated using the displacement interpolation method as describes previously. We typically use 3 frequency bands. The bottom two rows of Figure 2.3 illustrate the resulting more intuitive interpolation. We further discuss this point when describing the individual application (§ 3.3.4).

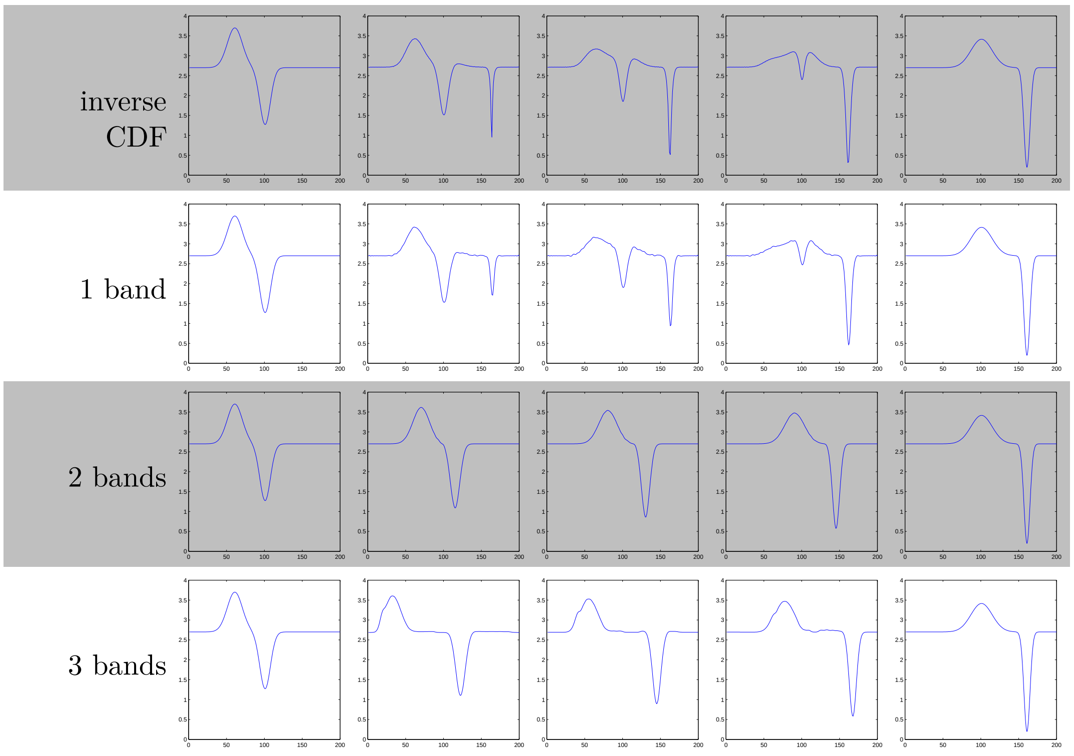


Figure 2.3: Left to right: 1D interpolation of a constant plus a difference of two gaussians. Top to bottom: inverse CDF formula (failure case, due to the constant); displacement interpolation with 1 band (same as inverse CDF); 2 bands; 3 bands. More bands can yield undesirable artifacts in this case.

2.1.2 Mass Transport Solver

Several algorithms exist for solving the mass transport problem on graphs. Here, we investigate methods based on the *transportation simplex* that are dedicated to the computation of Earth Mover's Distance [83] and the *network simplex* used to solve min-cost flow problems. We test the fast transportation simplex implementation of MacDonald's [114], the network simplex implemented in CPLEX [41], and our own code[1] obtained by optimizing the network simplex with *block search pivoting* implemented in the generic graph library LEMON [104].

All timings were re-measured at the time of writing this document (April 2018), with an improved faster implementation.

The transportation simplex has a worst-case exponential complexity but it has been observed to have a polynomial average case complexity under various distributions. The network simplex has a known complexity in $\mathcal{O}(n^3)$ [2]. To determine how these algorithms behave in practice in our context, we generate random mass transport instances of 2D mass transport problems between randomly weighted diracs. Figure 2.4 plots the resulting performance. The tests reproduce the $\mathcal{O}(n^3)$ complexity of the transportation simplex but also reveal that the network simplex behaves in $\mathcal{O}(n^2)$ in our context, which is a major gain at the scale at which we typically work,

[1]available at `https://github.com/nbonneel/network_simplex/` , with further comparisons

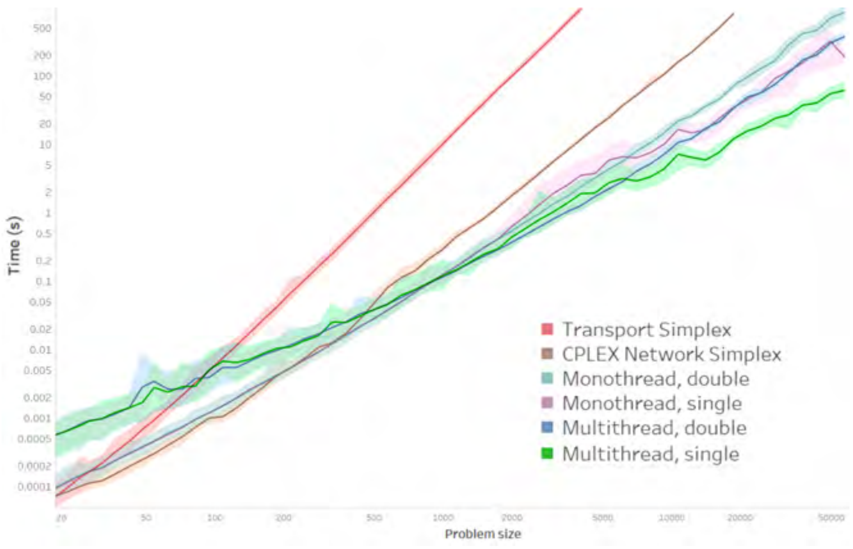


Figure 2.4: As of April 2018, speed comparison between our network simplex (in single and double precision, mono-threaded and multi-threaded), the network simplex implemented in CPLEX [41] and the transport simplex implementation of MacDonald's [114]. Even in double precision, we perform up to 25x faster than CPLEX, and solve large problems of more than 55000 particles in about 7 minutes. Our network simplex behaves as a $\mathcal{O}(n^2)$ algorithm in practice whereas the transport simplex runs in $\mathcal{O}(n^3)$.

i.e. thousands of particles. Our experiments presented in the paper [34] also show that fixed-point precision further speeds up computations, which we used to generate the results presented hereafter.

2.1.3 Results

In this section, we discuss specific applications and their associated details such as the choice of ground distance. We first present applications that handle continuous data, BRDFs and value functions of animation controllers. We then apply our method to discrete problems such as stipple rendering. Further discussions and results on environment map interpolation are available within our paper [34].

BRDF interpolation

We demonstrate our method for interpolating Bidirectional Reflectance Distribution Functions (BRDFs) that describe the appearance of materials. We use cosine-weighted BRDFs to ensure proper energy conservation, interpolate for each wavelength independently, and work in the log domain using a $\log(1 + x)$ transformation for perceptually more meaningful results [155]. This concave remapping ensures that our result does not break the energy preservation rule. Since energy preservation applies to the 2D slices representing the outgoing directions associated to a given incoming direction, we perform interpolation slice by slice. Reciprocity is not guaranteed in this process, but could be enforced in a postprocessing step. We use the squared geodesic distance on the sphere as the ground distance and use spherical linear interpolation for particles.

Interpolation results on a (sampled) parametric Ashikhmin-Shirley anisotropic BRDF model and on measured BRDFs, with comparisons to naive linear interpolation are shown respectively

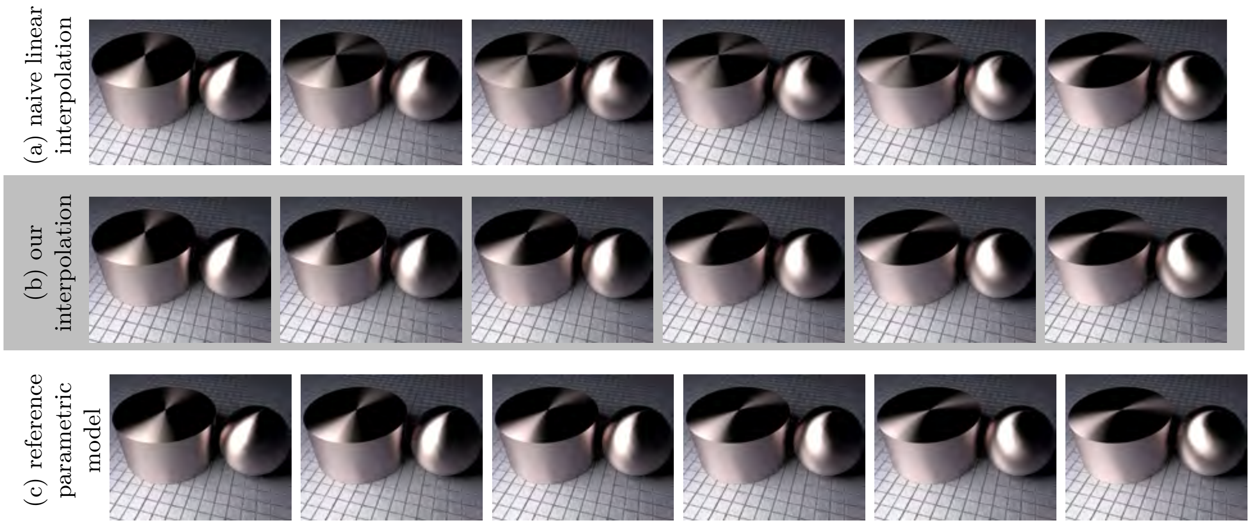
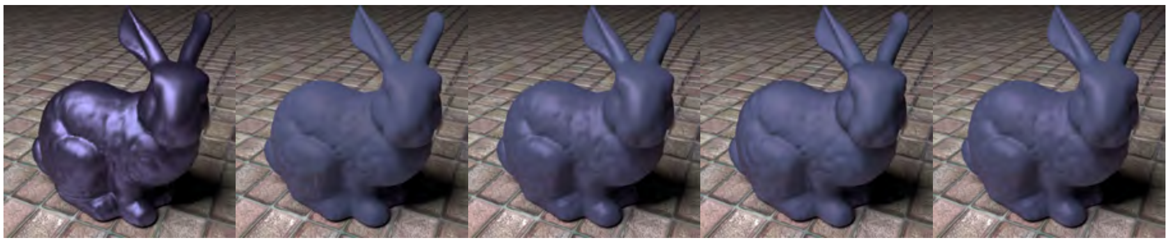

(a) naive linear interpolation
(b) our interpolation
(c) reference parametric model

Figure 2.5: Validation on an anisotropic Ashikhmin-Shirley synthetic BRDF. Naive interpolation (a) and our method (b) do not know about the underlying parametric model and are given only a sampled representation of the BRDFs. Naive linear interpolation (a) cross-fades the two BRDFs, which produces unrealistic double highlights. In comparison, our approach (b) rotates the highlights in a realistic way similar to interpolating the parameters of the model (c).

in Figure 2.1 and Figure 2.5.



(a) BRDF A (b) BRDF B (c) Linear interp. (d) Disp. interp. 1 band (e) Disp. interp. 3 bands

Figure 2.6: 80% interpolation between two measured BRDFs. Even at 80%, the linear interpolation remains specular, while the 1 band displacement interpolation increased the roughness of the surface: linear blending cannot control the roughness of the underlying microgeometry, as could be desired. The target BRDF does not contain high frequencies to be matched: the multi-scale approach does not perform much better than linear blending in this case.

Value function interpolation

In character animation and reinforcement learning, value functions are a convenient way to compute and store optimal control policies. A value function stores the optimal discounted cost-to-go across a state space that is defined by the state of the character with respect to a goal. Meaningful interpolation interpolation of value functions would allow for existing control policies to be reused for new sitations. In the example we shall demonstrate, this takes the form of an obstacle whose position is not known until run-time. We wish to be able to interpolate between two reference solutions for the value function that are computed for two particular locations of the obstacle.

We perform interpolation experiments where the value function is computed on the 4D state space of an oriented particle. The 4D state space represents the particle position, angular velocity and sagital acceleration. Particle trajectories can be seen in Fig.2.7. Initial curls at the start of the trajectories are due to the particle initial velocities and their initial angular velocities. As shown in Figures 2.7, the value function computed using displacement interpolation does significantly better at preserving the desired intent than the linearly interpolated value function.

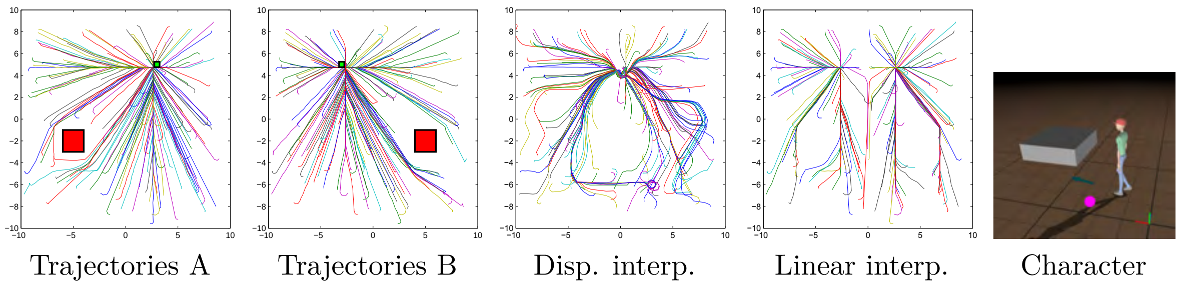


Figure 2.7: Interpolation between two 4D value functions. The red square is an obstacle and the green square is a target. From left to right: particles trajectories for reward A; for reward B; using our interpolated value function ($\alpha = 0.5$); using a linear reward interpolation as in [45]. The particle can be used to drive a 3D physically-based character. Although our method does not result in optimal trajectories, most of the particles achieved the expected interpolated goal location while avoiding the interpolated obstacle. Linear Bellman combination [45] results in an optimal behavior, but yields two different targets and obstacles.

Color distribution interpolation

We apply our method to transfer and interpolate color histograms. Given a source and target image, we use clustering and solve the mass transport on the resulting discrete problem. The pairing between source and target particles define a remapping of the source colors onto the target colors, or any distribution interpolated using this pairing.

In this particular context, our color transfer algorithm is similar to the method of Morovic and Sun [123]. However, our fast network simplex solver allows for a much finer quantization of colors, so we can use 16 000 particles and remove the artifacts stemming from the coarse sampling imposed by the transport simplex (Fig. 2.8). We also compared our approach to the method by Pitié et al. [140].

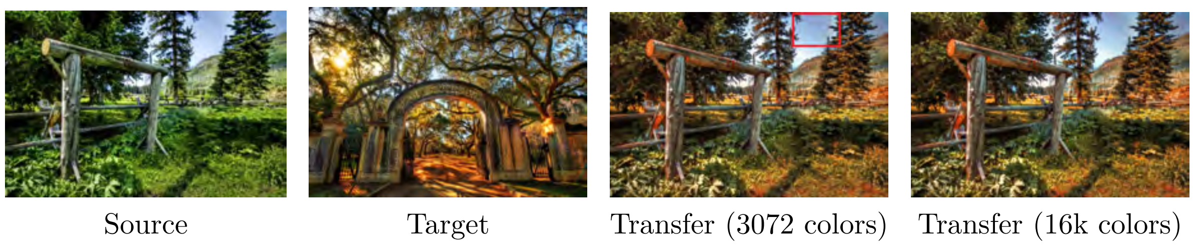


Figure 2.8: Our method can be used for histogram transfer as in [123]. The use of an efficient network simplex allows the use of 16000 colors while the transportation simplex used in [123] only achieves up to 3072 colors (artifacts due to color reduction are highlighted).

Animated Stipples

Our fast network simplex can be used to animate stipples in the context of non-photorealistic rendering (NPR). We use the method of Secord et al. [164] to generate particles on the source and target images, as a set of points or strokes. We set the method such that the number of source and target particles is the same. We solve the transport problem by assigning a unit mass to each particle. In this case, mass transport reduces to an assignment problem and its solution does not require splitting or merging particles [83]. For the ground distance, we use the standard L^2 distance that generates straight lines as geodesics, and the optimal transport solution guarantees paths are not crossing. For strokes, their orientations are linearly interpolated between input and target orientations. The results are shown in Figure 2.9.

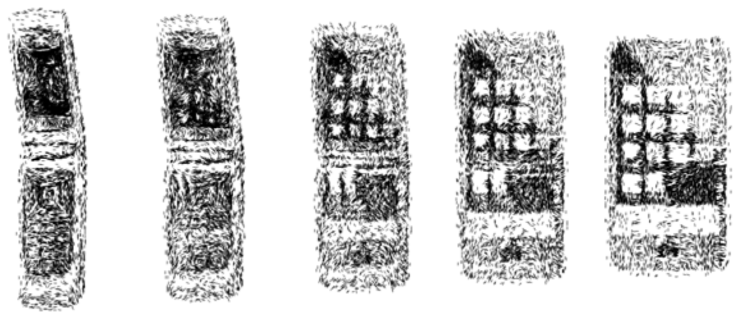


Figure 2.9: Stipple interpolation using displacement interpolation.

2.1.4 Discussion

We have presented a practical Lagrangian method for displacement interpolation of continuous distributions, by advecting frequency bands independently. We show that a network simplex algorithm is in many cases preferable to the transportation simplex algorithm, and we have demonstrated the utility of displacement interpolation in multiple applications.

The current implementation of the method cannot achieve real time interpolation for histograms larger than a few hundred bins. While simplex computations have been performed on the GPU [22], their performance is still much lower than efficient CPU implementations. Sparse QR approaches may also result in faster computation. Note that while the preprocessing can be slow, the interpolation itself is fast, and is trivially parallelized as it merely consists in computing a sum of Gaussian for each sample. In the context of Value Function interpolation, we also explored the displacement interpolation as a way to *extrapolate* (see paper [34]). We believe this is a promising direction for further investigations.

2.2 Radon Wasserstein Barycenters

As seen in subsection 1.1.6, it is computationally inexpensive to compute the Wasserstein barycenter of 1-D densities. It thus makes sense to seek for alternate definitions of barycenters of measures in $\mathbb{R}^d$ that rely on 1-D Wasserstein distances and barycenters.

This section briefly sketches the notion of Radon Barycenters as described in our paper [29]. The sliced barycenter, also proposed in this paper, is the work of my co-authors. More details, properties and connections between Sliced and Radon Wasserstein barycenters can be found in the paper.

2.2.1 Radon Transform

The classical Radon transform of functions (see [82]) is commonly used for tomographic imaging. The Radon transform of a function essentially projects the function onto lines, by integrating the function along the direction orthogonal to these lines (see Figure 2.10). This process is, up to an exponential decay, that of the absorption of an X-ray beam (or electrons, sound waves...) by a body – the problem being of recovering a 3D reconstruction from many such projections through an *inverse Radon transform*.

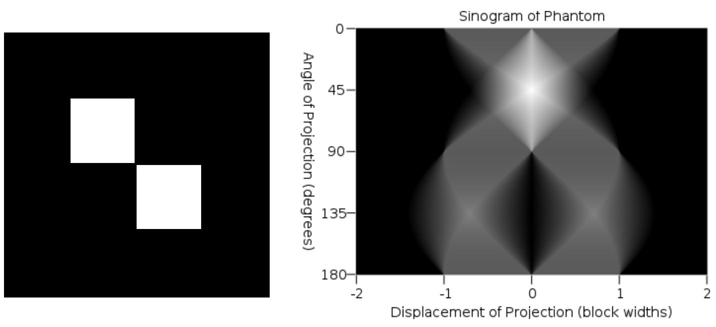


Figure 2.10: A 2-D indicator function of two squares (left) and its Radon transform (right). *Wikipedia*

The Radon transform is naturally extended to measures by duality. The Radon transform of a measure μ, denoted $R(\mu)$, gathers projections of the input measure along all possible directions. Intuitively, the Radon transform of a measure μ is a decomposition of μ into a set of measures $R(\mu)^\theta$, and each measure $R(\mu)^\theta$ is a conditional measure which projects μ on the line of direction θ.

We also define the inverse Radon transform of measures by duality using the operator $R^{+,*} \overset{\text{def.}}{=} R(R^*R)^{-1}$, where R^* denotes the adjoint of R (see paper [29]).

2.2.2 Radon Barycenter

The injectivity of the Radon transform of a measure allows us to define the Radon barycenter. Denoting $\mathrm{Bar}^R_{\mathbb{R}^d}(\mu_i, \lambda_i)_{i\in I}$ the Radon barycenter of measures $(\mu_i)_{i\in I}$ indexed by some index set I with weights $(\lambda_i)_{i\in I}$, and similarly, $\mathrm{Bar}^W_{\Omega^d}(\nu_i, \lambda_i)$ the 1-d Wasserstein barycenter applied to each slice θ of $\Omega^d \overset{\text{def.}}{=} \mathbb{R} \times \mathbb{S}^{d-1}$:

Definition 1 (Radon barycenter)**.** *Given weights λ_I and probability measures $(\mu_i)_{i\in I} \in \mathcal{M}_1^+(\mathbb{R}^d)^I$, we define*

$$\mathrm{Bar}^R_{\mathbb{R}^d}(\mu_i, \lambda_i)_{i\in I} \overset{\text{def.}}{=} R^+ \mathrm{Bar}^W_{\Omega^d}(R(\mu_i), \lambda_i)_{i\in I} \in \mathcal{D}^*(\mathbb{R}^d).$$

Since for $\nu \in \mathrm{Bar}^W_{\Omega^d}(R(\mu_i), \lambda_i)_{i\in I}$ one does not have in general $\nu \in \mathrm{Im}(R)$, $\mathrm{Bar}^R_{\mathbb{R}^d}(\mu_i, \lambda_i)_{i\in I}$ is composed of distributions and not necessarily measures.

It can be shown (see Appendix C of our paper) that the Radon Wasserstein barycenter minimizes an energy:

Proposition 1. *Denoting*

$$\mathcal{E}(\nu) \overset{\text{def.}}{=} \sum_{i\in I} \lambda_i \mathrm{W}_{\Omega^d}(R\mu_i, \nu)^2, \tag{2.3}$$

with

$$\mathrm{W}_{\Omega^d}(\nu_1, \nu_2)^2 \overset{\text{def.}}{=} \int_{\mathbb{S}^{d-1}} \mathrm{W}_{\mathbb{R}}(\nu_1^\theta, \nu_2^\theta)^2 \mathrm{d}\theta,$$

$\mathrm{W}_{\mathbb{R}}(\nu_1^\theta, \nu_2^\theta)$ *the Wasserstein distance with quadratic cost between slices of ν_1 and ν_2 along a direction $\theta \in \mathbb{S}^{d-1}$, and denoting $\bar{\mathcal{M}}_1$ the space of measures for which each slice is a (unit) probability measure, one has*

$$\mathrm{Bar}^R_{\mathbb{R}^d}(\mu_i, \lambda_i)_{i\in I} = R^+ \underset{\bar{\mathcal{M}}_1^+(\Omega^d)}{\mathrm{argmin}} \ \mathcal{E}. \tag{2.4}$$

$$\tag{2.5}$$

Our paper shows that this Wasserstein Radon Barycenter enjoys a number of properties shared with classical Wasserstein Barycenters [29]. In particular, it is invariant with respect to translation, scaling, rotation and symmetry, and the Wasserstein Radon Barycenter of translated and scaled copies of a given measure is also a translated and scaled copy.

2.2.3 Approximate Computation with Eulerian Discretization

Discretizing the input measures on a grid, our procedure relies on a discrete Radon transform and a discretization of the 1-D Wasserstein barycenter of Equation 1.12. The 1-D Wasserstein barycenter of Equation 1.12 is approximated by approximating cumulative distribution with sums and interpolation, as well as a finite difference derivation.

Marginals are obtained via a discrete Radon Transform. We investigate the use of the Fast Slant Stack Radon transform [10]. It has the property to faithfully approximate the geometry of the Radon transform, i.e., it exactly computes integrals over 1-D rays for band limited functions. This Fast Slant Stack implements both the computation of the Radon transform and its adjoint with fast algorithms. The inverse Radon transform is obtained using the Moore-Penrose pseudo-inverse of R, computed via a conjugate gradient descent.

The approximated Radon barycenter of the discretized measures $(\mu_i)_i$ is finally obtained by first computing the Radon transforms of all $(\mu_i)_i$ resulting in a set of sliced measures, computing the 1-D discretized Wasserstein barycenter of each slice, and finally performing an inverse Radon transform.

A typical Radon barycenter of three two-dimensional pdfs discretized on a 1024×1024 pixel grid, and the principled Fast Slant Stack Radon transform with 2048 slices, requires 11 seconds to precompute the initial Radon transforms, and 170 seconds to compute 32 Radon barycenters, with unoptimized parallel Matlab code. It is possible to accelerate this timing using less precise Radon transform. For instance, using Matlab's implementation of the Radon transform with 180 slices requires 14 seconds to compute these 32 barycenters on a single core. In comparison with the Eulerian proximal splitting method of Papadakis et al. [129], the Wasserstein barycenter between two 1024×1024 distributions with 32 time steps and $100,000$ iterations to achieve an acceptable convergence requires on average 72 hours, using an optimized C++ vectorized and parallel implementation (see Figure 2.11 for a display of the resulting barycenters). Comparing with the entropy regularized approach of Cuturi and Doucet [44], the result obtained with their method is produced in two hours on a GPU, using a 150×150 sampling grid (see Figure 2.12). Note however that the more recent convolutional method of Solomon et al. [174] would be much faster for the same result. In contrast, our Radon barycenter computed on a grid of 400×400 pixels (which is zero-padded to 1200×1200 pixels to avoid Radon transform artifacts) is obtained in 40 seconds using the fast slant stack approach with 1200 directions, and 2 seconds with Matlab built-in Radon transforms with 180 directions, on a single core of a laptop.

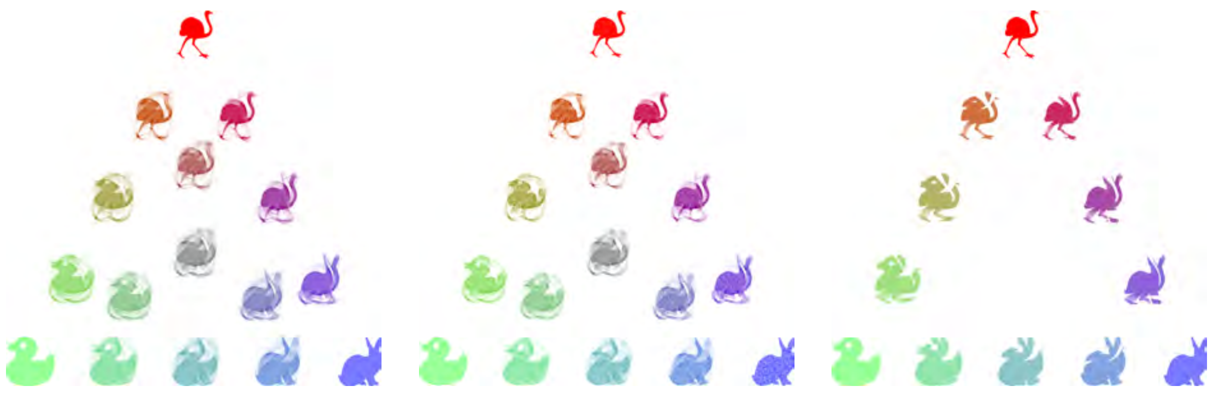


Figure 2.11: Comparison of the Radon, Sliced (see [29]) and Wasserstein barycenter computed using the method detailed in [129].

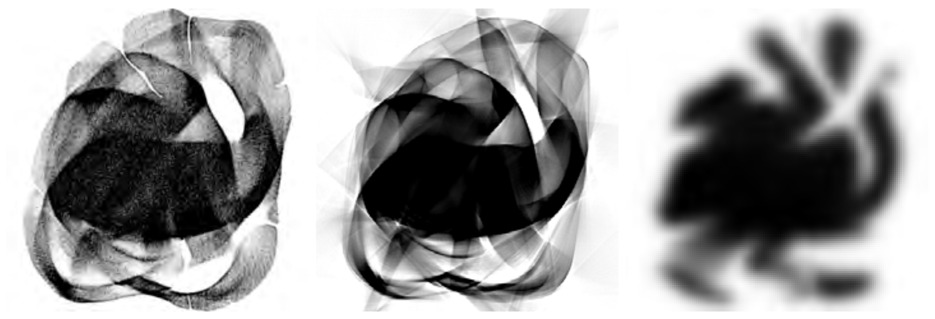

$\mathrm{Bar}^S_{\mathbb{R}^d}$
$\mathrm{Bar}^R_{\mathbb{R}^d}$
Cuturi et al.

Figure 2.12: Comparison of three methods (Sliced [29], Radon, and entropy-regularized [44]) to compute isobarycenters (i.e. using weights $(1,1,1)/3$) of the three input densities displayed at the vertices of Figure 2.11, bottom.

2.2.4 Application to Texture Mixing

To illustrate the usefulness of the Radon barycenter, we apply it to the problem of texture mixing. Texture synthesis seeks to produce textures that look “random” albeit following a texture examplar image. Texture mixing seeks to produce a similarly random texture, but *in-between* a number of texture examplars. The Radon barycenter is well suited to this application which requires an Eulerian discretization in order to interpolate power-spectra computed on the uniform grid of Fourier frequencies.

Spot-noise (SN) texture model. Following the work of [67], we consider stationary Gaussian random vectors F which take values in $\mathbb{R}^N$. These vectors are indexed on the image grid

$$F = (F_k)_{k \in \mathcal{G}} \quad \text{where} \quad \mathcal{G} = \{-n/2+1, \dots, n/2\}^2,$$

(for simplicity we assume that n is even) and we use periodic boundary conditions. Without loss of generality, we assume that they have zero mean $\mathbb{E}(F) = 0$. Such a random vector is thus entirely characterized by its (square root) power spectrum density (PSD)

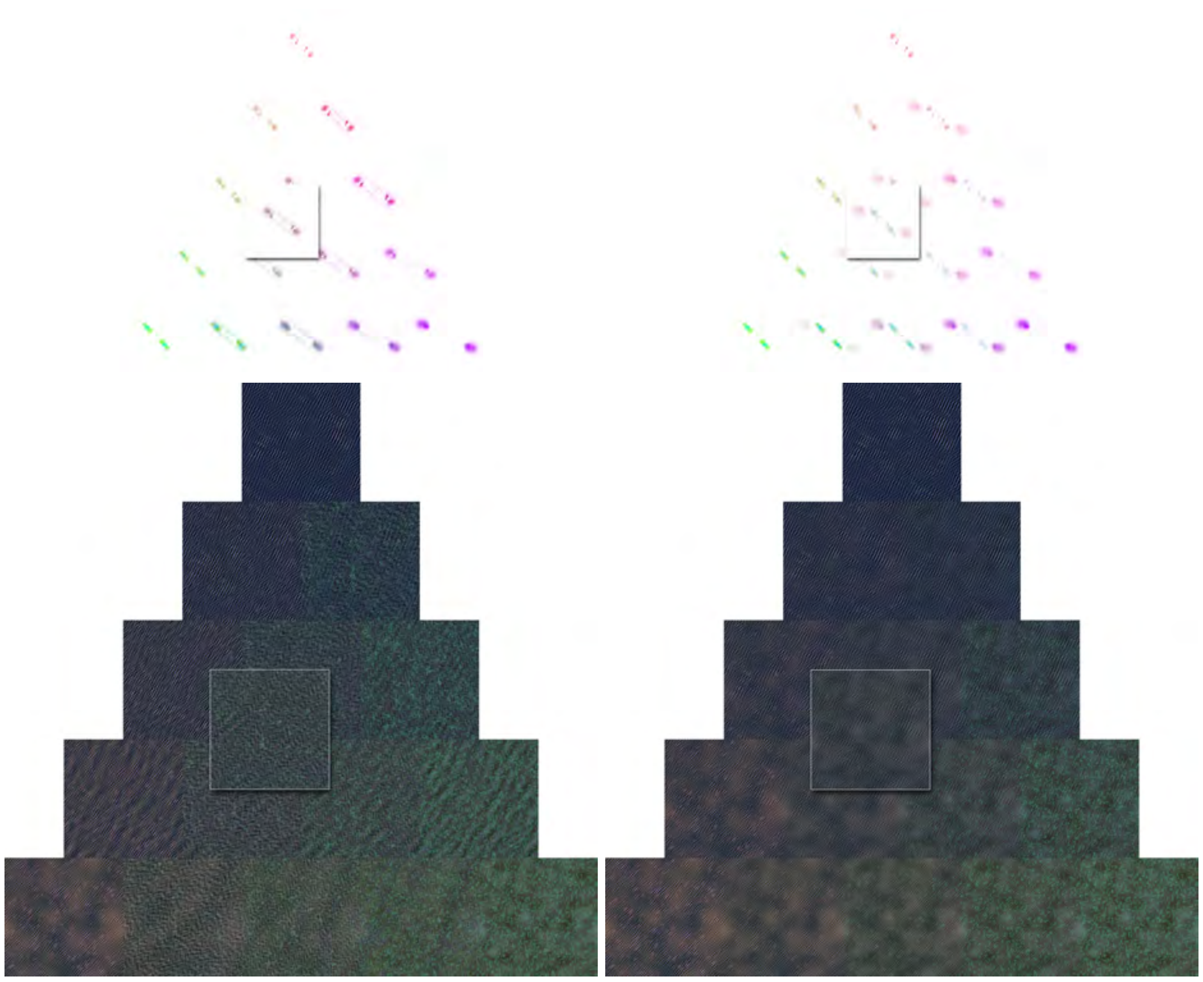
$$\forall\, \omega \in \mathcal{G}, \quad P_F(\omega) = \mathbb{E}(|\hat{F}(\omega)|^2)^{1/2}$$

with $\hat{F}$ the Fourier transform of F.

It is easy to draw a realization f of the vector F by convolving the inverse Fourier transform of P_F (the so-called texton, see [50]) by a realization w of a white noise $W \sim \mathcal{N}(0, \mathrm{Id}_N)$. We estimate the covariance of input Gaussian $\{F^{[i]}\}_{i \in I}$ by using the empirical periodogram (see [67, 61]):

$$\forall\, i \in I, \quad \forall\, \omega \in \mathcal{G}, \quad P_{F^{[i]}}(\omega) = |\hat{f}^{[i]}(\omega)|.$$

Examples. We demonstrate our Radon barycenter of power spectrum densities on several examples. A sparse hand-designed power spectrum is interpolated in Fig. 2.13 and a more natural, less sparse, power spectrum is used in Fig. 2.14. We handle colors by convolving the interpolated power spectrum of each color channel by the same white noise. Although the decoupling of color channels could occasionally lead to color artifacts, we did not observe such effects on our set of examples (further examples can be see in the additional material of the main paper). We hence leave the investigation of perceptually decoupled color spaces or the joint transportation of color channels for future work.



(a) Radon barycenter (our approach) (b) Linear interpolation [61]

Figure 2.13: (a) Eulerian Radon barycenter interpolates sparse amplitude spectra. (b) linear interpolation of the amplitude spectrum, as performed in [61]. The top row shows the interpolated spectra P_{F_λ}.

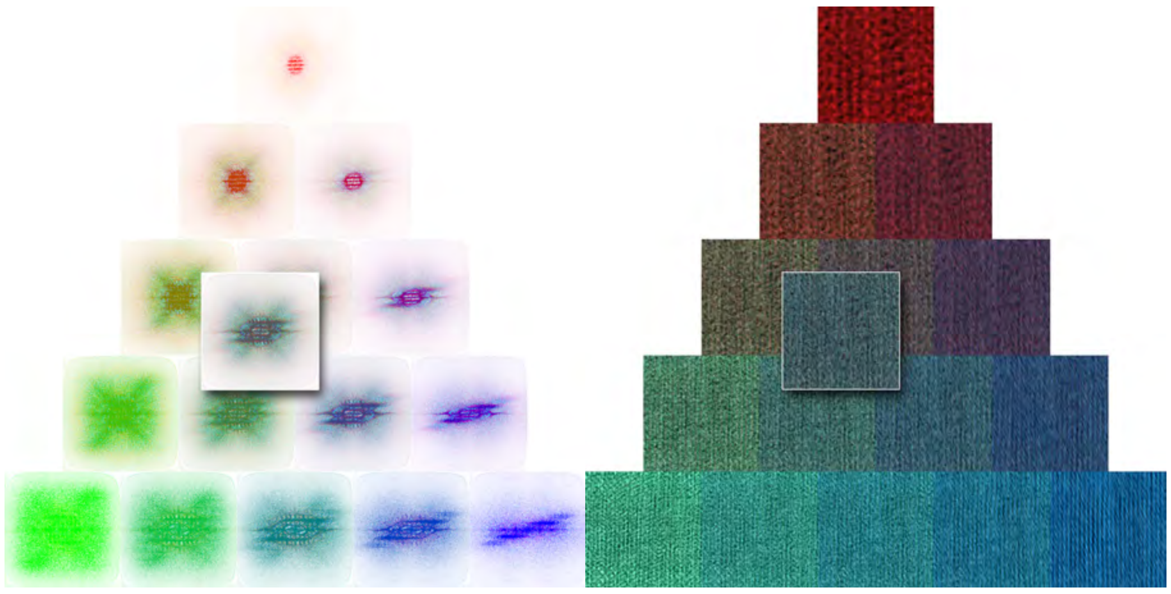


Figure 2.14: Eulerian Radon barycenter applied to the mixing of natural textures.

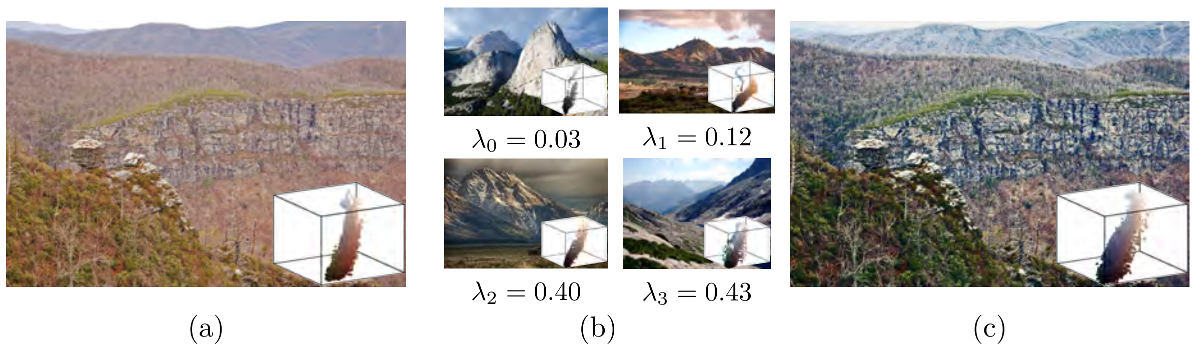


Figure 2.15: Illustration of Wasserstein Barycentric Coordinates. Our Wasserstein projection framework can be used to automatically color grade an input photo (a) using a database of stylized color histograms, with samples shown in (b). We propose to compute the optimal transport barycenter of these stylized palettes that can approximate best the original palette, and use that barycenter to carry out color transfer without large color distortions as shown in (c), where the modified image and the barycentric palette are represented. That barycentric palette is parameterized using only the weights appearing in the captions of figure (b). Other applications include inferring reflectance functions or missing geometry (see Sec. 2.3.5).

2.3 Wasserstein Barycentric Coordinates and Dictionary Learning

We consider in this section two inverse problems associated with histogram interpolation:

- that of forming, for a given histogram, a Wasserstein barycenter of reference histograms that approximates it best. The barycenter itself can be interpreted as a denoised version of the original input, with respect to the prior contained in those reference histograms. The usually much shorter vector of barycentric coordinates can serve as a handy representation to compress, visualize or carry out inference on the original histogram. The crucial novelty lies in the fact that the interpolation we consider here is in the *optimal transport metric sense*, which gives our barycentric coordinate system an intuitive and geometrically faithful flavor. We call this new notion of coordinates for histograms *Wasserstein barycentric coordinates* and is akin to a projection in the Wasserstein space (Fig. 2.15).

- that of approximating a large set of histograms (the data points) by a handful of representative histograms. This problem amounts to a non-linear dictionary learning: non-linearity comes from the fact that we replace the usual linear combination of dictionary atoms by Wasserstein barycenters. Our goal is to reconstruct data points using the closest Wasserstein barycenter to that point using the dictionary atoms according to an arbitrary fitting loss. We propose to learn simultaneously atoms and barycentric weights. We call this notion *Wasserstein Dictionary Learning.*

We provide algorithms to compute Wasserstein barycentric coordinates and Wasserstein Dictionary Learning efficiently. We apply our algorithms on histograms frequently encountered in computer graphics, ranging from color histograms to reflectance distributions as well as image processing applications.

Contributions. We first propose a method to project an input histogram q onto the set of all Wasserstein barycenters formed by S given histograms $(p_1, \cdots, p_S)$ (see Fig. 2.16). This corre-

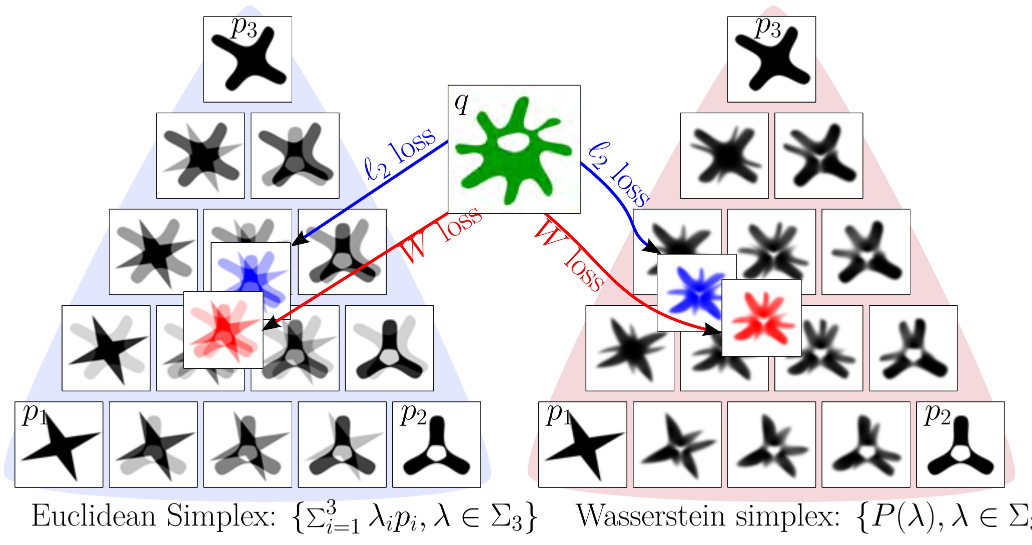

p_3
q
ℓ_2 loss
W loss
ℓ_2 loss
W loss
p_1
p_2
p_3
p_1
p_2
Euclidean Simplex: $\{\Sigma_{i=1}^3 \lambda_i p_i, \lambda \in \Sigma_3\}$
Wasserstein simplex: $\{P(\lambda), \lambda \in \Sigma_3\}$

Figure 2.16: Illustration of Wasserstein Barycentric Coordinates. We consider four monochrome 500×500 images, (q, p_1, p_2, p_3) whose total intensity is normalized to sum to 1. The three images $(p_i)_i$ generate their Euclidean simplex (left, blue background), which consists in all of their convex combinations. The $(p_i)_i$ also define their Wasserstein simplex (right, red background), which consists in all of their Wasserstein barycenters under varying weights λ. *(left arrows)* Finding the best approximation of q on the Euclidean simplex with a ℓ_2 loss is a simple constrained linear regression problem. Finding such an approximation with a Wasserstein loss was recently studied in [151]. *(right arrows)* projections of q onto the Wasserstein simplex of the histograms $(p_i)_i$ using either the ℓ_2 or the Wasserstein loss. This work proposes the first known algorithms to carry out such projections, which can be entirely parameterized by weight vectors λ. These coordinates (3 numbers here) are reflected in the projections' locations in their respective simplexes.

sponds to approximating the input histogram q by its closest (with respect to some loss) Wasserstein barycenter $P(\lambda)$ of $(p_1, \cdots, p_S)$, where $\lambda = (\lambda_1, \ldots, \lambda_S)$ is the optimal weight vector sought for. We call this weight vector λ the Wasserstein barycentric coordinates of q. We propose the first numerical scheme to compute Wasserstein barycentric coordinates. This scheme builds upon gradient descent, and thus requires the computation of the (usually high-dimensional) Jacobian of the barycenter operator $\lambda \mapsto P(\lambda)$. To be tractable, our solution relies on an approximation of $P(\lambda)$ that uses a fixed number of steps of a fixed-point iteration computation proposed by Benamou et al. [18]. We can therefore use a recursive differentiation method to compute that Jacobian efficiently. This leads to an algorithm which is both fast and stable, allowing for the computation of optimal barycentric weights on large scale dense 3-D grids and other domains. We showcase a set of typical applications of our methods to color analysis (Fig. 2.18, fitting sparse reflectance measurements (Fig. 2.20) and reconstructing 3D shapes (Fig. 2.22).

We then extend the Wasserstein barycentric coordinate algorithm to carry out non-linear dictionary learning. Suppose that M input datapoints (in our case, histograms) of interest can be stored in a matrix $X = (x_1, \ldots, x_M) \in \mathbb{R}^{N \times M}$. The aim of (linear) dictionary learning is to factorize the data matrix X using two matrices: a dictionary, D, whose elements (the atoms) have the same dimension N as those of X, and a list of codes Λ used to relate the two: $X \approx D\Lambda$. However, contrary to existing dictionary learning approaches, including those using the Wasserstein distance as the fidelity criterion (see subsection 1.1.8), our method makes full use of the Wasserstein space: instead of considering linear reconstructions for $X \approx D\Lambda$, our aim is to approximate columns of $X \approx \mathbf{P}(D, \Lambda)$ using the Wasserstein barycenter operator $\mathbf{P}$ with weights Λ (see Figure 2.17). We expand the Wasserstein barycentric coordinate approach by also deriving a gradient for the atoms themselves, and performing updates by gradient descent (or, in our case, a quasi-Newton approach) to both atoms and barycentric coordinates simultaneously. We also

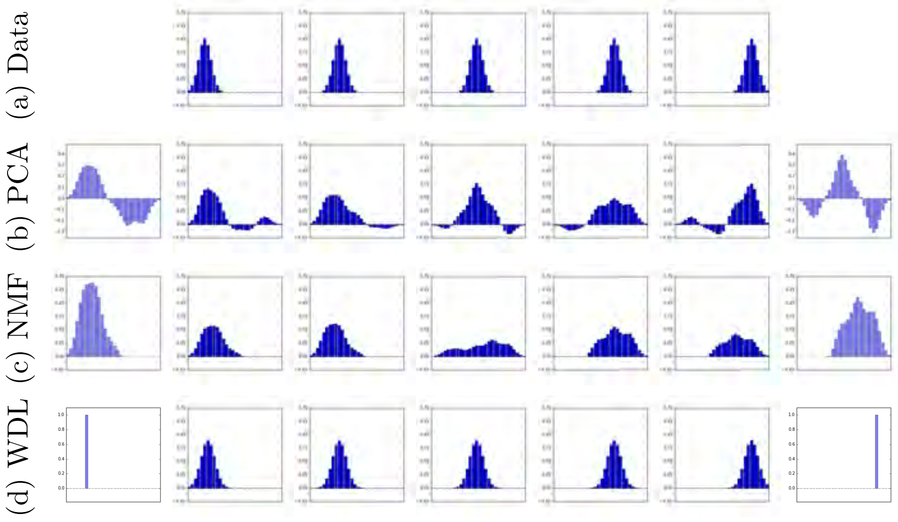


Figure 2.17: Illustration of Wasserstein Dictionary Learning. Top row: data points. Bottom three rows: On the far sides, in purple, are the two atoms learned by PCA, NMF and our method (WDL), respectively. In between the two atoms are the reconstructions of the five datapoints for each method. When the parameter associated with the entropy is high, our method yields atoms that are sharper than the dataset on which it was trained, as is observed here where the atoms are Dirac despite the dataset consisting of discretized Gaussians. See section 2.3.4 for a method to reach arbitrarily low values of the entropy parameter and counteract the blurring effect.

offer some variants to our method, such as a separable log-domain stabilization, a warm-start and a heavy-ball strategy, and accounting for unbalanced optimal transport.

2.3.1 Notations

We consider the simplex $\Sigma_N \overset{\text{def.}}{=} \{p \in \mathbb{R}_+^N \,;\, \sum_i p_i = 1\}$ of N-dimensional normalized histograms, and consider a family of S reference histograms $(p_1, \cdots, p_S)$ in Σ_N. To interpolate between these S histograms, we consider barycentric weights $\lambda \in \Sigma_S$. We write $\mathbb{1}$ for the vector with unit coordinates and whose size depends on the context. The ℓ_α norm for $\alpha \geq 1$ is $\|p\|_\alpha^\alpha \overset{\text{def.}}{=} \sum_i p_i^\alpha$. The Kullback-Leibler divergence between histograms is $\mathrm{KL}(p|q) \overset{\text{def.}}{=} \sum_i p_i \log(p_i/q_i)$. Here, multiplication ($\prod$ for products of many terms and $\odot$ for two terms) and division / operators between vectors are applied entry-wise, as well as exponential exp and logarithmic log maps.

We rewrite the fixed-point algorithm 2 in the following way:

Proposition 2. *[18, Prop. 2] Define for all $s \leq S$, $a_s^{(0)} = \mathbb{1}$, and then recursively for $l \geq 0, s \leq S$:*

$$P^{(\ell)}(\lambda) \overset{\text{def.}}{=} \prod_s \left(K^\top a_s^{(\ell)}\right)^{\lambda_s} \quad \textit{and} \quad \begin{cases} b_s^{(\ell+1)} \overset{\text{def.}}{=} \frac{P^{(\ell)}(\lambda)}{K^\top a_s^{(\ell)}}, \\ a_s^{(\ell+1)} \overset{\text{def.}}{=} \frac{p_s}{K b_s^{(\ell+1)}}. \end{cases} \tag{2.6}$$

where $K \overset{\text{def.}}{=} e^{-C/\gamma}$ is the $N \times N$ kernel matrix corresponding to the cost C and regularization γ. Then $P^{(\ell)}(\lambda) \xrightarrow[\ell \to \infty]{} P(\lambda)$.

2.3.2 Wasserstein Barycentric Coordinate

Overview

Given a histogram $q \in \Sigma_N$, our goal is to define and compute the barycentric coordinates of q within a family of S fixed reference histograms $(p_s)_s$, namely to find the vector of probability weights $\lambda \in \Sigma_S$ such that $q \approx P(\lambda)$ with respect to a loss function $\mathcal{L} : \Sigma_N \times \Sigma_N \to \mathbb{R}_+$:

Definition 2. Let $q, p_1, \ldots, p_S \in \Sigma_N$. The barycentric coordinates of q with respect to $(p_s)_s$ are any optimal solution to problem

$$\underset{\lambda \in \Sigma_S}{\operatorname{argmin}}\ \mathcal{E}(\lambda), \quad \text{where } \mathcal{E}(\lambda) \overset{\text{def.}}{=} \mathcal{L}(P(\lambda), q). \tag{2.7}$$

In contrast to the convexity of problem (1.17), the energy of problem (2.7) is in general not convex. Our goal is thus to recover a stationary point of that energy through gradient descent. The gradient of $\mathcal{E}$ with respect to λ can be computed using the chain rule:

$$\nabla \mathcal{E}(\lambda) = [\partial P(\lambda)]^\top \nabla \mathcal{L}(P(\lambda), q), \tag{2.8}$$

where $\partial P(\lambda)$ is the Jacobian of $\lambda \mapsto P(\lambda)$, $\nabla \mathcal{L}(p, q)$ is the gradient of the loss $p \mapsto \mathcal{L}(p, q)$, and, with these notations, $\nabla \mathcal{L}(P(\lambda), q)$ is that gradient evaluated at $P(\lambda)$.

Among the two quantities in Eq. (2.8), the gradient of the loss $\nabla \mathcal{L}(P(\lambda), q)$ is the least problematic since it can be easily derived for several common losses as shown below, and evaluated at $P(\lambda)$. Applying the transpose of the Jacobian $[\partial P(\lambda)]^\top$ to that gradient is more challenging, both in theory and practice: We showed that, although an exact expression for that Jacobian can be obtained by differentiating the fixed point of Sinkhorn iterates, computing it is impractical for large dimensions N (see full paper [28]). We hence present an efficient alternative, by replacing the true barycenter $P(\lambda)$ in the definition of the energy $\mathcal{E}$ by the running estimate $P^{(L)}(\lambda)$ obtained after L iterations of the map described in Eq. (2.6), where L is a number of iterations fixed beforehand.

Gradient of the loss

The gradient with respect to p of commonly used separable losses $\mathcal{L}(p, q)$ is

$$\nabla \frac{1}{2}\|p - q\|_2^2 = p - q, \quad \nabla \|p - q\|_1 = \operatorname{sign}(p - q), \tag{2.9}$$

$$\nabla \operatorname{KL}(p|q) = \log(\tfrac{p}{q}), \quad \nabla W(p, q) = \gamma \log(a), \tag{2.10}$$

where, for the gradient of $W(p, q)$, $a \in \mathbb{R}^N$ is the left scaling produced by Sinkhorn's fixed-point algorithm, namely the unique vector with geometric mean 1 such that the matrix $\operatorname{diag}(a) K \operatorname{diag}(q/K^\top a)$ has row-sum p and column-sum q [44, §5]. Note that the notation $\nabla \|p - q\|_1$ is not rigorous, since the ℓ_1-norm is not differentiable everywhere and the sign vector is only a subgradient of that quantity. We side-step this issue, which is only problematic for the 1-norm, by using quasi-Newton solvers such as L-BFGS that work well even with non-smooth objectives [109].

Algorithmic Differentiation of the Jacobian

To simplify this exposition, we introduce two bi-variate functions (Φ, Ψ) to rewrite the iterations of Proposition 2 as operating only on the scalings $b^{(\ell)}(\lambda) \overset{\text{def.}}{=} (b_s^{(\ell)}(\lambda))_{s=1}^S$:

$$P^{(\ell)}(\lambda) = \Psi(b^{(\ell)}(\lambda), \lambda) \text{ where } \Psi(b, \lambda) \overset{\text{def.}}{=} \prod_s \varphi_s(b_s)^{\lambda_s} \tag{2.11}$$

$$b^{(\ell+1)}(\lambda) = \Phi(b^{(\ell)}(\lambda), \lambda) \text{ where } \Phi(b, \lambda) \overset{\text{def.}}{=} \left(\frac{\Psi(b, \lambda)}{\varphi_s(b_s)} \right)_s, \tag{2.12}$$

and $\varphi_s(b_s) \overset{\text{def.}}{=} K^\top \frac{p_s}{K b_s}$, using this time the initialization $b_s^{(0)} = \mathbb{1}$.

We propose to minimize a loss on the *approximate* barycenter $P^{(L)}(\lambda)$ computed after a finite number of iterations $L \geq 1$, to solve:

$$\underset{\lambda \in \Sigma_S}{\operatorname{argmin}}\ \mathcal{E}_L(\lambda) \overset{\text{def.}}{=} \mathcal{L}(P^{(L)}(\lambda), q). \tag{2.13}$$

The gradient formula (2.8) thus needs to be replaced by

$$\nabla \mathcal{E}_L(\lambda) = [\partial P^{(L)}(\lambda)]^\top (u^{(L)}),\ u^{(L)} \overset{\text{def.}}{=} \nabla \mathcal{L}(P^{(L)}(\lambda), q). \tag{2.14}$$

Because $P^{(L)}(\lambda)$ is obtained by recursively applying the same map L times, the application of the transposed Jacobian $[\partial P^{(L)}(\lambda)]^\top$ to the vector $u^{(L)}$ can be computed using *backward recursive differentiation* [124]. This turns out to be particularly efficient, since the overall complexity of computing $[\partial P^{(L)}(\lambda)]^\top (u^{(L)})$ is the same as that of computing the approximate barycenter $P^{(L)}(\lambda)$. Proposition 3 shows that one can compute $[\partial P^{(L)}(\lambda)]^\top (u^{(L)})$ and thus $\nabla \mathcal{E}_L(\lambda)$ using a simple *backward* recursion. Its proof is given in Appendix B of the full paper [28].

Proposition 3. *Let us denote, for $\ell \geq 0$,*

$$\Phi_\lambda^{(\ell)} \overset{\text{def.}}{=} [\partial_\lambda \Phi(b^{(\ell)}(\lambda), \lambda)]^\top \quad \textit{and} \quad \Phi_b^{(\ell)} \overset{\text{def.}}{=} [\partial_b \Phi(b^{(\ell)}, \lambda)]^\top,$$
$$\Psi_\lambda^{(\ell)} \overset{\text{def.}}{=} [\partial_\lambda \Psi(b^{(\ell)}(\lambda), \lambda)]^\top \quad \textit{and} \quad \Psi_b^{(\ell)} \overset{\text{def.}}{=} [\partial_b \Psi(b^{(\ell)}, \lambda)]^\top.$$

One has

$$\nabla \mathcal{E}_L(\lambda) = \Psi_\lambda^{(L)}(u^{(L)}) + \sum_{\ell=0}^{L-1} \Phi_\lambda^{(\ell)}(v^{(\ell)}) \tag{2.15}$$

where $u^{(L)}$ is defined in (2.14) and the vectors $(v^{(\ell)})_{\ell=0}^{L-1}$ are computed using the following backward recursion

$$\forall\, \ell = L-1, L-2, \dots, 0, \quad v^{(\ell-1)} \overset{\text{def.}}{=} \Phi_b^{(\ell-1)}(v^{(\ell)}) \tag{2.16}$$

initialized with $v^{(L)} \overset{\text{def.}}{=} \Psi_b^{(L)}(u^{(L)})$.

The overall numerical scheme to compute $\nabla \mathcal{E}_L(\lambda)$ is detailed in Algorithm 3, which can be obtained by plugging the expression for the differential of (Φ, Ψ) (see their expression in Appendix A of the full paper [28]) into the formulas of Proposition 3. The algorithm first performs a forward loop to compute the barycenter $P^{(L)}(\lambda)$ and then an inverse loop to implement (2.16) and accumulate the sum appearing in (2.15). This efficient implementation computes both the gradient and barycenter in twice as many Gibbs kernel K and $K^\top$ applications as required to compute the barycenter alone, making it a competitive approach, even against naive approximate numerical finite differentiation, which would run $(S+1)/2$ times slower. To summarize, this algorithm only requires $4SL$ convolutions and additional storage for $3NLS$ scalar values at each gradient step carried out to minimize the energy $\mathcal{E}_L$.

```
function SINKHORN-DIFFERENTIATE-WEIGHTS($(p_s)_{s=1}^S, q, \lambda$)
    $\forall\, s, b_s^{(0)} \leftarrow \mathbb{1}$
    $(w, r) \leftarrow (0^S, 0^{S\times N})$
    for $\ell = 1, 2, \dots, L$    // Sinkhorn loop
        $\forall\, s, \varphi_s^{(\ell)} \leftarrow K^\top \frac{p_s}{K b_s^{(\ell-1)}}$
        $p \leftarrow \prod_s \left(\varphi_s^{(\ell)}\right)^{\lambda_s}$
        $\forall\, s, b_s^{(\ell)} \leftarrow \frac{p}{\varphi_s^{(\ell)}}$
    $g \leftarrow \nabla\mathcal{L}(p, q) \odot p$
    for $\ell = L, L-1, \dots, 1$    // Reverse loop
        $\forall\, s, w_s \leftarrow w_s + \langle \log \varphi_s^{(\ell)}, g \rangle$
        $\forall\, s, r_s \leftarrow -K^\top(K(\frac{\lambda_s g - r_s}{\varphi_s^{(\ell)}}) \odot \frac{p_s}{(K b_s^{(\ell-1)})^2}) \odot b_s^{(\ell-1)}$
        $g \leftarrow \sum_s r_s$
    return $P^{(L)}(\lambda) \leftarrow p$, $\nabla\mathcal{E}_L(\lambda) \leftarrow w$
```

Algorithm 3: Given a database of histograms $(p_s)_{s=1}^S$, the input distribution q, weights λ, this function computes $\nabla\mathcal{E}_L(\lambda) \in \mathbb{R}^S$. The barycenter $P^{(L)}(\lambda)$ is obtained as a by-product.

Barycentric Coordinates using Quasi-Newton

With the function SINKHORN-DIFFERENTIATE-WEIGHTS at hand, which is able to compute both the current barycenter estimate $P^{(L)}(\lambda)$ *and* the gradient $\nabla\mathcal{E}_L(\lambda)$, one can now efficiently compute barycentric coordinates λ as a local minimizer of (2.13) through a descent method. Quasi-Newton methods proved very efficient in our experiments. We tested two methods, which turned out to be equally effective: The PQN constrained quasi-Newton of Schmidt et al. [159], which can optimize smooth functions such as $\mathcal{E}_L$ over the simplex Σ_S; a standard quasi-Newton (L-BFGS) over a logarithmic domain using the change of variables $\lambda = \frac{e^\alpha}{\sum_s e^{\alpha_s}} \in \Sigma_S$ and carrying out the optimization over $\alpha \in \mathbb{R}^S$. Despite the energy being non-convex in theory, our paper illustrates the energy function in practice [28].

2.3.3 Wasserstein dictionary learning

Overview

Given data $X \in \mathbb{R}^{N\times M}$ in the form of histograms, *i.e.*, each column $x_i \in \Sigma_N$ (for instance a list of M images with normalized pixel intensities), and the desired number of atoms S, we aim to learn a dictionary D made up of histograms $(d_1, \dots, d_S) \in (\Sigma_N)^S$ and a list of barycentric weights $\Lambda = (\lambda_1, \dots, \lambda_M) \in (\Sigma_S)^M$ so that for each input, $P(D, \lambda_i)$ is the best approximation of x_i according to some criterion $\mathcal{L}$ (see subsection 2.3.2, Gradient of the Loss for examples). Namely, our representation is obtained by solving the problem

$$\min_{D\in(\Sigma_N)^S, \Lambda\in(\Sigma_S)^M} \mathcal{E}(D, \Lambda) \overset{\text{def.}}{=} \sum_{i=1}^M \mathcal{L}\left(P(D, \lambda_i), x_i\right). \tag{2.17}$$

Note the similarity between the usual dictionary learning formulation that seeks to approximate X as a *linear* combinations of atoms $\{d_j\}_j$, and the one above. In our case, however, the reconstruction of the original data happens *via* the nonlinear Wasserstein barycenter operator, $\mathbf{P}(D, \Lambda) = (P(D, \lambda_i))_i$, instead of the (linear) matrix product $D\Lambda$.

```
function SINKHORN-DIFFERENTIATE-DICTIONARY(Data $x \in \Sigma_N$, atoms $d_1, \dots, d_S \in \Sigma_N$, current weights $\lambda \in \Sigma_S$)
    SINKHORN-DIFFERENTIATE-WEIGHTS($(d_s)_{s=1}^S, q, \lambda$)
  $y \leftarrow \mathbf{0}_{S\times N}$
  $z \leftarrow \mathbf{0}_{S\times N}$
  $n \leftarrow \nabla\mathcal{L}(p, x)$
  for $\ell = L, L-1, \dots, 1$      // Backward loop - dictionary
    $\forall s, c_s \leftarrow K((\lambda_s n - z_s) \odot b_s^{(\ell)})$
    $\forall s, y_s \leftarrow y_s + \frac{c_s}{Kb_s^{(\ell-1)}}$
    $\forall s, z_s \leftarrow -\frac{\mathbb{1}_N}{\varphi_s^{(\ell-1)}} \odot K^\top \frac{d_s \odot c_s}{(Kb_s^{(\ell-1)})^2}$
    $n \leftarrow \sum_s z_s$
  return $P^{(L)}(D, \lambda) \leftarrow p, \nabla_D \mathcal{E}^{(L)} \leftarrow y, \nabla_\lambda \mathcal{E}^{(L)} \leftarrow w$
```

Algorithm 4: Computation of dictionary and barycentric weights gradients. This function calls the previously defined SINKHORN-DIFFERENTIATE-WEIGHTS inline

Differentiation of (2.17) relies in part on the computation of the Wasserstein barycenter operator's Jacobians with regard to either the barycentric weights or the atoms. Again, while it is possible to obtain their analytical formulae, they rely on solving a linear system of prohibitive dimensionality for our settings of interest where N is typically large. Following our approach for Wasserstein barycentric coordinates, we instead take the path of automatic differentiation. That is, we recursively differentiate the iterative scheme yielding our algorithm instead of the analytical formula of our Wasserstein barycenter. Instead of (2.17), we thus aim to minimize

$$\min_{D \in (\Sigma_N)^S, \Lambda \in (\Sigma_S)^M} \mathcal{E}_L(D, \Lambda) \overset{\text{def.}}{=} \sum_{i=1}^{M} \mathcal{L}\left(P^{(L)}(D, \lambda_i), x_i\right), \tag{2.18}$$

where $P^{(L)}$ is the approximate barycenter after L iterations. Even when using an entropy penalty term, we have no guarantee on the convexity of the above problem, whether jointly in D and Λ or for each separately. We thus aim to reach a local minimum of the energy landscape $\mathcal{E}_L$ by computing its gradients and applying a descent method. By additivity of $\mathcal{E}_L$ and without loss of generality, we will focus on the derivations of such gradients for a single datapoint $x \in \Sigma_N$ (in which case Λ only comprises one list of weights $\lambda \in \Sigma_S$).

Differentiation of (2.18) yields

$$\nabla_D \mathcal{E}_L(D, \Lambda) = \left[\partial_D P^{(L)}(D, \lambda)\right]^\top \nabla\mathcal{L}(P^{(L)}(D, \lambda), x), \tag{2.19}$$

$$\nabla_\lambda \mathcal{E}_L(D, \Lambda) = \left[\partial_\lambda P^{(L)}(D, \lambda)\right]^\top \nabla\mathcal{L}(P^{(L)}(D, \lambda), x). \tag{2.20}$$

The right-hand term in both cases is the gradient of the loss which is typically readily computable (see subsection 2.3.2, Gradient of the Loss) and depends on the choice of fitting loss. The left-hand terms are the Jacobians of the Wasserstein barycenter operator with regard to either the weights or the dictionary. These can be obtained either by performing the analytical differentiation of the $P^{(l)}$ operator (see 4), or by using an automatic differentiation library such as Theano [180].

Using these gradients, one can find a local minimum of the energy landscape (2.18), and thus the eventual representation Λ and dictionary D, by applying any appropriate optimization method under the constraints that both the atoms and the weights belong to their respective simplices Σ_N, Σ_S. For the applications shown in subsection 2.3.6, we used a Quasi-Newton solver, and the same log-domain change of variables as before to enforce positivity.

Note that in this case, both gradients are fed to the solver as a concatenated vector. It is then beneficial to add a "variable scale" hyperparameter ζ and to multiply all gradient entries related to the weights by that value. Otherwise, the solver might reach its convergence criterion when approaching a local minimum with regards to either dictionary atoms or weights, even if convergence is not yet achieved in the other. This strategy avoids the computation of two different forward Sinkhorn loops to obtain the derivatives in both variables, compared to alternate descent.

Algorithmic Differentiation of the Jacobian

Using the same notations as before, we derive explicit formulas for the derivatives of Wasserstein barycenters with respect to the dictionary D and weights λ.

Proposition 4.

$$\nabla_D \mathcal{E}_L(D,\lambda) = \Psi_D^{(L-1)}\left(\nabla\mathcal{L}(P^{(L)}(D,\lambda),x)\right) + \sum_{l=0}^{L-2}\Phi_D^{(l)}\left(v^{(l+1)}\right), \tag{2.21}$$

$$\nabla_\lambda \mathcal{E}_L(D,\lambda) = \Psi_\lambda^{(L-1)}\left(\nabla\mathcal{L}(P^{(L)}(D,\lambda),x)\right) + \sum_{l=0}^{L-2}\Phi_\lambda^{(l)}\left(v^{(l+1)}\right), \tag{2.22}$$

where:

$$v^{(L-1)} \overset{\text{def.}}{=} \Psi_b^{(L-1)}\left(\nabla L(P^{(L)}(D,\lambda),x)\right), \tag{2.23}$$

$$\forall l < L-1, v^{(l-1)} \overset{\text{def.}}{=} \Phi_b^{(l-1)}\left(v^{(l)}\right). \tag{2.24}$$

See Appendix X of our paper [160] for more details and a proof. An implementation is proposed in 4.

2.3.4 Extensions

Our paper [160] extends these optimization strategies in various ways, including a warm start which initializes the result of a descent iteration with results from previous iterations, an unbalanced formulation that allows for some slack in the mass preservation constraint, and a heavyball strategy that extrapolates the Sinkhorn iterations to accelerate convergence. In this document, I will only develop a log-domain formulation that preserves the separability of the kernel K, and refers the reader to our paper for other extensions.

Stabilization

In its most general framework, representation learning aims at finding a useful representation of data, rather than one allowing for perfect reconstruction. In some particular cases, however, it might also be desirable to achieve a very low reconstruction error, for instance if the representation is to be used for compression of data rather than a task such as classification. In the case of our method, the quality of the reconstruction is directly linked to the selected value of the entropy parameter γ, as it introduces a blur in the reconstructed images. In the case where sharp features in the reconstructed images are desired, we need to take extremely low values of γ, which can lead to numerical problems, *e.g.* because values within the scaling vectors a and b can then tend to infinity. As suggested by Chizat et al. [40] and Schmitzer [161], we can instead perform the generalized Sinkhorn updates in the log-domain. Indeed, noting $u_s^{(l)}, v_s^{(l)}$ as the dual scaling variables, that is,

$$a_s^{(l)} \overset{\text{def.}}{=} \exp\left(\frac{u_s^{(l)}}{\gamma}\right), \qquad b_s^{(l)} \overset{\text{def.}}{=} \exp\left(\frac{v_s^{(l)}}{\gamma}\right),$$

the quantity $-c_{ij} + u_i + v_j$ is known to be bounded and thus remains numerically stable. We can then introduce the stabilized kernel $\tilde{K}(u, v)$ defined as

$$\tilde{K}(u,v) \overset{\text{def.}}{=} \exp\left(\frac{-C + u\mathbb{1}^\top + \mathbb{1}v^\top}{\gamma}\right), \tag{2.25}$$

and notice that we then have

$$\begin{aligned} u_s^{(l)} &= \gamma\left[\log(d_s) - \log(Kb_s^{(l-1)})\right], \\ \left[\log(Kb_s^{(l-1)})\right]_i &= \log\left[\sum_j \exp\left(\frac{-c_{ij} + v_j^{(l-1)}}{\gamma}\right)\right] \\ &= \log\left(\sum_j \tilde{K}(u_s^{(l-1)}, v_s^{(l-1)})_{.j}\right) - \frac{\left[u_s^{(l-1)}\right]_i}{\gamma}. \end{aligned}$$

With similar computations for the v_s updates, we can then reformulate the Sinkhorn updates in the stabilized domain as

$$u_s^{(l)} \overset{\text{def.}}{=} \gamma\left[\log(d_s) - \log\left(\sum_j \tilde{K}(u_s^{(l-1)}, v_s^{(l-1)})_{.j}\right)\right] + u_s^{(l-1)}, \tag{2.26}$$

$$v_s^{(l)} \overset{\text{def.}}{=} \gamma\left[\log(P^{(l)}) - \log\left(\sum_i \tilde{K}(u_s^{(l)}, v_s^{(l-1)})_{i.}\right)\right] + v_s^{(l-1)}. \tag{2.27}$$

This provides a forward scheme for computing Wasserstein barycenters with arbitrarily low values of γ, which could be expanded to the backward loop of our method either by applying an automatic differentiation tool to the stabilized forward barycenter algorithm or by changing the steps in the backward loop of 4 to make them rely solely on stable quantities. However, this would imply computing a great number of stabilized kernels as in (2.25), which relies on nonseparable operations. Each of those kernels would also have to either be stored in memory or recomputed when performing the backward loop. In both cases, the cost in memory or number of operations, respectively, can easily be too high in large scale settings.

Separable log kernel

These issues can be avoided by noticing that when the application of the kernel K is separable, this operation can be performed at a much lower cost. For a d-dimensional histogram of $N = n^d$ bins, applying a separable kernel amounts to performing a sequence of d steps, where each step computes n operations per bin. It results in a $O(n^{d+1}) = O(N^{\frac{d+1}{d}})$ cost instead of $O(N^2)$. As mentioned previously, the stabilized kernel (2.25) is not separable, prompting us to introduce a new stable and separable kernel suitable for log-domain processing. We illustrate this process using 2-dimensional kernels without loss of generality. Let $\mathcal{X}$ be a 2-dimensional domain discretized as an $n \times n$ grid. Applying a kernel of the form $K = \exp(-\frac{C}{\gamma})$ to a 2-dimensional image $b \in \mathcal{X}$ is performed as such:

$$R(i,j) \overset{\text{def.}}{=} \sum_{k=1}^{n}\sum_{l=1}^{n} \exp\left(-\frac{C((i,j),(k,l))}{\gamma}\right) b(k,l),$$

where $C((i,j),(k,l))$ denotes the cost to transport mass between the points (i,j) and (k,l).

Assuming a separable cost such that $C((i,j),(k,l)) \overset{\text{def.}}{=} C_y(i,k) + C_x(j,l)$, it amounts to performing two sets of 1-dimensional kernel applications:

$$A(k,j) = \sum_{l=1}^{n} \exp\left(\frac{C_x(j,l)}{\gamma}\right) b(k,l),$$
$$R(i,j) = \sum_{k=1}^{n} \exp\left(\frac{C_y(i,k)}{\gamma}\right) A(k,j)\,.$$

In order to stabilize the computation and avoid reaching representation limits, we transfer it to the log-domain ($v \overset{\text{def.}}{=} \log(b)$). Moreover, we shift the input values by their maximum and add it at the end. The final process can be written as the operator $K_{LS} : \log(b) \to \log(K(b))$ with K a separable kernel, and is described in 5.

```
**function** LogSeparableKer(Cost matrix $C \in \mathbb{R}^{N\times N}$, image in log-domain $v \in \mathbb{R}^{n\times n}$)
    $\forall k, j,\ x_l(k,j) \overset{\text{def.}}{=} \frac{C_x(j,l)}{\gamma} + v(k,l)$
    $\forall k, j,\ A'(k,j) \overset{\text{def.}}{=} \log\left(\sum_l^n \exp(x_l - \max_l x_l)\right) + \max_l x_l$
    $\forall i, j,\ y_k(i,j) \overset{\text{def.}}{=} \frac{C_y(i,k)}{\gamma} + A'(k,j)$
    $\forall i, j,\ R'(i,j) \overset{\text{def.}}{=} \log\left(\sum_k^n \exp(y_k - \max_k y_k)\right) + \max_k y_k$
    **return** Image in log-domain $K_{LS}(v) = R'$
```

Algorithm 5: `LogSeparableKer` K_{LS}: Application of a 2-dimensional separable kernel in log-domain

This operator can be used directly in the forward loop (see Appendix B of our paper [160]). For backward loops, intermediate values can be negative and real-valued logarithms are not suited. While complex-valued logarithms solve this problem, they come at a prohibitive computational cost. Instead, we store the sign of the input values and compute logarithms of absolute values. When exponentiating, the stored sign is used to recover the correct value.

2.3.5 Applications of Wasserstein Barycenters

This section illustrates our histogram barycentric coordinates for various computer graphics applications. Our paper [28] illustrates an additional application for MRI brain imaging, not detailed here for conciseness.

Optimal Image Color Palettes

We propose in this section a new approach to define target palettes adaptively using multiple images for automatic color grading. Given an input photograph f and a small database of relevant color palettes, we compute first, among all barycenters of these palettes, the one that is the closest to the color palette of f. We then modify the distribution of colors in f to make it match that of this closest palette. Color transfer towards a single predefined palette often results in artefacts, especially when the target palette is very far from the original one [143]. Our approach sidesteps this problem, and considers automatically an infinite family of target palettes that retain the characteristics of the database.

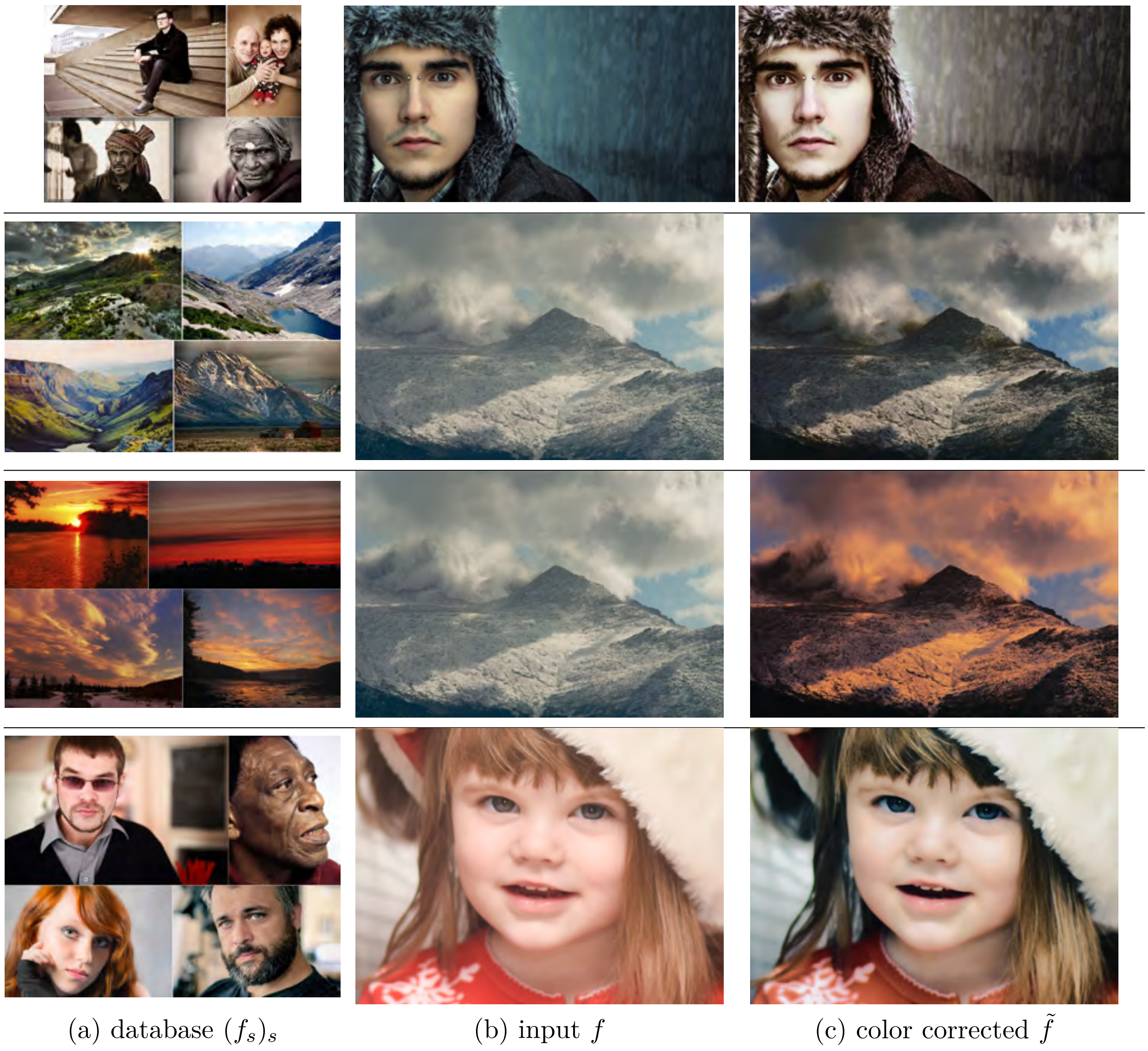


Figure 2.18: From a set $(f_s)_{s=1}^S$ of professional photographs (consisting of, from top to bottom: $S = 7$, $S = 12$, $S = 6$, and $S = 15$ photos), our algorithm projects a photograph f (b) to improved color-corrected or stylized photographs (c). Note that only the four most contributing professional photos (i.e. highest weights λ_s) are shown in (a). Their corresponding optimal weights $(\lambda_s)_s$, from top to bottom, are: $\lambda = (5.10^{-6}, 2.10^{-5}, 0.40, 0.60)$, $\lambda = (10^{-6}, 7.10^{-4}, 0.34, 0.66)$, $\lambda = (3.10^{-6}, 6.10^{-6}, 0.23, 0.77)$ and $\lambda = (0.05, 0.09, 0.29, 0.54)$.

We discretize RGB histograms with $N = 128^3$ values on a uniform grid $(x_i)_{i=1}^N$ of the RGB cube. This defines the histograms $(p_s)_{s=1}^S$ of the input database $(f_s)_{s=1}^S$, and the histogram q of the image f to process. Our algorithm computes the optimal barycenter $P(\lambda)$ using the ground cost $C_{i,j} = \|x_i - x_j\|^2$ and the quadratic loss function $\mathcal{L}(p, p') = \|p - p'\|^2$. The image f is modified into an image $\tilde{f}$, so that the histogram of $\tilde{f}$ is equal, up to a small approximation error, to $P(\lambda)$. This is achieved using the barycentric projection method detailed in [174].

Figure 2.18 illustrates our method using a database of professional photographs. Figure 2.19 shows an application in the context of text-based user interfaces. We use the top 10 results of the Flickr image search engine (`www.flickr.com`) for the query *autumn* to stylize an input summer photograph with a more autumnal aspect. Among various loss functions, the quadratic loss offered the best quality/speed tradeoff for this application.

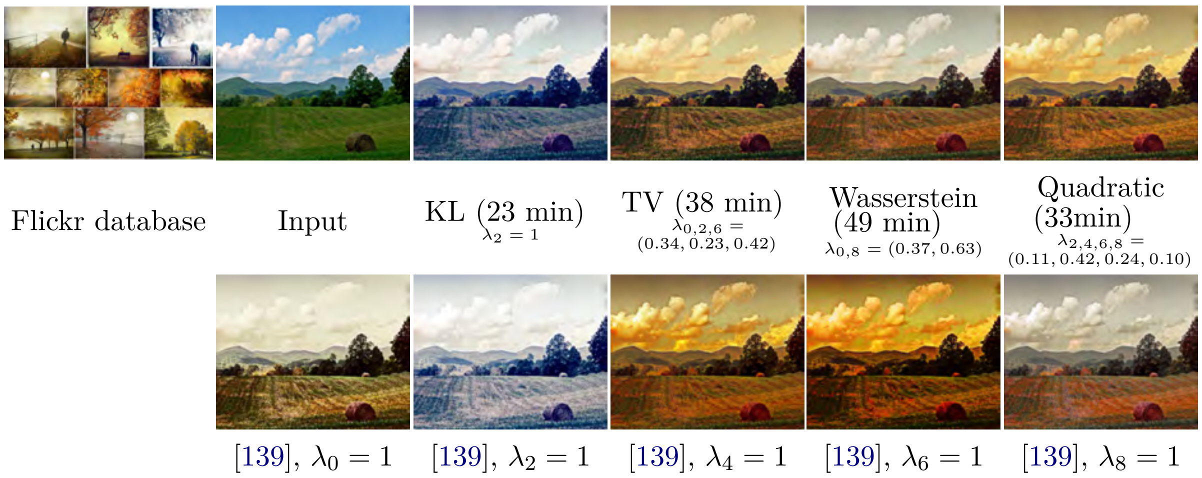


Figure 2.19: Using the image search engine Flickr, we use the top 10 results for the query *autumn* (here, with *Commercial use allowed* and sorted by *Interesting*) and use them to color grade a *summer* image. (First row) For different loss functions, we show the non-zero barycentric coordinates and total computation time using 128^3 voxel RGB color histograms, $L = 60$ and our CPU implementation. (Second row) We use the color matching of Pitie et al. [139] to transfer colors from the most contributing photographs (numbered 0, 2, 4, 6 and 8). As existing techniques use a single target histogram, this can lead to large color distortion.

Sparse Reflectance Inference

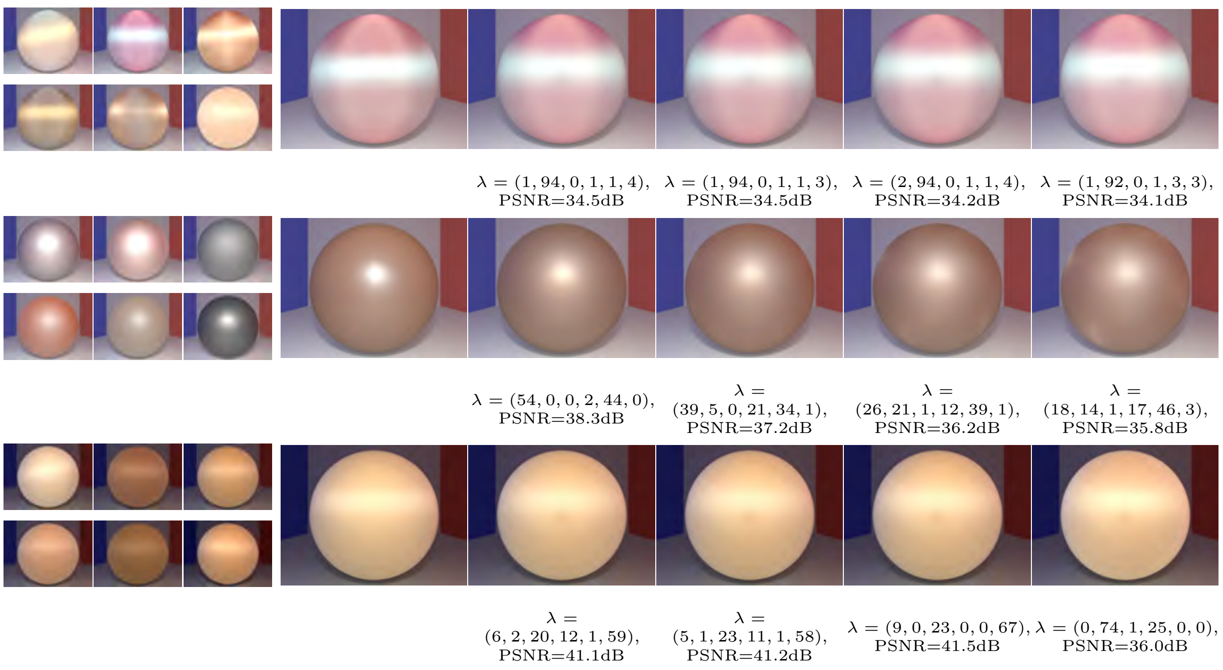
Acquiring reflectance data can be cumbersome. Our method makes it possible to infer reflectance values from sparse data given a database of densely measured reflectances. A Bidirectional Reflectance Distribution Function (BRDF) $\bar{p}$ is a function $\bar{p}(\omega, \xi)$ describing the probability for a photon hitting a surface with a direction ω to be reflected off that surface with a direction ξ, or to be absorbed by that surface. These functions are sampled on discretized hemispheres $(\omega, \xi) \in \Omega^2$ of $N = 288$ points. Since $\bar{p}$ describes distributions of energy, we consider $(\bar{p}(\omega, \cdot))_{\omega \in \Omega}$ as a set of un-normalized histograms on the hemisphere. We thus first normalize the BRDF and define $p(\omega, \xi) \overset{\text{def.}}{=} \bar{p}(\omega, \xi) / \sum_{\xi'} \bar{p}(\omega, \xi')$, so that $(p(\omega, \cdot))_{\omega \in \Omega}$ is a collection of normalized histograms.

Given a database $(p_s)_{s=1}^{S}$ of such normalized BRDF $p_s(\omega, \xi)$, we extend (1.17) to define barycenters by jointly optimizing over all incoming direction ω

$$P(\lambda) \overset{\text{def.}}{=} \underset{(p(\omega,\xi))_{\xi,\lambda}}{\operatorname{argmin}} \sum_{\omega \in \Omega} \sum_{s} \lambda_s W(p(\omega, \cdot), p_s(\omega, \cdot)). \tag{2.28}$$

We use the cost matrix $C_{i,j} = d(x_i, x_j)^2$ where d is the geodesic distance on the hemisphere. Using this extended notion of barycenters, we use (2.13) to define barycentric coordinates of a normalized BRDF q computed from some BRDF $\bar{q}$, where the loss $\mathcal{L}$ is the Wasserstein loss , extended to collections of histograms. The gradient of $\mathcal{E}_L$ is now obtained by summing the gradient contribution of all incident directions ω, so that we can use Algorithm 3 for the computation of the optimal λ. One finally recovers the interpolated BRDF $\bar{P}(\lambda)$ from $P(\lambda)$ by re-introducing the initial scaling factor of $\bar{q}$, i.e. $\bar{P}(\lambda)(\omega, \xi) \overset{\text{def.}}{=} P(\lambda)(\omega, \xi) \sum_{\xi'} \bar{q}(\omega, \xi')$

Fig. 2.20 shows typical results for isotropic and anisotropic BRDFs from the UTIA database [63]. To simulate a sparse acquisition setup, we progressively decimate (which corresponds to replacing some histogram values by 0) an input BRDF $\bar{q}_0$ by up to 95% to obtain the input BRDF q, prior to computing barycentric coordinates λ. We observe relatively good reconstruction quality even for highly degraded BRDFs. This suggests our approach could alleviate the BRDF capture process when reasonably similar materials have already been captured at higher resolution.

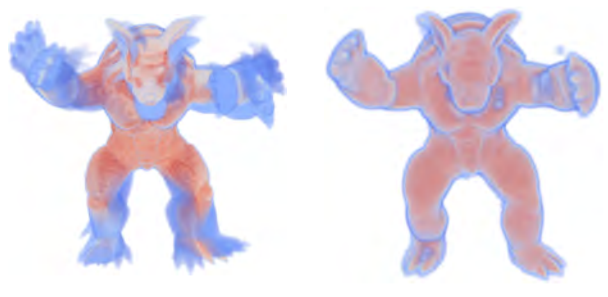

$\lambda = (1, 94, 0, 1, 1, 4)$, PSNR=34.5dB
$\lambda = (1, 94, 0, 1, 1, 3)$, PSNR=34.5dB
$\lambda = (2, 94, 0, 1, 1, 4)$, PSNR=34.2dB
$\lambda = (1, 92, 0, 1, 3, 3)$, PSNR=34.1dB
$\lambda = (54, 0, 0, 2, 44, 0)$, PSNR=38.3dB
$\lambda = (39, 5, 0, 21, 34, 1)$, PSNR=37.2dB
$\lambda = (26, 21, 1, 12, 39, 1)$, PSNR=36.2dB
$\lambda = (18, 14, 1, 17, 46, 3)$, PSNR=35.8dB
$\lambda = (6, 2, 20, 12, 1, 59)$, PSNR=41.1dB
$\lambda = (5, 1, 23, 11, 1, 58)$, PSNR=41.2dB
$\lambda = (9, 0, 23, 0, 0, 67)$, PSNR=41.5dB
$\lambda = (0, 74, 1, 25, 0, 0)$, PSNR=36.0dB

(a) BRDF database $(\bar{p}_s)_s$ (b) Reference $\bar{q}_0$ (c) $\bar{P}(\lambda)$, all data (d) $\bar{P}(\lambda)$, 25% data (e) $\bar{P}(\lambda)$, 10% data (f) $\bar{P}(\lambda)$, 5% data

Figure 2.20: We fit a measured anisotropic BRDF $\bar{q}$ (b) using a basis of $S = 6$ measured BRDFs $(\bar{p}_s)_{s=1}^S$ (a). We obtain an approximation $\bar{P}(\lambda)$ (c) that remains robust when decimating measurements prior to the fitting (d,e,f). Reported λ are normalized so that $\sum_s \lambda_s = 100$, and PSNR values computed on BRDFs.

Inferring missing geometry

Capturing geometries can be difficult due to partial occlusions, measurement noise, or unreachable camera angles. Given a database of input 3-D models, our tool can be used to infer missing geometry in an input mesh. We voxelize all shapes on a $N = 192^3$ uniform 3-D grid $(x_i)_{i=1}^N$ and we use a ground cost $C_{i,j} = \|x_i - x_j\|^2$, and $\mathcal{L} = \mathrm{KL}$ as loss function. Each shape is represented as a normalized histogram representing the uniform distribution inside this shape, and a uniform mass of $\varepsilon = 10^{-4}$ outside for compatibility with KL loss. We account for the mass missing in the input geometry by roughly estimating the amount of missing mass, and normalizing the input histogram accordingly. Specifically, if α percent of the input shape is missing, we use a loss of the form $\mathrm{KL}(P(\lambda), (1-\alpha)\frac{q}{\sum q})$. Figure 2.21 compares the Wasserstein and Euclidean projections, and Figure 2.22 illustrates additional results.



Euclidean Projection Our Projection $P(\lambda)$

Figure 2.21: Euclidean ($\lambda = (0.25, 0.29, 0.01, 0.08, 0.11, 0.19)$) and our Wasserstein projection ($\lambda = (0.30, 0.12, 0.07, 0.08, 0.16, 0.27)$. The Euclidean projection yields linearly blended shapes.

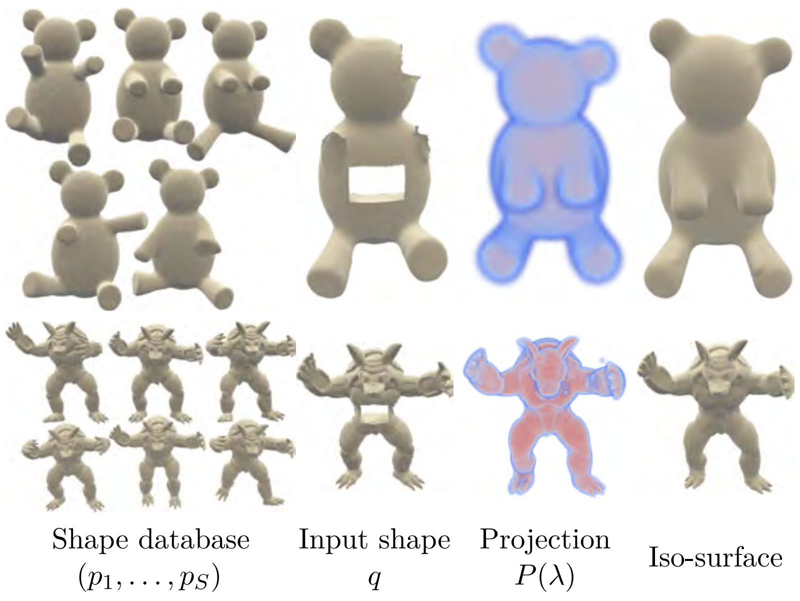


Figure 2.22: We fit a 192^3 voxelized digital shape q, on a database of similar shapes $(p_1, \ldots, p_S)$ [39]. We obtain a projection $P(\lambda)$, with computed weights $\lambda = (7.10^{-4}, 0.93, 0.07, 6.10^{-4}, 4.10^{-4})$ (top) and $\lambda = (0.30, 0.12, 0.07, 0.08, 0.16, 0.27)$ (bottom), from which we extract a smooth iso-surface.

2.3.6 Applications of Wasserstein Dictionary Learning

This section details a few applications we developed. An additional application for Point Spread Function learning has been developed by our co-authors at CEA, and is exposed in the paper [160].

Cardiac sequences

We tested our dictionary learning algorithm on a reconstructed MRI sequence of a beating heart (Figure 2.23). The goal is to learn a dictionary of four atoms, representing key frames of the sequence. For this application, we used 13 frames of 272×240, a regularization $\gamma = 2$, and a scale between weights and atoms of $\zeta = N/(100 * M)$, $N = 272 \times 240$, $M = 13$ frames. We used a quadratic loss because it provided the best results in terms of reconstruction and representation. We found 25 iterations for the Sinkhorn algorithm to be a good trade-off between computation time and precision. The recovered "barycentric path" (polyline of the barycentric points) is a cycle and residuals are small, which means the algorithm is successful at finding those key frames that, when interpolated, represent the whole dataset.

Wasserstein faces

It has been shown that images of faces, when properly aligned, span a low-dimensional space that can be obtained via PCA. These principal components, called Eigenfaces, have been used for face recognition [182]. We show that, with the right setting, our dictionary learning algorithm produces atoms that can be interpreted more easily than their linear counterparts.

We illustrate this application on the MUG facial expression dataset [3]. From the images shown in Figure 2.24(a), we inverted colors, because it produced lower residuals. We used a

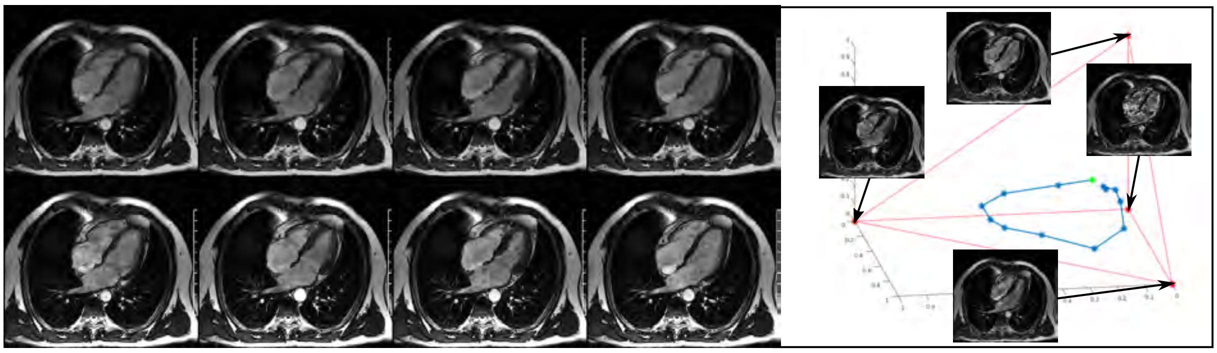


Figure 2.23: Left: Comparison between four frames (out of 13) of the measures (lower row) and the same reconstructed frames (upper row). Right: plot of the reconstructed frames (blue points) by their barycentric coordinates in the 4-atom basis, with each atom (red points) at the vertices of the tetrahedra. The green point is the first frame.

total of 20 (224 × 224) images of a single person performing five facial expressions and learned dictionaries of five atoms using PCA, NMF, a K-SVD implementation [152], and our proposed method. For the last, we set the number of Sinkhorn iterations to 100 and the maximum number of L-BFGS iterations to 450. As illustrated by Figure 2.24, using a KL loss, our method reaches similarly successful reconstructions given the low number of atoms, with a slightly higher mean PSNR of 33.8 compared to PSNRs of 33.6, 33.5 and 33.6 for PCA, NMF and K-SVD respectively. We demonstrate a face editing application and compare various loss functions in our paper [160]. In particular, a Wasserstein loss produces a visually more appealing dictionary albeit with slightly higher reconstruction errors.

Literature learning

We use our algorithm to represent literary work. To this end, we use a bag-of-words representation [156], where each book is represented by a histogram of its words. In this application, the cost matrix C (distance between each word) is computed exhaustively and stored. We use a semantic distance between words, computed from the Euclidean embedding provided by the GloVe database (Global Vectors for Word Representation) [133].

To demonstrate our algorithm's performance, we created a database of 20 books by five different authors. To keep the problem tractable we only considered words that are between seven and eight letters long.

Our algorithm is able to group novels by author, recognizing the proximity of lexical fields across the different books (see Figure 2.25). Atom 0 seems to be representing Charlotte Brontë's style, atoms 1 and 4 that of Mark Twain, atom 2 that of Arthur Conan Doyle, and atom 3 that of Jane Austen. Charles Dickens appears to share an extended amount of vocabulary with the other authors without it differing enough to be represented by its own atom, like others are.

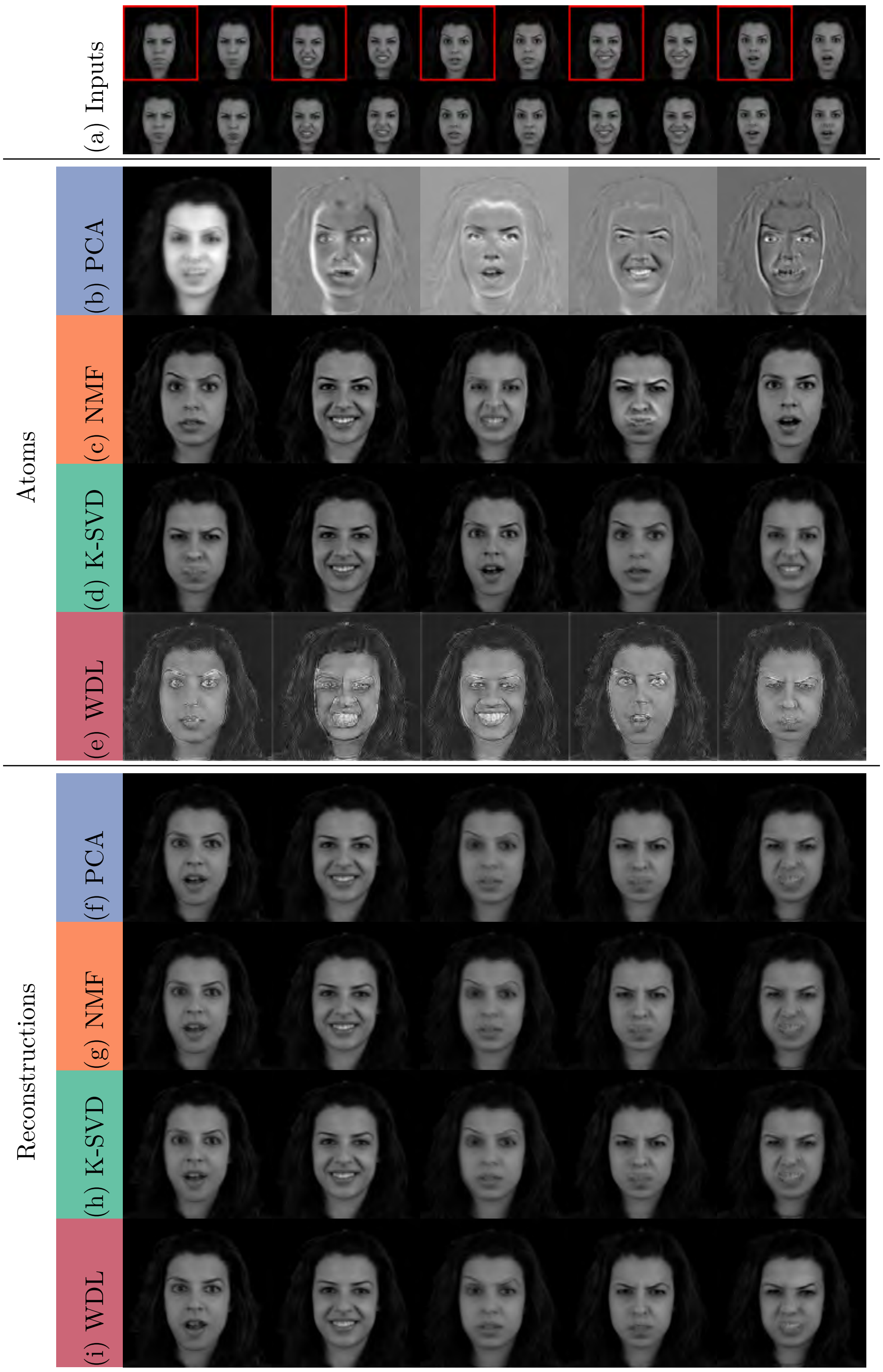


Figure 2.24: We compare our method with Eigenfaces [182], NMF and K-SVD [152] as a tool to represent faces on a low-dimensional space. Given a dataset of 20 images of the same person from the MUG dataset [3] performing five facial expressions four times (row (a)), we project the dataset on the first five Eigenfaces (row (b)). The reconstructed faces corresponding to the highlighted input images are shown in row (f). Rows (c) and (d), respectively, show atoms obtained using NMF and K-SVD and rows (g) and (h) their reconstructions. Our method results in five atoms shown in row (e) with reconstructions shown in row (i).

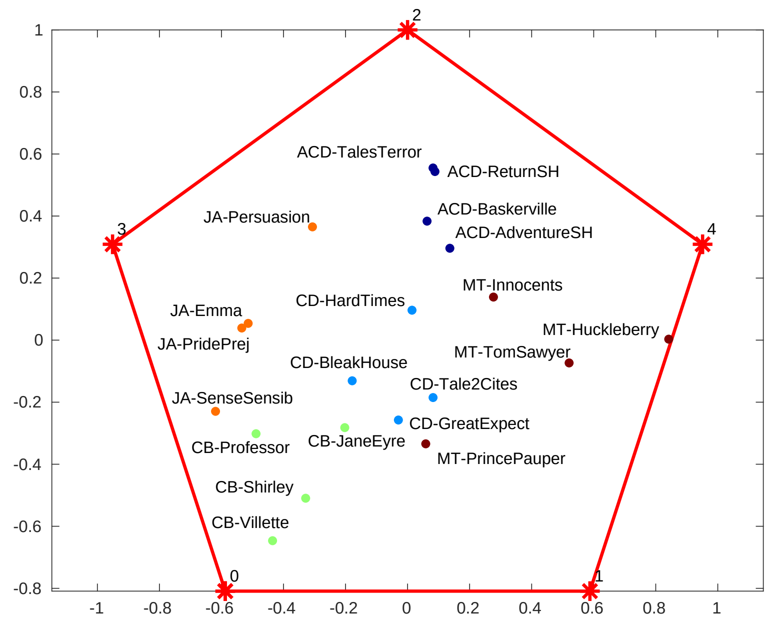


Figure 2.25: Our Wasserstein dictionary learns five atoms among 20 books by five authors. Each book is plotted according to its barycentric coordinates with regard to the learned atoms.

2.4 Conclusions on Optimal Transport

Generally speaking, optimal transport has been widely successful throughout the years, with a Nobel loreate in economics (Leonid Kantorovich, 1975), and two Fields Medalists (Cedric Villani, 2010, and Alessio Figalli, 2018). However, since I started investigating this subjects, I have been the witness of an explosion of interest towards this topic in the computer science community – perhaps not nearly as dramatic as the explosion towards deep learning during the same time span, but still... I am happy to have started contributing near the onset of this wave.

This explosion has provided numerical optimal transport with now relatively fast tools, at least compared to the state-of-the-art nearly 10 years ago. My contributions towards these numerical tools is first a fast network simplex implementation (subsection 2.1.2) that, despite its high theoretical complexity, remains fast, in the order of 3 minutes for transporting 50k particles using a quadratic cost function and orders of magnitude faster than commercial solvers such as IBM CPLEX. I believe it is still one of the fastest exact solver to compute transport plans between arbitrary discrete measures with arbitrary cost functions. My second contribution to numerical optimal transportation is an approximation based on Radon transforms to Wasserstein barycenters, which is even faster and more stable than entropy regularized methods, and came concurrently with a similar method by Tóth and Csébfavi [181]. While it enjoys similar properties as the actual N-d Wasserstein barycenter, it however does not converge in the general case to the real Wasserstein barycenter (while, in theory, entropy regularized methods can approximate as close as desired to the Wasserstein barycenter) and does not provide transport plans.

I implemented and played with many solvers. Table 2.26 summarizes various optimal transport solvers I had the chance to implement, in their functionalities and speed (unless explicited otherwise, timings were measured on my own implementations).

This trend for optimal transport has been intensified by the machine learning community, who saw optimal transport as an interesting tool to measure the mismatch between various types of data points (including histograms), and who saw the interest of the Wasserstein geometry on which to carry machine learning tasks. My contributions in this area has been to develop efficient Wasserstein barycentric projections and dictionary learning methods, and to show applications in computer graphics.

	Discussion	Setup	Cost	# measures	Typical Speed	Notes
Network Simplex	Sec. 2.1.2	Discrete	General	2	3 min for 50k weighted diracs (my implem.)	Most general ; speed depends on cost function
Power Diagrams	Sec. 1.2.2	Semi-Discrete (continuous 2d or 3d density often discretized on a grid towards weighted Diracs)	Quadratic	2	[108] reports tens of seconds for 10 millions Diracs using both CPU and GPU	Extension to 1-Wasserstein distances studied in [80]
Entropy Regularized	Sec. 1.2.1	Discrete	General (regular grids + quadratic cost if using fast convolutions)	Many	≈25 seconds for barycenter of three 1024x1024 grids (100 iterations, convolutional)	Approximate ; speed depends on regularization parameter ; quite unstable unless using much slower log-domain approaches
Sliced	Sec. 1.2.3	Discrete with non-weighted n-d Diracs.	N/A	Many	barycenter of three 40K Dirac distributions in 18 seconds	Inexact: does not converge towards n-d optimal transport
Radon	Sec. 2.2	Continuous 2d densities, discretized on regular grids	N/A	Many	14 seconds for 32 barycenters of three 1024x1024 grids on single core	Inexact: does not converge towards n-d optimal transport ; does not give transport plans.
Fluid PDE	[17, 129]	Continuous 2d densities discretized on grids	quadratic cost	2	2 minutes for 256x256 grids using my implementation of [129]	

Figure 2.26: Non-exhaustive and approximate comparison of optimal transport numerical solvers that I had the opportunity to implement myself at least in some variant.

3 Temporal Coherence of Image Processing Algorithms

Many image processing algorithms, when applied to each frame of a video, result in temporal inconsistencies, such as flickering or lower frequency artifacts. My post-doc at Harvard University in Hanspeter Pfister's group was funded by an NSF grant "Images through Time" to investigate such issues. This led at the time to propose solutions to the problems of video color grading (which turned out to largely rely on optimal transport theory, section 3.1) and intrinsic video decomposition (section 3.2). While at CNRS, I later proposed a general purpose approach to make most types of image filtering operations temporally consistent for both single and then multi-view videos (section 3.3).

3.1 Video Color Grading

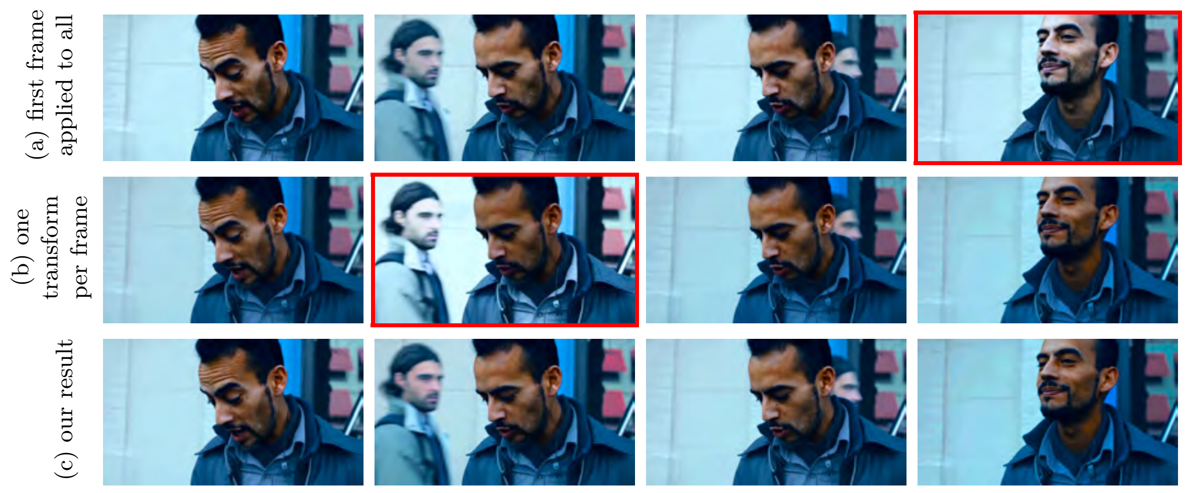


Figure 3.1: The first three frames are nearly consecutive and the last one is more distant. (a) Computing a color transform once at the beginning and applying it to the entire sequence yields results that degrade as time passes (e.g., the bluish face). (b) Evaluating the transform at each frame produces temporally inconsistent results (e.g., brightening when the man appears). (c) Our result is stable and does not drift.

The color palette used in a movie often plays a critical role in establishing its visual look, to locate a movie in place and time or to evoke certain emotions. Over time, certain looks have come

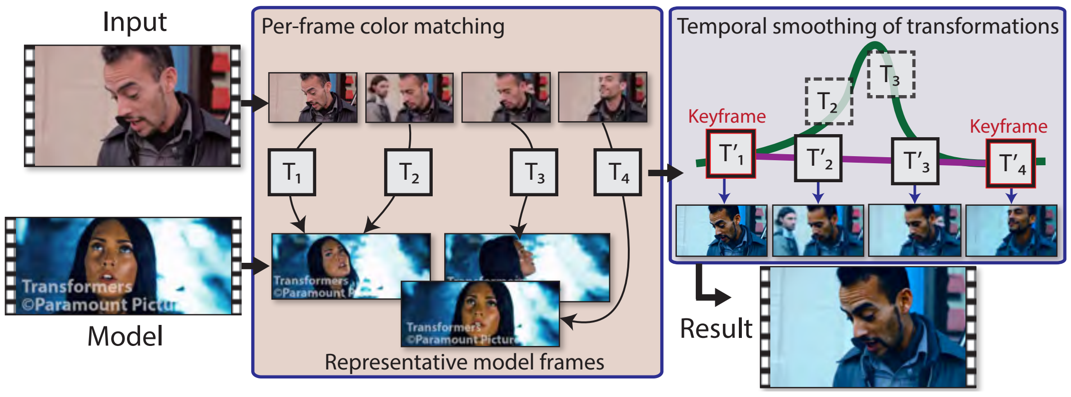

Input
Per-frame color matching
Temporal smoothing of transformations
T1
T2
T3
T4
Keyframe
Keyframe
T'1
T'2
T'3
T'4
Transformers ©Paramount Pictures
Model
Representative model frames
Result

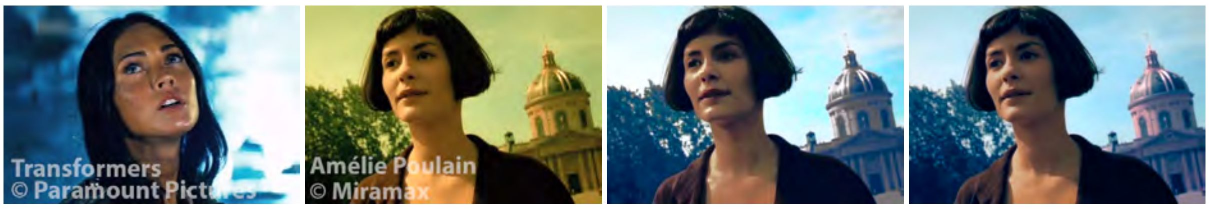
Figure 3.2: The first stage color matches each frame of the input video to one image out of a set of representative model video frames. The second stage filters these color transformations using a novel approximate curvature flow technique. This technique treats the set of transformations as a curve in high-dimensional space and detect points of low curvature (i.e., keyframes). Interpolating the color transformations at these keyframes produces a temporally coherent set of transformations that result in a high-quality color graded video.

to represent entire genres of movies – e.g., *Film Noir's* use of low-key lighting and contrast between light and shadows. The visual style of a movie is often carefully devised by the cinematographer, and executed by a team of skilled colorists who manipulate the movie's colors – through a process known as *color grading*. The goal of our work published at ACM SIGGRAPH [30], is to make it possible for amateur users to apply popular color grading styles to their own home videos with minimal user interaction. We achieve this using an example-based approach; users are asked to specify a *model* video (or image) that represents the color grading style they like, and our technique transfers the color palette of this model video to their clip.

This approach of matching colors has been extensively studied for images [148, 140]. However, applying image color matching naively to every frame of video sequences leads to temporal artifacts, and computing a single color transformation for the entire video sequence results in poor color matching (see Fig. 3.1). Our approach exposed in this section alleviates these issues.

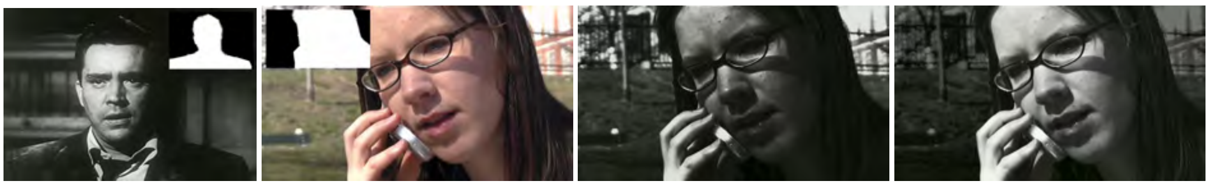
3.1.1 Overview

Given a user-specified model video $\mathbf{M}$ and an input clip $\mathbf{I}$, we seek to transfer the color palette of $\mathbf{M}$ to $\mathbf{I}$ to create the color graded result $\mathbf{O}$. Optionally, a segmentation of the two videos into foreground and background regions can be used for improved matching results. We propose a two-stage approach, shown in Figure 3.2 First, we estimate a color transfer function $\mathbf{T}_t$ for *every* input video frame $\mathbf{I}_t$ that maps its colors to those of a *representative model video frame* (subsection 3.1.2), and apply it to produce the result, i.e., $\mathbf{O}_t = \mathbf{T}_t(\mathbf{I}_t)$. Applying these transformations naively to the input video leads to temporal artifacts. To avoid this, in a second step (Sec. 3.1.3), we filter the per-frame transformations to make them temporally coherent. We interprete the series of color transformations $\mathbf{T}$ as a curve in a transform space: points at which this curve has a high curvature correspond to frames with temporal artifacts. We hence detect points on this curve that have a low curvature (called *keyframes*) and interpolate the transformations in-between these points to build a set of temporally coherent color transformations $\mathbf{T}'$. Applying $\mathbf{T}'_t$ to the input frames $\mathbf{I}_t$ produces the final color graded output video frames $\mathbf{O}_t$ (Fig. 3.1, bottom). Our technique is able to handle a number of visual styles (including *Film Noir*, *bleach bypass*, *orange-teal*) and a wide range of input video sequences.



(a) Model video frame "Transformers" (b) Input video frame "Amélie" (c) Naive histogram matching (d) Our color transfer model

Figure 3.3: We would like to transfer the color palette of the model video frame (a) to an input video frame (b). (c) The simplest way is to apply histogram matching in each color channel (and the foreground and background) independently. This often produces artifacts (see Amélie's face). (d) Our color transfer model removes these artifacts.



(a) Model video frame "D.O.A" (b) Input video frame (c) Color transfer without segmentation (d) Color transfer with segmentation

Figure 3.4: Some color styles have spatially-varying characteristics (a) that cannot be replicated with global color adjustments. (c) Global color adjustments do not replicate the style of (a). (d) Using a user-specified segmentation (shown in a and b) reproduces more faithful colors.

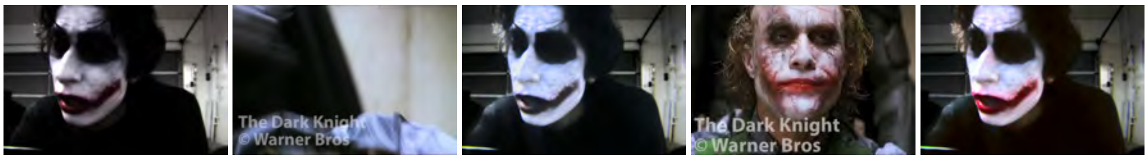
3.1.2 Single-frame Color Matching

Our method first computes per-frame color transformations $\mathbf{T}_t$ to match each input frame $\mathbf{I}_t$ to a desired image or model frame $\mathbf{M}_t$.

Color Transfer Model

We seek to transfer the color distribution of a model video frame $\mathbf{M}_t$ to an input video frame $\mathbf{I}_t$. A color transfer method that is too flexible such as full color histogram matching would highly depend on the footage content and be prone to flickering. On the other hand, a transfer model without enough flexibility such as matching only the mean color would not be expressive enough. We strike a balance by first transferring the luminance values (CIE-Lab, D65 illuminant [149]) using a smooth remapping curve $\mathbf{T}^\ell = H^{-1}(\mathbf{I})(H(\mathbf{M})) * \mathcal{N}(0, \sigma)$, with $H(\mathbf{I})$ the cumulative luminance histogram of $\mathbf{I}$. This corresponds to an optimal transport map, convolved by a Gaussian (typically, $\sigma = 10\%$). We then match the chrominance values using three affine transforms, one for each of shadows, midtones, and highlights, computed via the closed-form optimal transport solution of Pitié and Kokaram [139] for gaussian distributions (Equation 1.14).

In cases where a segmentation into foreground and background objects is needed (Fig. 3.4), we compute a hard segmentation using SnapCut [12], transform it to a soft matte [105] and regularize it in time by filtering [102]. We then compute our color transformation for foreground and background, respectively, for a total of six transformations per frame (one luminance curve and two chrominance affine transforms per segment).



(a) Input video frame (b) Bad model frame "The Dark Knight" (c) Color transfer with bad model match (d) Good model frame "The Dark Knight" (e) Color transfer with good model match

Figure 3.5: When an input frame (a) is color graded with a bad model frame (b), the results are often bad (c). We pre-cluster the model video into a few representative model frames and match each input frame to the most similar representative (d) improving the result (e).

Finally, to reduce halo artifacts, we follow an approach inspired by Rabin et al. [143] and Pouli and Reinhard [141]. We compute the difference between the original and transformed luminance images, apply an edge-aware smoothing filter [71] to this difference, and add this filtered difference back to the original image. Fig. 3.3 illustrates the advantage of using our color transfer model.

Representative model frames

We automatically pair each input video frame with a model video frame. We first perform a K-medoids clustering [132] to summarize the model sequence by a few *representative frames*, using the gaussian optimal transport metric of Equation 1.13 summed over the three luminance bands. Given the representative frames, we then match each input frame to the nearest representative frames using the same metric, and estimate the corresponding color transformation. The benefits of this process is illustrated in Fig. 3.5.

3.1.3 Differential Color Transform Filtering

Each of the transformations $\mathbf{T}_t$ has been computed in isolation and applying them directly to the video frames produces a temporally inconsistent result. One could apply a spatio-temporal filter to the video pixels but this would produce a smooth result in which high-frequency details are lost. We address this issue by temporally filtering the *color transforms* applied to video frames, as briefly described below.

We analyze the color transforms, $\mathbf{T}$, as a curve in the space of color transforms. We observe that rapid changes of colors resulting from sudden changes in the scene (for e.g., when the passer-by enters the frame in Fig. 3.1) correspond to frames where the estimated transforms vary sharply, and consequently *the curvature of the transformation curve is high at these points* and need processing. First, we extend the notion of curvature to the color transforms previously computed. Second, we find points of low curvature and select them as keyframes. Finally, we interpolate the color transforms at the detected keyframes to compute a temporally smooth set of color transformations for the entire sequence. Fig. 3.6 illustrates our approach.

Estimating the curvature

We would like to define a notion of curvature for the curve $\mathbf{T}$ in the space of color transformations, as the rate of change of the tangent to the curve. A simple approach is to work in a

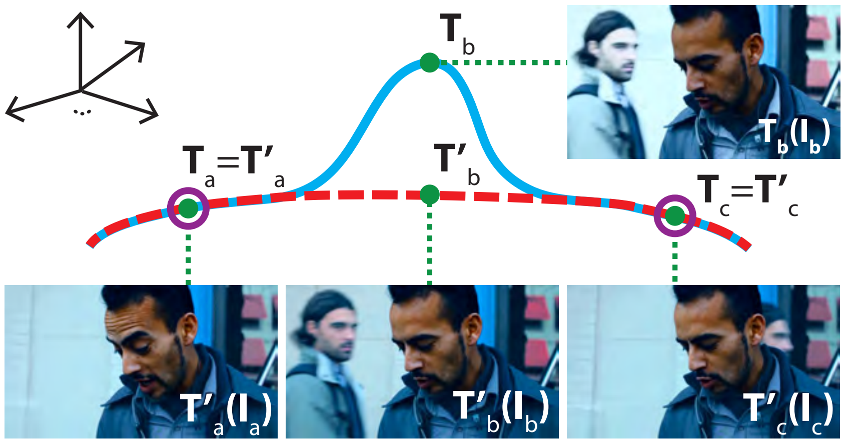


Figure 3.6: The per-frame color transformations $\mathbf{T}_t$ can be analyzed as points on a curve in a high-dimensional space (blue curve, green dots). Regions of high curvature correspond to transformations that cause temporal artifacts (for e.g., the background in the top right image brightens when a person walks across the back). We detect keyframes (purple circles) by sampling regions of low curvature and compute a smooth transform curve by interpolating the original transformations at the keyframes (red dashed curve, green dots). Applying the interpolated transforms to the input frames produces a temporally consistent result (bottom middle).

Euclidean space, thus defining curvature as the magnitude of the second derivative of the color transform curve with respect to time, $||\ddot{\mathbf{T}}||$, and linearly interpolating color transforms between keyframes. However, linear interpolation does not accurately model the rotations of the color wheel and looses correlations between color channels. Therefore, we use a Euclidean space only for the translational components of the color transformations. We instead handle rotations and scalings via a Wasserstein space that accounts for their properties [178] more accurately. But this requires accounting for the curvature of the Wasserstein space itself.

The rate of change of tangent vectors along the curve is characterized using the *covariant derivative* $\nabla_{\dot{\mathbf{T}}}\dot{\mathbf{T}}$ [52], that generalizes directional derivatives to manifolds. Standard curvature flow techniques operate on a modified version of the covariant derivative – the *second fundamental form* [86] – but this does not account for all the variations we are interested in. For instance, a non-constant speed geodesic results in a vanishing second fundamental form. Instead, we compute the curvature K_t at time t as the magnitude of the covariant derivative vector, i.e., $K_t = ||\nabla_{\dot{\mathbf{T}}}\dot{\mathbf{T}}||$.

Our color transformation model consists of a non-linear luminance mapping, and three rotations/scalings and translations for the chrominances. To simplify the analysis, we approximate the 1D luminance mapping by an affine transformation (i.e., a scale and a translation) only when estimating curvatures. As a result, the color transform space is a finite-dimensional space of dimension 17 (or 34 when using a segmentation). These 17 dimensions consist of two 1D spaces (luminance scaling and translation), three 2D spaces (chrominance translations for shadows, midtones and highlights), and three 3D spaces (one 2×2 symmetric matrix representing the chrominance rotation and scaling for shadows, midtones, and highlights), and these transforms can be represented by matrices. We use the property that the squared curvature $||\nabla_{\dot{\mathbf{T}}}\dot{\mathbf{T}}||^2$ in a Cartesian-product space can be computed as the sum of the squared curvature for each subspace [52]. The subspace of the luminance and chrominance translations is Euclidean, and the covariant derivative here corresponds to the standard second derivative with respect to time. We will thus focus on the luminance scaling and the chrominance rotation sub-components of the color transformations.

We parameterize the space of transformations using a d-dimensional basis. For instance, for the 2×2 symmetric chrominance linear transform matrices, a natural basis consists of $\{\mathbf{x_1} = \left[\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right], \mathbf{x_2} = \left[\begin{smallmatrix}0&1\\1&0\end{smallmatrix}\right], \mathbf{x_3} = \left[\begin{smallmatrix}0&0\\0&1\end{smallmatrix}\right]\}$. Any 2×2 symmetric matrix can be decomposed as a weighted sum of these $d = 3$ basis matrices. We denote the i^{th} components of the matrix $\mathbf{T}_t$ in this basis as T^i $(i = 1 \dots d)$. Obtaining the covariant derivative then requires a set of real coefficients called the *Christoffel symbols*, $\Gamma^i_{k,\ell}$, of the Wasserstein metric that account for the space curvature [4], where i, k and ℓ range from 1 to the dimension d of the space ($d = 1$, 2 or 3), defined in Equation 3.2. We can compute the i^{th} component of the covariant derivative $\nabla_{\dot{\mathbf{T}}}\dot{\mathbf{T}}$ as:

$$\nabla^i_{\dot{\mathbf{T}}}\dot{\mathbf{T}} = \ddot{T}^i + \sum_{k,\ell} \Gamma^i_{k,\ell}\dot{T}^k\dot{T}^\ell, \quad i = 1 \dots d \tag{3.1}$$

where $\dot{T}^i$ and $\ddot{T}^i$ are the first and second derivatives of the i^{th} component of the transformations $\mathbf{T}$ in the basis $\mathbf{x_k}$ w.r.t. time and are approximated with standard finite differences.

The Christoffel symbols are expressed by:

$$\Gamma^i_{k,\ell} = \frac{1}{2}\sum_{m=1}^{d} g^{im}\left(\frac{\partial g_{mk}}{\partial x^\ell} + \frac{\partial g_{m\ell}}{\partial x^k} - \frac{\partial g_{k\ell}}{\partial x^m}\right), \quad i, k, \ell = 1 \dots d, \tag{3.2}$$

g^{ij} correspond to the coefficients of the inverse of the $d \times d$ first fundamental form of coefficients g_{ij} defined by Takatsu's metric g [178], computed as in subsection 1.1.6. For the case $d = 1$ of our 1D luminance scaling, we have $\Gamma^1_{1,1} = -\frac{1}{2\,s}$ where s is the luminance scaling at the current point on the curve. The squared curvature at time t, K_t^2, is finally computed as the sum of the squared norms of the covariant derivatives in the individual subspaces. The squared norm in one subspace is computed as $g(\nabla_{\dot{\mathbf{T}}}\dot{\mathbf{T}}, \nabla_{\dot{\mathbf{T}}}\dot{\mathbf{T}})$. More details on the terms introduced, a closed form derivation, C++ code and additional results are provided in supplemental material of our paper [30].

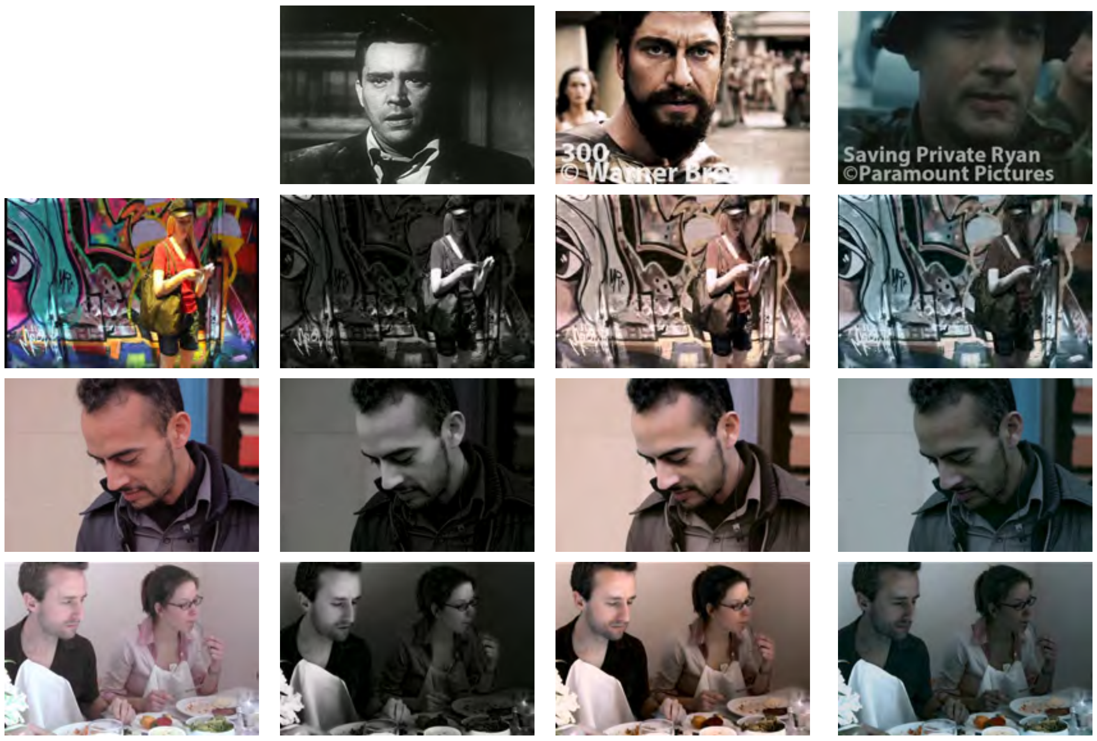
Approximate curvature flow

Curvature flow is a mathematical method that advects a manifold locally with a speed proportional to its local curvature, and is often used to smooth 2D surfaces using the surface mean curvature (the trace of its second fundamental form). Having computed a curvature value, K_t, for each frame, we similarly reduce the total curvature of our curve $\mathbf{T}$ by detecting segments of high curvature and replacing them by (smoother) geodesics.

We first generate a set of r keyframes in areas of low curvature by random sampling the probability function corresponding to the inverse of the curvature, i.e., $p(t) \propto 1/K_t$, and locally moving these samples to the location of lowest curvature. This produces more keyframes in areas of low curvature. We then interpolate each sub-component of $\mathbf{T}$ between keyframes to produce the desired smooth color transforms $\mathbf{T}'$. In particular, the symmetric positive definite chrominance transform matrices are interpolated using Equation 1.14, and the non-linear luminance mappings are linearly interpolated. Finally, the interpolated color transformations, $\mathbf{T}'$, are applied to the input frames to produce the color graded output video frames.

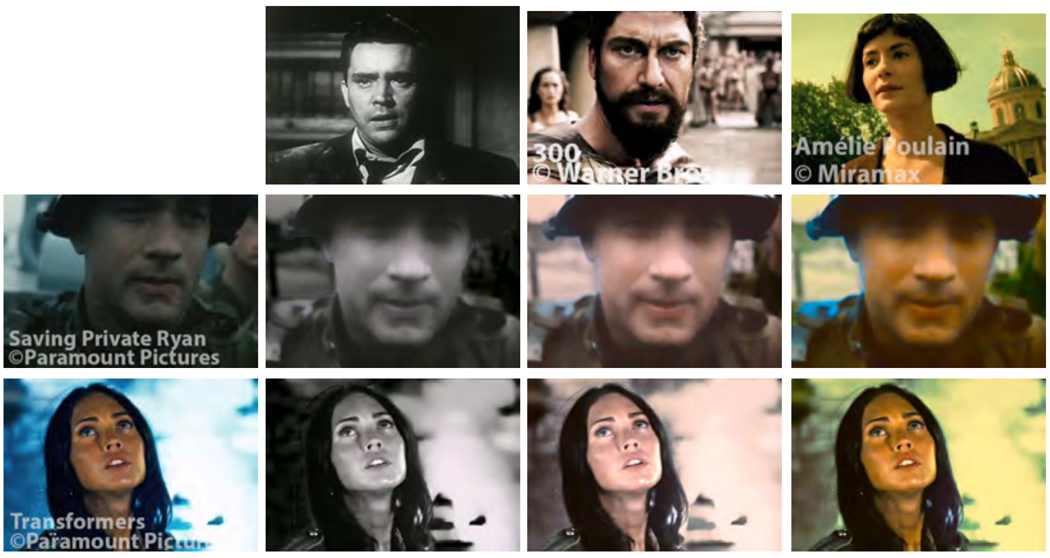
3.1.4 Results and Limitations

We show a subset of our results in Fig.3.7, but advise to appreciate temporal consistency in the supplemental videos. Results were obtained within minutes for 100 frames. In particular, our method is able to color grade videos that have already been stylized (see Fig. 3.8) – a process we call "color re-grading". Still, the (optional) semi-automatic segmentation [12] requires significant user input, and could benefit from the multi-label video segmentation technique we later developed [106]. Grayscale input video sequences produce rank-deficient matrices that result in artifacts. We refer the reader to video colorization techniques [192] for such applications.



(a) Input Video Frame (b) Color graded to "D.O.A" (Film Noir) (c) Color graded to "300" (sepia tones) (d) Color graded to "Saving Private Ryan" (bleach-bypass)

Figure 3.7: Our color grading method successfully transfers a variety of color styles to a range of video sequences. Here we demonstrate this for sepia tones, Film Noir, and bleach-bypass. In addition to capturing the color palette of the model videos, our results are temporally consistent.



(a) Color graded input video frame (b) Re-graded to “D.O.A” (c) Re-graded to “300” (d) Re-graded to “Amélie”

Figure 3.8: Our technique can take stylized video clips and change their visual look. We call this process “color re-grading”. Here we show results for re-grading (top to bottom) “Transformers” and “Saving Private Ryan” to the styles of the films (left to right) “D.O.A.”, “300”, and “Amélie”.

3.2 Intrinsic Videos

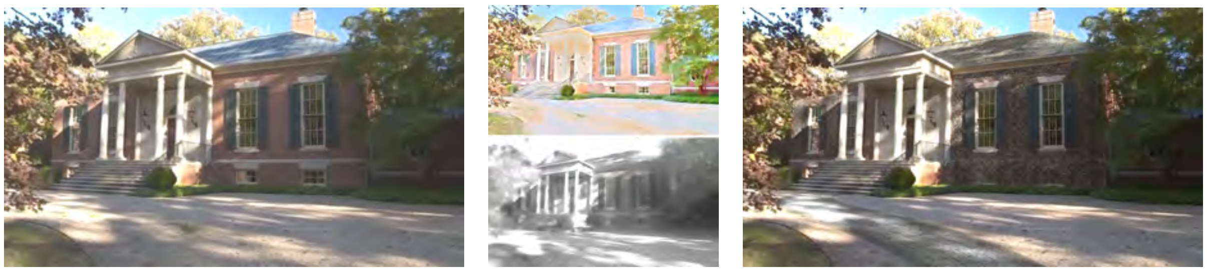

(a) Input video (b) Intrinsic decomposition (c) Editing the reflectance only

Figure 3.9: A video sequence (a) is interactively decomposed into temporally consistent components for reflectance (b, top) and illumination (b, bottom). Now, editing the textures in the reflectance image does not affect the illumination (c): changes to the brick walls, the roof tiles, and the pathway all maintain the complex illumination of the light through the trees.

A number of image editing operations, including recoloring, relighting, white balancing, and texture editing, require the materials properties of the objects in the photograph, and the illumination in the scene to be handled independently. This requires the separation of an image into a product of reflectance and illumination layers: the *intrinsic decomposition* problem. This is a challenging decomposition because the effects of reflectance and illumination are combined into a single observation, which makes the inverse problem of separating them severely ill-posed.

Significant progress has been made on this problem, involving sophisticated and slow algorithms and requiring scene-specific parameter tuning which make them unsuited to video. We have later thoroughly reviewed and evaluated them in a state-of-the-art report [26]. Recent work has extended these approaches to video [193], but, at the time of our 2014 paper [31], the computation time for short sequences were measured in hours. To achieve interactive editing, we need a significant increase in speed from minutes per frame to less than seconds.

We introduce a new algorithm designed for the fast intrinsic decomposition of videos that enables interactive reflectance and illumination editing. The centerpiece of our approach is a hybrid $\ell_2 - \ell_p$ formulation that enforces a smooth prior on illumination gradients and a sparse prior on reflectance gradients and is solved efficiently using look-up tables. We reconstruct the reflectance and illumination from these gradients by adding spatial and temporal constraints.

3.2.1 Efficient Intrinsic Decomposition

In this section, we describe our algorithm to efficiently decompose an input video I into an illumination layer S and a reflectance layer R. We assume that the illumination is monochromatic (i.e the illumination in the scene has a constant color). The observed intensity at pixel x is then given by:

$$\mathbf{I}(x) = S(x)\,\mathbf{R}(x), \tag{3.3}$$

where $\mathbf{I}$ and $\mathbf{R}$ are RGB-vectors and S is a scalar.

The core of our algorithm is a hybrid ℓ_2–ℓ_p energy formulation in the gradient domain that separates image gradients into reflectance and illumination gradients by enforcing a sparsity prior on reflectance values and a smoothness prior on illumination.

Hybrid ℓ_2–ℓ_p Gradient Separation

For the sake of efficiency, we perform most of the computation on single-channel luminance images. We define the image luminance as the geometric mean of the individual RGB components, i.e., $I = (\mathbf{I}_r\,\mathbf{I}_g\,\mathbf{I}_b)^{1/3}$, and reflectance luminance R, similarly, as the geometric mean of the reflectance components. This gives us the relation $I(x) = R(x)\ S(x)$, which we solve for R and S. Finally, the color reflectance $\mathbf{R}(x)$ can be estimated as $\mathbf{I}(x)/S(x)$.

We work in the log domain to transform the image formation into a sum: $\log I(x) = \log S(x) + \log R(x)$ and formulate our approach in the gradient domain. For this, we introduce lowercase variables to represent logarithmic gradients, e.g., $i = \nabla \log I$. We can then write: $i(x) = s(x) + r(x)$ and express this constraint as a least-squares energy term: $\|i(x) - s(x) - r(x)\|^2$.

We address the ill-posedness of this formulation by adding priors on s and r. We assume that reflectance values are sparse [126, 85], i.e., that scenes are mostly made up of objects of constant colors separated by hard boundaries. This is typically modeled using a ℓ_p term on the gradients with $p < 2$, i.e., with our notation: $\|r(x)\|^p$. Low p values assume very sparse reflectance gradients, while $p = 2$ would assume normally distributed reflectance gradients. We assume that illumination exhibits smoother variations [101], by using an ℓ_2 term on illumination gradients, i.e.: $\|s(x)\|^2$. This yields an energy E:

$$E(s,r) = \sum_x \|i - s - r\|^2 + \lambda_s \|s\|^2 + \lambda_r \|r\|^p \tag{3.4}$$

where λ_s and λ_r control the influence of the priors on s and r (Figure 3.11) ; dependencies on pixel x have been omitted for conciseness.

Minimizing the energy Solving mixed-norm optimization problems as Equation 3.4 often requires time-consuming combinatorial approaches. Inspired by Krishnan et al. [97], we show that we can minimize Equation 3.4 using a simple look-up table. Note that this formulation treats the horizontal and vertical components of the gradients independently. The formulations we derive below, can therefore be applied to each axis separately. Differentiating $E(s,r)$ with respect to s first gives an expression of s:

$$s \;=\; \frac{i - r}{1 + \lambda_s} \tag{3.5}$$

Substituting the expression for s from Equation 3.5 into Equation 3.4 gives us:

$$E(r) = \sum_x \frac{\lambda_s}{1 + \lambda_s} \|i - r\|^2 + \lambda_r \|r\|^p \tag{3.6}$$

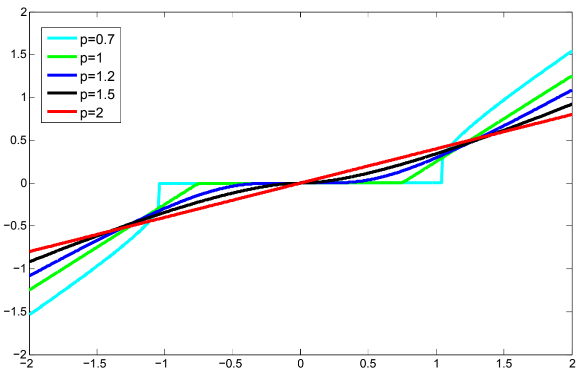
To minimize this expression, we use Iterative Re-weighted Least Squares to replace the ℓ_p term [24, § 4.5] with a weighted ℓ_2 term. This iterates solutions $\tilde{r}_k$ to quadratic minimization problems converging towards the optimal solution $r_\star$ of the mixed-norm problem (Equation 3.6):

$$\sum_x \frac{\lambda_s}{1 + \lambda_s} \|i - \tilde{r}_{k+1}\|^2 + \lambda_r w_{k+1} \|\tilde{r}_{k+1}\|^2 \tag{3.7a}$$

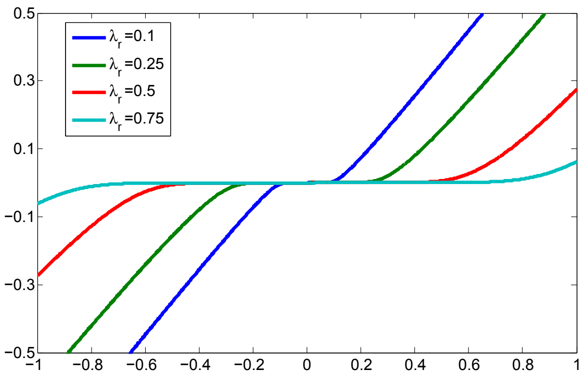
$$\text{with } w_{k+1} = \frac{p}{2} |\tilde{r}_k|^{p-2} \tag{3.7b}$$

Differentiating this cost function and setting it to 0 gives us a closed form recurrence:

$$\tilde{r}_{k+1} = \frac{2\lambda_s i}{2\lambda_s + \lambda_r (1 + \lambda_s) p |\tilde{r}_k|^{p-2}} \tag{3.8}$$

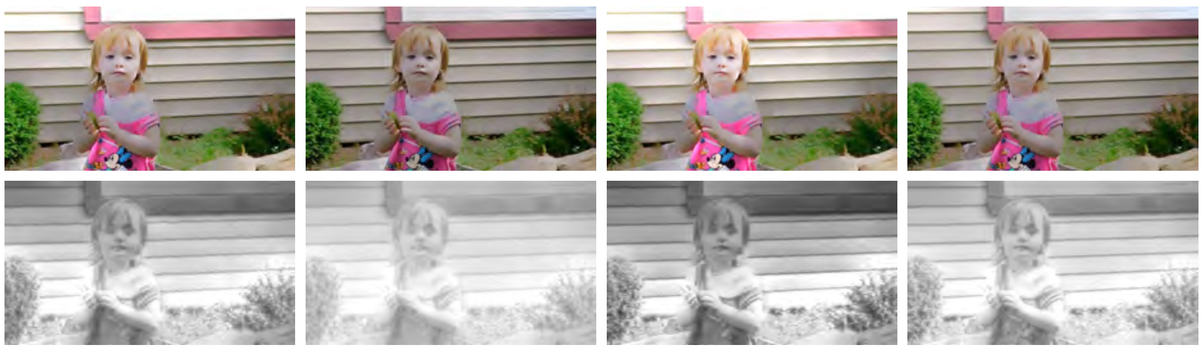


(a) Varying p
($\lambda_s = \lambda_r = 0.5$)



(b) Varying λ_r
($p = 1.1$ and $\lambda_s = 0.5$)

Figure 3.10: Behavior of $\text{lut}_{\lambda_s,\lambda_r,p}(\tilde{r}_k)$ (Eq. 3.8); $\tilde{r}_k$ varies along the x-axis. p controls the smoothness of the separation – at values closer to 1 it starts approximating a clipping function (a). Higher λ_r makes it more like a clipping function, while a larger λ_s makes it smoother.



(a) Baseline decomposition (b) Increased ℓ_2 weight (c) Increased ℓ_p weight (d) Increased p

Figure 3.11: Effect of the different parameters in Equation 3.8 on the reflectance (top) and illumination (bottom), without non-local terms and nor color handling. Increasing the ℓ_2 weight (b) leads to a blurry illumination, but the reflectance is not sparse; increasing the ℓ_p weight (c) pushes stronger gradients to the illumination component. A denser $p = 1.2$ norm (d) makes the decomposition smoother albeit with less separated gradients than a sparser $p = 0.4$ norm.

In practice, given i, λ_s, and λ_r, we iterate Equation 3.8 K times (typically, $K = 100$). Further, given λ_s and λ_r, the output r value depends only on i. When λ_s and λ_r are fixed, we use this property to precompute a look-up table $\text{lut}_{\lambda_s,\lambda_r}(\tilde{r}_k)$ for varying values of $\tilde{r}_k$. We can also build a higher-dimensional table to enable varying parameters.

Taking chrominance variations into account The energy function in Equation 3.4 encourages sparse but large reflectance gradients and dense but smaller illumination gradients. This does not account for illumination effects like shadows that can lead to large gradients. However, shadows typically do not give rise to large variations in *chrominance* values; these are more likely to be caused by changes in reflectance [76]. Similar to them, we set the shading gradient to zero whenever the chrominance of the image gradient is above a user-specified threshold T^{chr}.

Discussion Figure 3.10 illustrates the form the function $\mathrm{lut}_{\lambda_s,\lambda_r}(\tilde{r}_k)$ takes for different values of λ_s, λ_r, and p. For $p \approx 0.5$, this function approximates the thresholding function that the Retinex algorithm uses to separate image gradients into reflectance and illumination gradients. This suggests that our look-up table is a generalization of the Retinex algorithm. However, unlike the Retinex algorithm, our look-up table is a softer function that allows for non-zero reflectance and illumination gradients at the same pixel. As shown in Figs. 3.15 and 3.12, this leads to higher quality results at depth discontinuities.

Reconstructing the Layers

For given values of λ_r and λ_s, we precompute $\mathrm{lut}_{\lambda_r,\lambda_s}(i)$. Then, for each pixel, we estimate the horizontal and vertical gradients of r by applying $\mathrm{lut}_{\lambda_r,\lambda_s}$ for each axis. We use Equation 3.5 to recover s. Then, we solve for S'_t by minimizing $E_p(S'_t) = \sum_x \|\nabla S'_t(x) - s_t(x)\|^2$ and exponentiate the result to get the illumination layer $S = \exp(S')$. Finally, we estimate the color reflectance layer $\mathbf{R}$ using the image formation model (Equation 3.3).

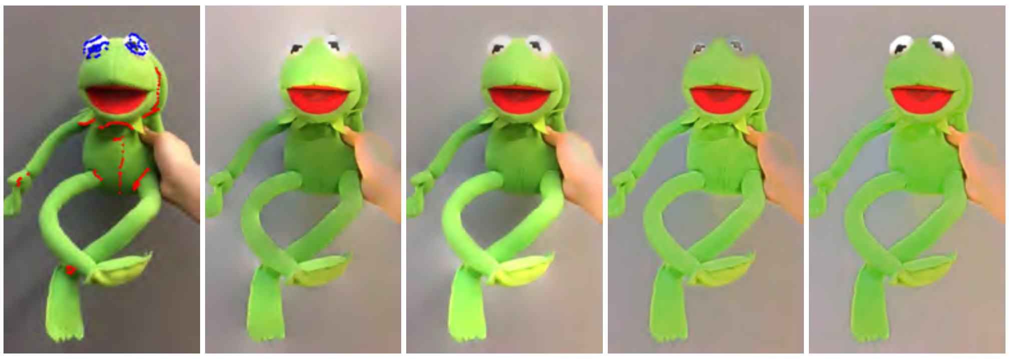
Spatial reflectance constraints Reconstructing the reflectance and illumination from gradients can lead to results with low-frequency errors. Previous work has addressed this by incorporating non-local reflectance constraints, typically by enforcing that pixels with similar chrominances have similar reflectances [169, 194]. We propose a faster approach that only incorporates a few non-local constraints that add negligible overhead to the linear system we solve. We cluster the video into a set of C chrominance clusters using k-means, and for every frame, we select a representative pixel of each cluster. We then add a non-local penalty term to our energy enforcing the reflectance of every pixel to remain close to the reflectances of the N nearest cluster representatives weighted by a function w of its distance in chrominance space:

$$\begin{aligned} E_{nl}(S'_t) &= \sum_x \sum_{n=1}^{N} w(x,x_n)\|\log R_t(x) - \log R_t(x_n)\|^2 \\ &= \sum_x \sum_{n=1}^{N} w(x,x_n)\|S'_t(x) - S'_t(x_n) - \log I_t(x) + \log I_t(x_n)\|^2, \end{aligned} \tag{3.9}$$

where x_n is the representative coordinate of the n^{th} nearest cluster in chrominance space. We adopt the definition of the chrominance as $\mathbf{C}_k = \log(\mathbf{R}_k) - \sum_k \log(\mathbf{R}_k)$ for the three color channels [76]. In our implementation, $C = 8$ and $N = 2$, and $w(x,x_n) = \exp(-\|\mathbf{C}(x) - \mathbf{C}(x_n)\|^2/100)$. This adds only $N = 2$ non-zero terms to the linear system for every pixel.

Temporal constraints Applying the previous technique frame by frame can lead to results with unpleasant flickering artifacts. However, the reflectance of a scene is typically temporally coherent and we would like to enforce this in our optimization. While solving for the intrinsic decomposition at a particular frame, we add a temporal regularization that keeps the reflectance close to the computed reflectance from the previous frame. In addition to being fast, solving for the decomposition in chronological order is a user-friendly option because it ensures that frames that have already been annotated by the user are not affected by subsequent strokes.

Denoting $u_t(x_t)$ the forward optical flow, i.e., the pixel x_t at frame t moves to $x_{t+1} = x_t + u_t(x_t)$



(a) Input frame and scribbles (b) Retinex (d) $\ell_2 - \ell_p$ term function (e) Non-local term added (f) User input added

Figure 3.12: Effect of the different terms in our optimization. The hard thresholding of Retinex-based methods (b) leads to aliasing artifacts at object boundaries. Our $\ell_2 - \ell_p$ decomposition is smoother (c) especially at object boundaries. Adding the non-local constraints reduces low frequency artifacts (d). The user can refine the result using a few strokes (e); constant reflectance scribbles are shown in red, constant illumination in blue.

at frame $t+1$, we define a temporal regularization term:

$$\begin{aligned} E_{t+1}(S'_{t+1}) =& \sum_x \|\log R_{t+1}(x_{t+1}) - \log R_t(x_t)\|^2 \\ =& \sum_x \|S'_{t+1}(x_{t+1}) - S'_t(x_t) - \log I_{t+1}(x_{t+1}) + \log I_t(x_t)\|^2. \end{aligned} \tag{3.10}$$

Multi-scale solver We combine all terms to obtain $E(S'_t) = E_p(S'_t) + \lambda_{nl} E_{nl}(S'_t) + \lambda_t E_t(S'_t)$, where λ_{nl} and λ_t control the weights given to the non-local and temporal constraints. λ_t is modulated by a per-pixel optical flow confidence. We solve for the illumination that minimizes this energy at every time instant, starting with the first frame and moving onto each subsequent frame. Minimizing this energy leads to a sparse linear system with only $N + 4$ off-diagonal non-zeros per pixel. We use parallelized Jacobi iterations on a coarse-to-fine multi-resolution pyramid.

User strokes We provide users with a scribble tool to specify constraints in the *gradient* domain, ultimately only altering the right hand side of existing equations. We use two strokes, for *constant reflectance* and *constant illumination*, that specify that the gradient of the reflectance (and illumination, respectively) at those points is 0. These constraints are straightforward to apply to s and r, and do not require us to perform the gradient separation again. To speed up user interaction, we propagate user strokes both spatially and forward in time to similar pixels using a fast coherency-sensitive hashing [96] (CSH) using the LSHKIT library [54] and feature vectors estimated via a Principal Component Analysis (PCA) of 11×11 patches.

3.2.2 Results and Discussion

In this section, we evaluate our algorithm on both single images and video sequences and compare with state-of-the-art techniques. The quality of our results is better evaluated on the video accompanying our paper [31]. In our prototype implementation, we let the user interactively adjust the parameters p, λ_r, λ_s, T^{chr}, λ_t and λ_{nl}. Typical ranges are $p \in [0.4, 0.5]$, $\lambda_r \in [0, 0.5]$, $\lambda_s \in [0, 10]$, $\lambda_{nl} \in [0, 1]$, $T^{\text{chr}} \in [0, 0.2]$. We precompute the optical flow using the method of

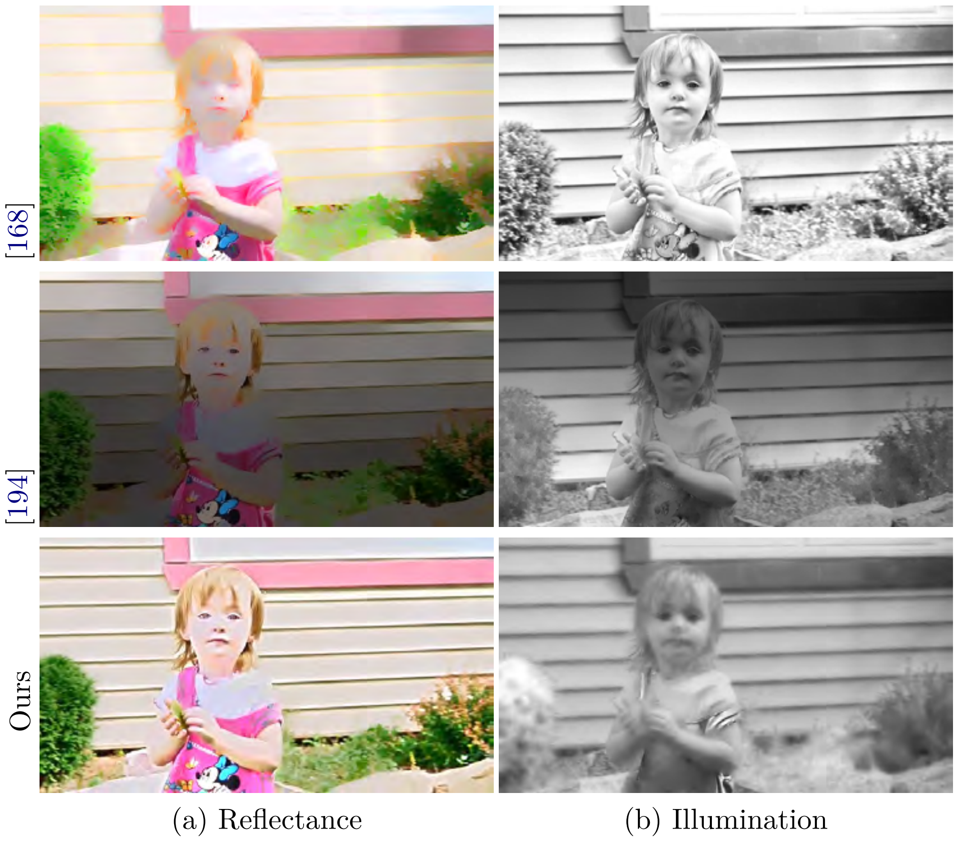


Figure 3.13: We compare our technique (third row) to the decompositions of Shen et al. [168] (first row) and Zhao et al. [194] (second row). Shen et al. has brushes and takes 6 min. while Zhao et al. is automatic and takes 6 s. Our method takes 0.5 s per frame, and provides a more satisfactory decomposition for our applications.

Liu et al. [111] which takes 0.25-0.50s to process a 0.5MP video frame, and propagate the user constraints forward by 12 frames using background threads.

Figure 3.12 evaluates the different terms in our decomposition algorithm. We observe that non-local terms remove low-frequency errors, temporal constraints lead to time-coherent results, and user input can resolve other ambiguities. Importantly, every video is different and might need slight variations of these terms. By making the solver interactive, we allow users to quickly adapt the result to their requirements.

Static images Figure 3.13 compares to the work of Shen et al. [168] and Zhao et al. [194]. A few strokes are sufficient to create comparable results, but at interactive speeds. Our subsequent state-of-the-art report [26] evaluates our method against numerous more recent approaches. Our work [26] also provides a new dataset of 45 realistic synthetic scenes[1] under various illumination conditions resulting in 149 renderings – a database that we call ARAP for "As Realistic As Possible" dataset. It provides, along with the synthetic rendering, a ground truth reflectance, illumination, depth map, segmentations, and normal map among other data. Sample images are shown in Fig. 3.14. Unfortunately, this dataset was not available at the time of this intrinsic video decomposition project.

[1]https://perso.liris.cnrs.fr/nicolas.bonneel/intrinsicstar/ground_truth/

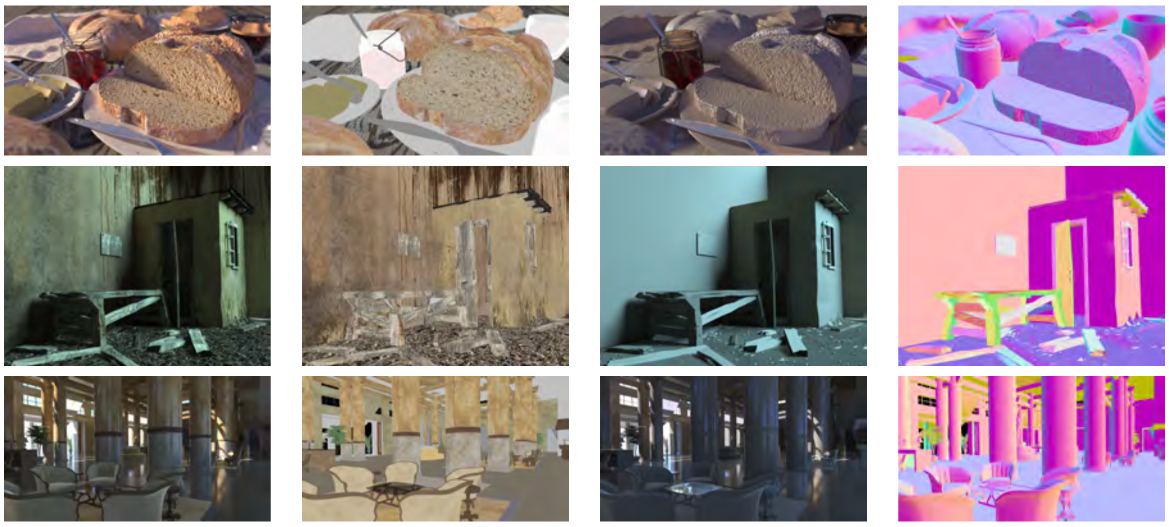


Figure 3.14: Our ARAP dataset [26] contains (from left to right), among other things, a photorealistic synthetic rendering, ground truth reflectance and shading, and other data such as a normal map, for 45 scenes under different lighting conditions for a total of 149 photorealistic renderings. It also illustrates the difficulty in defining a ground-truth decomposition in complex scenes.

Sequence	Figure	# Frames	Resolution	Solver (per frame)	# Strokes	Interaction
House	3.9	91	1024×576	0.60 s	325	20 min
Kermit	3.12 (f)	291	320 ×568	0.18 s	101	6 min
San Miguel	3.15 (d)	100	1280×960	1.30 s	32	15 min
Kid	3.16 (top)	285	960 ×540	0.51 s	10	6 min
Chicken	3.17 (c)	148	640 ×360	0.24 s	0	0 min
Girl	3.19	98	640 ×360	0.24 s	276	15 min
SIGGRAPH Cart	Video (3'24)	141	1024×576	0.59 s	0	0 min
Shadow art	3.20	134	1024×576	0.62 s	28	10 min
Compositing (background)	3.21 (a)	215	1024×576	0.60 s	330	20 min
Compositing (foreground)	3.21 (b)	107	1024×576	0.62 s	207	20 min

Table 3.1: Statistics for our input videos. We report approximate interaction times and the number of strokes for each complete video sequence.

Videos First, we evaluate our technique on a realistically-rendered [137] animation of the San Miguel 3D scene (Figure 3.15), 100 frames at 1280×960. This complex sequence is a good approximation of a real-world example, with the advantage of providing a ground-truth decomposition. As shown in Figure 3.15), our technique produces a result that compares well with the ground truth and is significantly better than the state-of-the-art single image technique of Zhao et al. [194]. Additionally, we compare our method with the concurrent work of Ye et al. [193] on a real-world video sequence in Figure 3.16 and an animated video in Figure 3.17. Like us, they incorporate user input to disambiguate challenging areas, but unlike us, this user input is specified on the first frame and the solution is propagated to the rest of the video. Also, their method runs in the order of hours, precluding any interactive refinement. Please refer to our paper [31] for a numerical evaluation of temporal consistency.

Discussion We believe performance could be further improved by implementing our algorithm on graphics hardware. Due to the nearest-neighbor search, our algorithm could slow down if too many strokes are specified. In practice, the examples we show were edited with less than 20

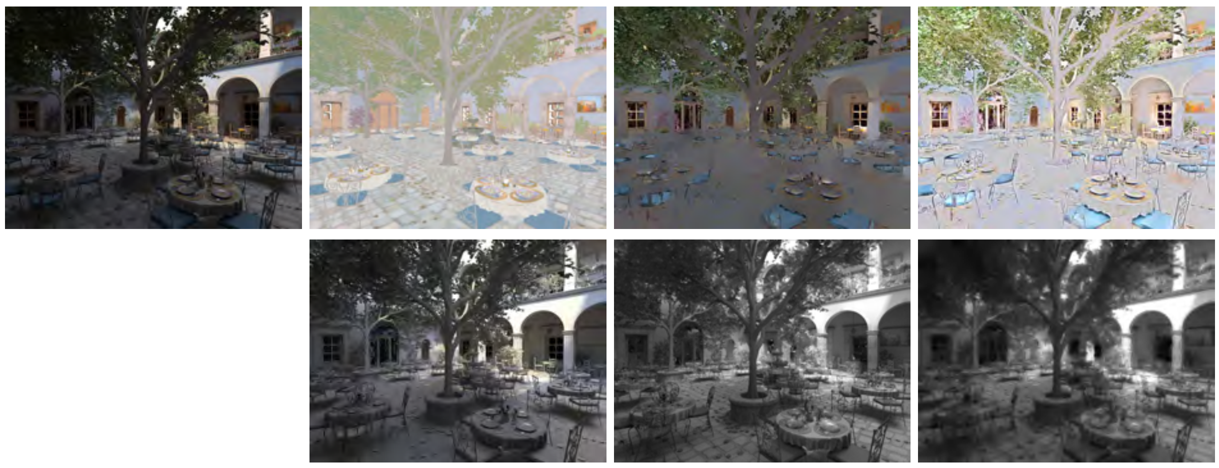


Figure 3.15: We demonstrate our technique on one frame of a rendered video sequence (a) with ground truth reflectance and illumination (b). Using the single image method of Zhao et al. [194] per frame produces a result that is spatially inaccurate and temporally inconsistent (c).

minutes of user interaction and sometimes, were produced with no interaction at all. Table 3.1 contains detailed statistics about our input videos, and computation and interaction times that were required to produce our results.

Our tool assumes monochromatic lighting for efficiency. However, in presence of illumination with spatially varying colors, or strong colored inter-reflections, this can lead to artifacts (e.g., see the shirt in Fig.3.18, and the pink inter-reflections on the girl's arms in Fig. 3.19).

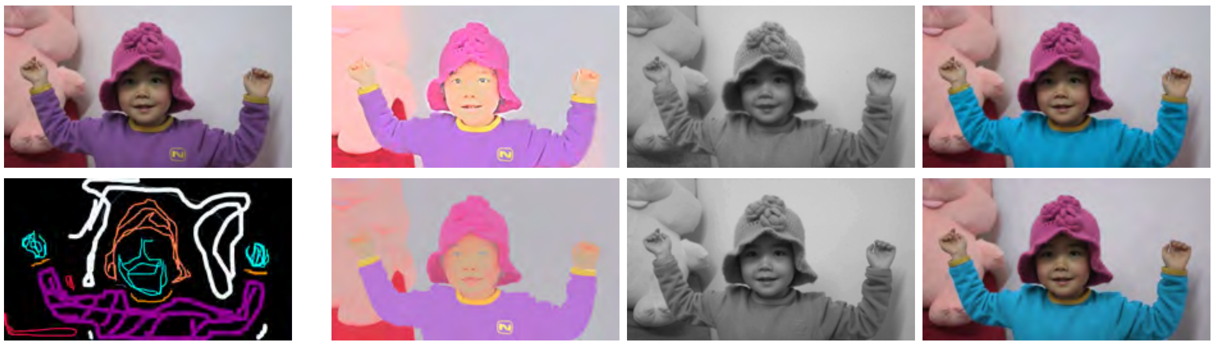
Our $\ell_2 - \ell_p$ model assumes that illumination is smoothly varying; this assumption can be violated at object boundaries where our technique produces an overly smooth shading layer. This can be observed in Fig. 3.13 where our illumination is smoother than other methods. Generally, the decomposition obtained using Ye et al.'s slower approach [193] with user interaction is more accurate when compared to our method without any user interaction.

3.2.3 Applications

We demonstrate our method on various applications benefiting from an editable and temporally consistent intrinsic decomposition.

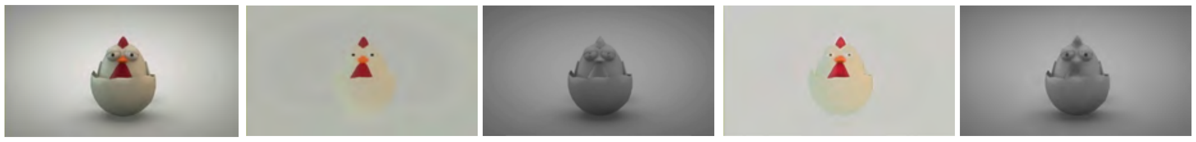
Reflectance editing Editing the reflectance of an object in a video is easy when the color is uniform; in such cases, a simple chrominance change suffices. However, this becomes laborious in the presence of more complex textures that are also visible in the luminance. With our decomposition, painting in the reflectance layer performs the desired operation since illumination remains unchanged. We demonstrate this in Figures 3.9 and 3.19 (top).

Illumination editing Turning hard shadows into soft shadows is a challenging operation if one only has access to raw data since blurring the shadows ends up blurring the textures in the



(a) Input video / Ye et al. scribbles (b) Reflectance (c) Shading (d) Recoloring

Figure 3.16: Comparison of our approach *(top)* to Ye et al. [193] *(bottom)*. Both require user input, with Ye et al.'s all coming on the first frame (a). Their method requires four hours to compute, while our approach only takes two minutes and gives immediate feedback to user scribbles. This allows to easily correct the decomposition such as the yellow logo on the shirt.



(a) Input video frame (b) Ye et al. [193] (with user strokes) (c) Our result (without user strokes)

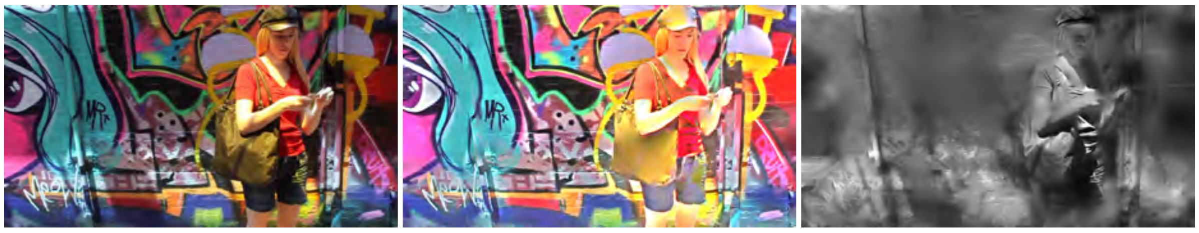
Figure 3.17: On this animated video sequence (a), our technique is able to, without any user strokes, produce a comparable result (c) to the user-assisted result of Ye et al. [193] (b).

scene too. In comparison, with our decomposition, this task becomes straightforward since one needs to blur the illumination layer (Fig. 3.20).

Lighting-consistent video compositing Naively compositing videos that contain shadows produces unsightly results in which the shadows overlap instead of merging. Our decomposition enables the proper merging. Denoting S_1 and S_2 the shading layers of the two videos to be composited, we compute $\min(S_1, S_2)$ as the new shading layer within a segmentation of the foreground video obtained with Video Snapcut for instance [12] (Fig. 3.21).

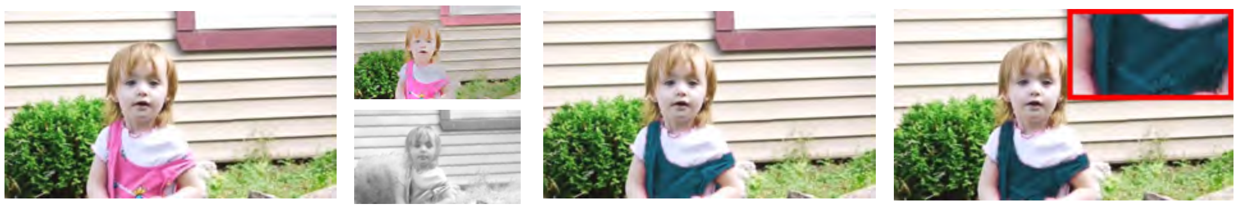
3.2.4 Conclusions

In this work, we have presented the first interactive intrinsic decomposition algorithm for video sequences. Our technique is built on top of a hybrid $\ell_2 - \ell_p$ algorithm that efficiently computes reflectance and shading gradients from a image. We present a fast solver that combines these gradients with spatial and temporal constraints to reconstruct temporally consistent shading and reflectance videos at interactive rates. The interactive nature of our solver is a significant contribution over previous work, making it possible for users to navigate the parameter space of the algorithm and make annotations while receiving immediate feedback. Our technique can be used to efficiently derive high-quality intrinsic decompositions for a wide variety of real-world video sequences. Most importantly, our decomposition enables a number of video editing tools that are otherwise difficult to implement; these include recoloring, retexturing, illumination editing, and lighting-consistent video compositing.



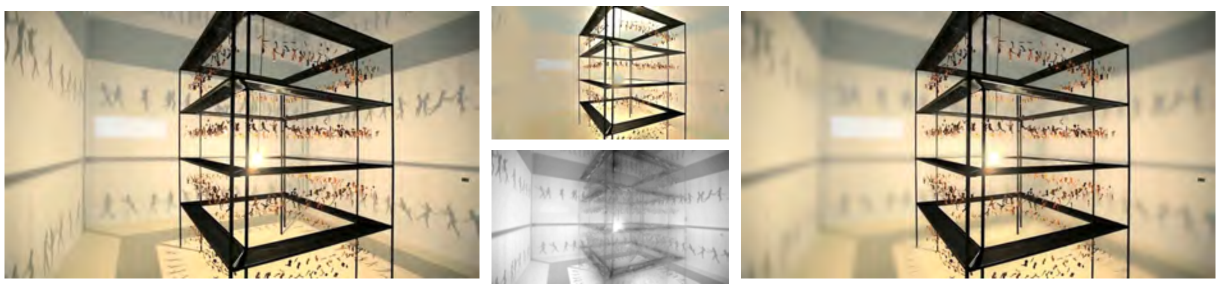
(a) Input frame (b) Albedo (c) Shading

Figure 3.18: Our method assumes monochromatic lighting. The presence of chromatic lighting or strong interreflections that produce colored shading violate our model, and can produce significant artifacts that cannot be corrected, even with heavy user interactions.



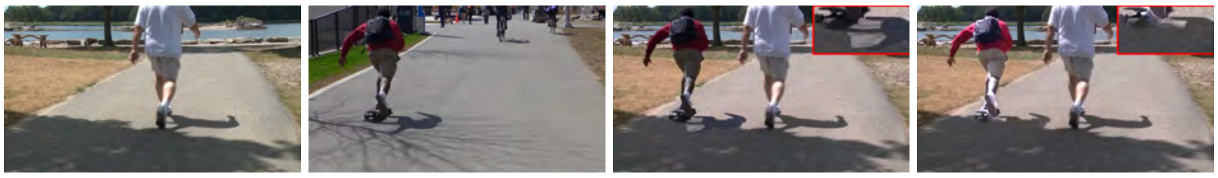
(a) Input video sequence (b) Intrinsic decomposition (c) Reflectance-edited video (d) Naive chrominance editing

Figure 3.19: Our decomposition enables easy reflectance editing such as re-coloring the girl's t-shirt. By separating the two components, we are able to make these edits while retaining the original illumination in the video. A naive editing of the chrominance leaves luminance variations due to the illustration on the shirt.



(a) Input video sequence (b) Intrinsic decomposition (c) Shadow-edited video

Figure 3.20: Lighting editing. After intrinsic decomposition, we blur the illumination component, and recombine the original reflectance and the blurred illumination to produce a video sequence with soft shadows.



(a) Background video frame (b) Foreground video frame (c) Naive compositing (d) Our compositing

Figure 3.21: Video compositing. Here, we decompose two video sequences shot with different viewpoints (a, b). Compositing the two reflectance and illumination layers separately and combining them allows us to create a video composite with realistic shadows and lighting (d). A naive compositing produces inconsistent shadows (c).

3.3 Blind Consistency for Videos and Camera Arrays

Upon developing algorithms tailored for color processing (chapter 3) and then intrinsic decomposition (section 3.2), the need for a general-purpose method became obvious. This section describes an approach to make various image processing filters temporally consistent, which considers the image processing filter as a black box. We hence called this class of algorithms *blind*. We developped this algorithm both for single- [33] and multi-camera [32] video sequences, and base our method on a gradient domain formulation in the spirit of our video intrinsic decomposition (section 3.2).

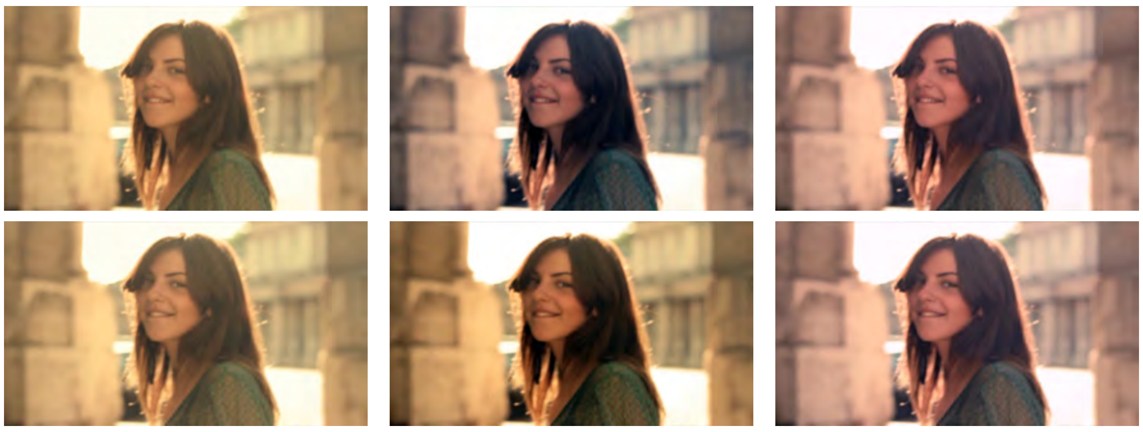


Figure 3.22: With many image filters, such as this automatic color, tone, and contrast adjustment, processing an input video (a) (frames 167-168) frame by frame results in temporal discontinuities (b). We take the two video sequences (a) and (b) and automatically generate a temporally consistent video (c), without knowing the image filter used to produce the unstable video (b). This enables the application of our technique to a wide range of video effects such as color constancy, stylization, color grading, intrinsic decomposition, depth prediction, and dehazing.

With advances in processing, effective filters for restoration, enhancement, creative edits, and analysis are now common for static images. Videos, on the other hand, do not enjoy the same rich and diverse toolbox. Several video processing algorithms have been developed along this line, such as color grading [30], dynamic range compression [11], intrinsic decompositions [193, 31, 94], and tonal stabilization [60]. While effective, these techniques are specific to each task and do not generalize to other problems. Paris [130] and Lang et al. [102] propose more generic approaches to extend still image operators to videos. However, these continue to assume a specific filter formulation, which limits their application. For operators outside this set, Lang et al. resort to temporal low-pass filtering. As we shall see, this reduces flickering but does not fully remove it.

We aim for a more general approach to extending image filters to videos, and propose an algorithm that is agnostic to the internal design of the filter, i.e., we treat image filters as black boxes that take input frames and generate processed frames. In that sense, our approach to temporal consistency is *blind* to the image filter being applied. We have two requirements: 1) that the original video is available and that optical or feature flow is recoverable, such that it can be used as a temporal consistency guide, and 2) that the filter does not generate new content uncorrelated with its input — for instance, inpainting techniques that synthesize new content is outside our scope. That said, our approach covers a wide variety of filters such as dehazing, automatic photographic enhancement, color grading, and intrinsic decomposition. We formulate our algorithm in the gradient domain and propose an energy function that amounts to a spatial screened Poisson equation with temporal constraints that we can solve efficiently. We formally characterize the properties of our approach in terms of frequency content.

Further, capturing video with multiple cameras is an integral component of many popular applications. For example, camera array rigs record content for 3D displays, for stitched 360° video for virtual reality [134], for virtual-camera-based post production [65], and for refocusing and depth-aware processing [125, 183]. This makes filtering challenging, to recover consistency both across cameras and in time, and to make this process efficient. Camera arrays often produce 10–100× more video data than single-view cameras, and so *any* kind of filtering is expensive, both in terms of computation and memory. As an example, a 91-view sequence at 1280×768 resolution contains about 8GB of data *per second.*

We present an efficient approach that reduces the cost of filtering the data by an order of magnitude, without a large reduction in quality, by skipping the filtering step on many frames completely, and instead, *transferring* the filter response from a small subset of input frames to the rest of the data. Further, our approach is causal as it sequentially processes frames, and so does not need multiple passes, or even to keep multiple views or entire videos in memory. Combined, these properties reduce the computational burden and allow for of spatio-temporally consistent filtering of camera array video.

3.3.1 Restoring Single Video Temporal Consistency

Our algorithm takes as input an unprocessed video $\{V_0, V_1 \ldots\}$ and the same video $\{P_0, P_1 \ldots\} = \{f(V_0), f(V_1) \ldots\}$ processed frame by frame by some image filter f. The filter f has introduced temporal artifacts that we seek to remove to create a temporally stable video $\{O_0, O_1 \ldots\}$.

Spatially, the artifacts in P can be either global or local. For instance, intrinsic image decompositions are defined up to a global multiplicative factor and algorithms often set this factor arbitrarily, leading to random offsets in each frame. Algorithms that rely on sophisticated optimization schemes are prone to local minima, which makes them overly sensitive to initial conditions and can generate discontinuous local variations between adjacent frames. Further, many optimization schemes are spatially regularized, so variations typically impact an entire object or image region at once — they rarely occur at the scale of a few pixels. In the temporal domain, the profile of these artifacts is arbitrary: they can vary slowly, be random at each frame, or be anywhere in between. We design our algorithm with these characteristics in mind.

One naive approach would be to enforce perfect temporal consistency by warping the first frame by optical flow to recreate each subsequent frame. However, this ignores the inevitable imprecision of accumulated flow fields, and would eventually cause large errors. Further, this scheme does not account for issues like occlusions and appearance variations, e.g., due to lighting changes. In other words, enforcing temporal consistency can come at the expense of scene dynamics. Our solution balances these two aspects.

Joint optimization of temporal consistency and scene dynamics

We describe our approach in a causal setting: we consider the n^{th} frame $(n > 1)$ assuming that the previous frames have already been processed. This processes frames one at a time, which keeps the memory requirement small and enables the processing of arbitrarily-long videos without resorting to complex memory management schemes [130].

We formulate the temporal consistency objective with a simple least-squares energy: $\text{argmin}_{O_n} \int \|O_n - \text{warp}(O_{n-1})\|^2 \text{d}x$, where x represents the spatial position in the frame, and warp() uses backward flow to advect the previous frame toward the current frame.

For the scene dynamics term, one naive option would be a simple data attachment term: $\operatorname{argmin}_{O_n} \int \|O_n - P_n\|^2 \mathrm{d}x$. However, P implicitly suffers from temporal inconsistency, and so this term would go against our objective and transfer instability to the output video O. Ideally, we would like to attach O only to the part of P that represents the scene and discard the part that is inconsistent. To achieve this, we draw inspiration from the work of Elder [59], who showed that a scene is well represented by its edges. We are also inspired by Poisson Image Editing [135], which reproduces the appearance of image regions by copying their gradients. Thus, instead of requiring pixel values to be similar, we require their gradients to be similar. We minimize: $\operatorname{argmin}_{O_n} \int \|\nabla O_n - \nabla P_n\|^2 \mathrm{d}x$. Intuitively, this can be seen as a data attachment on the scene edges, where the gradients are approximations of the edges. We further analyze this aspect in Section 3.3.1.

We combine the two terms after modulating the influence of the temporal consistency term by a weight, $w(x)$, that measures the input video consistency V. We set the first frame as a boundary condition and compute O as the minimum of the least-squares energy:

$$\int \|\nabla O_n - \nabla P_n\|^2 + w(x)\|O_n - \operatorname{warp}(O_{n-1})\|^2 \mathrm{d}x \tag{3.11a}$$

$$\text{with:} \quad w(x) = \lambda \exp(-\alpha \|V_n - \operatorname{warp}(V_{n-1})\|^2) \tag{3.11b}$$

$$O_0 = P_0 \tag{3.11c}$$

The weighting function $w(x)$ is key to our approach because it relaxes the temporal consistency requirement when the input video V itself is not consistent or the warp inaccurate, which we detect when the warped previous frame does not explain well the current frame (Eq. 3.11b). In other words, we only impose temporal consistency when the input video is consistent. We use the first frame as a boundary condition (Eq. 3.11c) which corresponds to the common scenario where users tune the image filter f to achieve the desired result on the first frame and would like to propagate the same quality of results to the rest of the video. It would be straightforward to adjust our scheme to use any frame as reference and to process the video forwards and backwards in time from it. Our formulation is parametrized by λ, which controls the regularization strength (see analysis Sec. 3.3.1) and α which indicates our confidence in the warp.

To minimize Equation 3.11a, we use the Euler-Lagrange formula [189] to derive a differential property that O_n must satisfy at the minimum:

$$-\Delta O_n + w(x) O_n = -\Delta P_n + w(x) \operatorname{warp}(O_{n-1}) \tag{3.12}$$

Frequency-domain Analysis

Our approach has a varying impact upon different spatial frequencies in the video, with the low frequencies being more constrained to be temporally consistent than the high frequencies. We analyze Equation 3.12 in the frequency domain using $\mathcal{F}(\cdot)$ for the Fourier transform and ξ for the spatial frequency. For the sake of simplicity and in this section only, we assume that the weighting function w is constant and equal to λ_0, i.e., $w(x) = \lambda_0$ for all x. Applying the identity $\mathcal{F}(\Delta\cdot) = -4\pi^2\xi^2\mathcal{F}(\cdot)$ to Equation 3.12 gives:

$$4\pi^2\xi^2\mathcal{F}(O_n) + \lambda_0\mathcal{F}(O_n) = 4\pi^2\xi^2\mathcal{F}(P_n) + \lambda_0\mathcal{F}(\operatorname{warp}(O_{n-1}))$$

which leads to:

$$\mathcal{F}(O_n) = \frac{4\pi^2\xi^2\mathcal{F}(P_n) + \lambda_0\mathcal{F}(\operatorname{warp}(O_{n-1}))}{4\pi^2\xi^2 + \lambda_0} \tag{3.13}$$

which is the basis of our analysis.

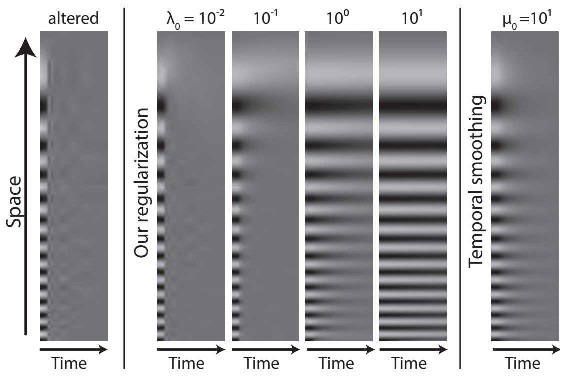


Figure 3.23: We perform a synthetic experiment on a 1D video: each frame is a vertical line of pixels. The first frames show a linear chirp, i.e., a signal of increasing spatial frequency, and the remaining frames are uniformly gray. With a low λ_0 value, there is no temporal consistency and the output video transitions instantly back to a uniform color. For intermediate values, the temporal consistency is enforced more strongly on the low frequencies which remain visible for a long time. For a large λ_0 value, temporal consistency is enforced on the whole spectrum and the chirp is propagated to all frames (see § 3.3.1). *Right:* In contrast, temporal smoothing enforces consistency uniformly over the spectrum.

First, we look at the effect of λ_0. For large values, i.e., $\lambda_0 \to +\infty$, we have $\mathcal{F}(O_n) \approx \mathcal{F}(\text{warp}(O_{n-1}))$, that is, we warp the previous frame — this would not account for scene dynamics. For small values, i.e., $\lambda_0 \to 0$, for $\xi \neq 0$, we have $\mathcal{F}(O_n) \approx \mathcal{F}(P_n)$, that is, we copy the frequency content of the frame processed by the image filter without imposing any temporal consistency. The DC component ($\xi = 0$) is treated differently though. If $\lambda_0 = 0$, we have a $0/0$ indeterminacy that corresponds to the well-known ill-posedness of the Poisson equation in the absence of boundary conditions. More interestingly, if $\lambda_0 \neq 0$, we have $\mathcal{F}(O_n)(0) = \mathcal{F}(\text{warp}(O_{n-1}))(0)$, i.e., we copy the DC component of the previous frame. That is, even with a small temporal weight, as long as it is non-zero, our approach removes the flickering due to a constant spatial offset.

Next, we assume a general non-zero value of λ_0 and analyze the influence of the unstabilized filtered frame P_n vis-a-vis the stabilized previous frame O_{n-1}. The influence of P_n is proportional to the square of the frequency ξ. As a result, the low frequencies of the stabilized frame O_n are dominated by the previous frame O_{n-1}, but the high frequencies are closer to those of the output P_n of the image filter. This means that the temporal consistency is more strongly enforced on low frequencies. We illustrate this property on a synthetic example in Figure 3.23. Recall that our goal is to remove temporal inconsistencies that are mostly constant or large-scale while preserving the scene structure captured by the image edges. By enforcing temporal consistency more strongly on the low frequencies and preserving the high frequencies, our approach fulfills our objective.

In comparison, temporal smoothing can be modeled as averaging the current frame with the previous frame, i.e., $O_n = (P_n + \mu_0 \,\text{warp}(O_{n-1}))/(1 + \mu_0)$ where μ_0 controls the smoothing strength [130]. This directly translates to $\mathcal{F}(O_n) = (\mathcal{F}(P_n) + \mu_0 \mathcal{F}(\text{warp}(O_{n-1})))/(1 + \mu_0)$. All frequencies are affected equally by the previous frame, i.e., temporal consistency is enforced uniformly over the spectrum.

Our paper [33] experimentally analyzes the behavior of our temporal smoothing over various spatial scales of synthetic random noise, and shows that even extreme noise can be made temporally consistent if it is of spatially low-frequency.

3.3.2 Single-Video Results

This section provides implementation details, comparisons with state-of-the-art techniques, and applications of our technique to several video processing applications.

Implementation

The choice of the warp() operator in Equation 3.11a is critical, as inaccurate correspondences across the input video V can result in flickering or bleeding in the stabilized result O. After testing several optical flow techniques [111, 190, 177], we found that the method of Sun et al. [177] produced satisfactory results on a wide range of videos, despite occasionally introducing minor bleeding. Its main drawback is computational cost, taking 1–2 hours for 100 frames at 1024×576 resolution. We also considered several nearest neighbor field techniques [13, 21, 78]. PatchMatch [13] provides a complementary option which generates satisfactory results on many examples, including on videos which are challenging for optical flow, and at a fraction of the cost: less than 30 seconds for 100 frames. However, PatchMatch sometimes introduces minor flickering when the estimated correspondence field is discontinuous between two successive frames. In general, both methods were able to produce high-quality results: videos with rapid motion are often better handled with PatchMatch, while applications which remove texture cues such as depth prediction and intrinsic decomposition are better with optical flow. We generated the results in Figures 3.22, 3.24, 3.26, 3.27, and 3.28 using PatchMatch, and optical flow was used for 3.25, and 3.38.

Having pre-computed correspondences which define the warp() operator, we solve the linear system of Equation 3.12 using a fast multiscale solver with Gauss-Seidel iterations. Our approach takes less than 0.40 seconds per frame at 1024×576 resolution. The temporal weight (Eq. 3.11b) is computed using $\alpha = 0.2$ for all our experiments. For λ, we start with $\lambda = 1.0$, which works in about 75% of cases. We reduce it when we observe bleeding due to optical flow inaccuracy. In practice, a λ value between $[0.05; 1.0]$ produced results without spatial bleeding nor flickering.

Comparisons

We compare our approach to the unaltered filtering algorithm of Lang et al. [102] on a typical automatic color and tone enhancement task, using a combination of Adobe Photoshop's 'Auto Color', 'Auto Contrast', and 'Auto Tone' tools. Applying these tools on a per-frame basis results in strong high-frequency flickering (Fig. 3.22) and slow color drifts (Fig. 3.24). As shown in Figure 3.24, our approach generates a temporally-consistent output without any loss of spatial detail, while the method of Lang et al. either leaves residual flickering or result in undesirable spatial blurring, depending on the smoothing parameter. Comparisons with Lang et al. [102] on all sequences can be found in supplementary material.

We also compare our technique to the commercially available and unpublished RE:Vision DE:Flicker software [150]. DE:Flicker operates only on the processed video, P, with no knowledge of the input video, V, and so can be more widely applied than our technique. We evaluated this software with two types of users: naive users, i.e., us, where we tested on all of our sequences, and expert users, i.e, the authors of DE:Flicker, where three sequences were tested. Please see supplementary material for these results. Naive application often fails to remove all inconsistency, whereas our approach is broadly successful. Expert use is able to reduce more inconsistency (see Old Man autocolors), but still has problems on difficult cases (see Bedroom intrinsic decomposition).

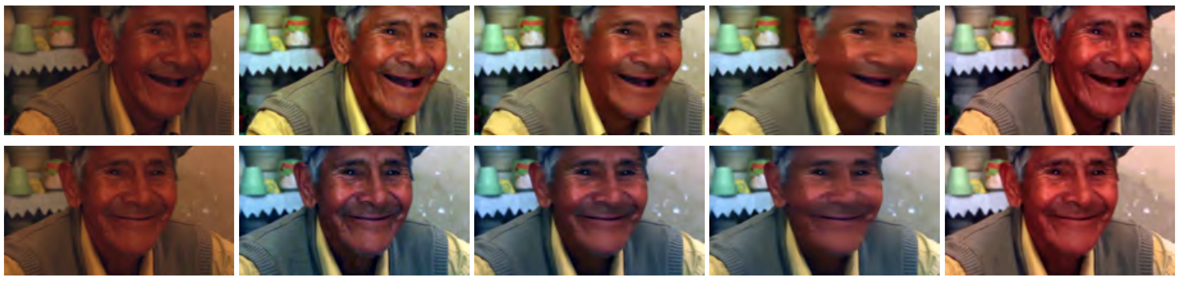


Figure 3.24: We process the original video (frames 75 and 90) (a) using the Auto-Color, Auto-Contrast, and Auto-Tone tools in Adobe Photoshop (b). The method of Lang et al. [102] does not eliminate low temporal frequency variations for short temporal kernels ($\sigma = 2$)(c) and creates spatial blurring for longer kernels ($\sigma = 10$)(d). Our method produces temporally consistent results while retaining the spatial details (e).

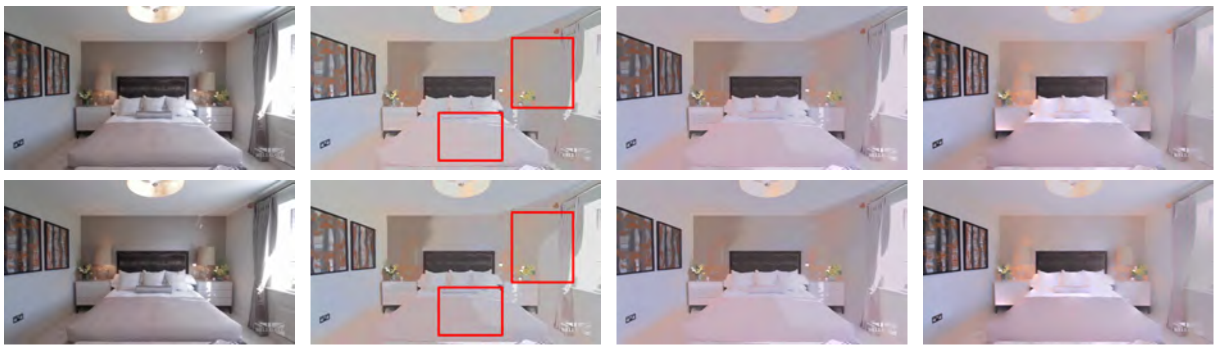
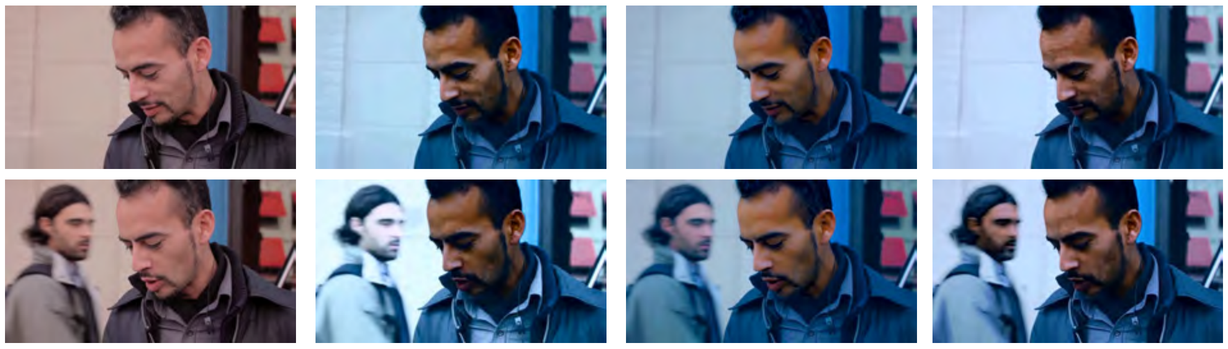


Figure 3.25: The intrinsic decomposition problem is inherently unstable. Processing the original frames (a) using the single-image technique of Bell et al. [16] produces reflectance frames with temporal inconsistencies (emphasized in red) (b). In spite of having no knowledge about the intrinsic image problem, our approach produces stabilized reflectance frames (c), that are qualitatively similar to our interactive solution (section 3.2) (d).

Applications

The strength of our technique lies in the fact that it makes very few assumptions about the underlying image processing applied to the video frames, and we have explored applying it to a wide range of applications detailed in this section.

Intrinsic Images The intrinsic decomposition problem is naturally ill-posed: multiplying one layer by a constant and dividing the other by the same constant results in the same product. The state-of-the-art intrinsic image decomposition algorithm by Bell et al. [16] produces large temporal inconsistencies when applied per frame, which we regularized with our approach (Fig. 3.25). We also successfully regularized the intrinsic image decomposition of Zhao et al. [194] applied per frame (see supplemental material). We produce results with similar quality as our temporally-consistent approach (see Sec. 3.2), which is tailored to the intrinsic decomposition problem and cannot benefit from the improvements of the Bayesian approach of Bell et al., nor any future work on this problem, without modification.



(a) Original frames (b) Single-image color grading (c) Bonneel et al. [30] (d) Our result

Figure 3.26: Applying the per-frame color model of section 3.1 produces temporal inconsistencies (see brightening in the background) (b). We filtered color-transforms in a higher-dimensional space as proposed in section 3.1 (c); we achieve a similar consistency with our algorithm that is blind to the color transfer (d).

Color Grading By-example color grading allows the transfer of color style between photographs, which is often performed by matching color statistics. We previously proposed a model consisting of a 1D luminance histogram and a 2D chrominance covariance matrix matching in each segment of foreground-background segmented frames (see section 3.1). This model produces temporal inconsistencies that are regularized in a second differential geometric step. We obtain similar temporal consistency from a more general framework (Fig. 3.26).

Color Constancy These algorithms find the white point of an image and balance it accordingly. This process can be unstable as the white point determined by the system can vary significantly from frame to frame. Color correction requires a linear transform of the red, green, and blue components of each pixel. We applied two gamut mapping methods of Gijsenij et al. [74, 75], based on edge derivatives (Fig. 3.27) and edge weighting (see additional material). Both produce relatively low temporal frequency inconsistencies that our technique eliminates.

Spatially-varying White Balance We experimented with the two-illuminant white balancing scheme of Hsu et al. [85]. This algorithm clusters a photograph into regions with the same albedo and uses them to recover the spatially-varying weights of the two illuminants. This algorithm is sensitive to the clustering step and initial light color parameters. We adjusted its parameters for the first frame of the video sequence, and then relied on our algorithm to temporally regularize the output of Hsu et al.'s algorithm (Fig. 3.28).

Color Harmonization This consists in matching and averaging color statistics of multiple images to register their color palettes. It can be used to simulate different photos being taken on the same device, or same setup, or during the same day, even when they were not. We used our sliced Wasserstein barycenter [29] to harmonize colors of three videos per frame. This resulted in minor flickering which our technique stabilizes.

Style Transfer Generalizing example-based color grading, Aubry et al. [8] introduced a method to transfer the style of a particular photograph to an input image. Used per frame, this algorithm yields minor flickering, which we are able to remove with our method (Fig. 3.29).

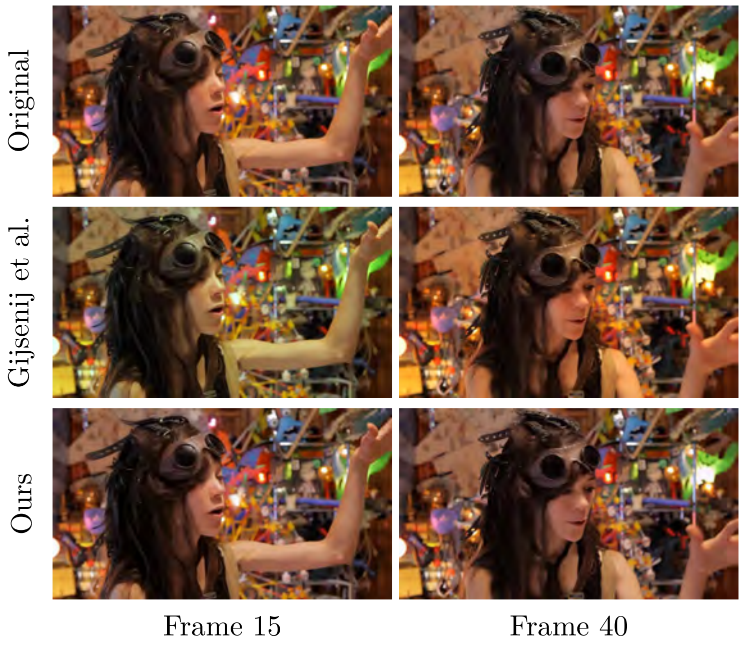


Figure 3.27: The gamut mapping of Gijsenij et al. [74] is not stable, producing different white balance on two different frames depicting Zina Lahr[2]. Our approach directly propagates the white balance from previous frames.

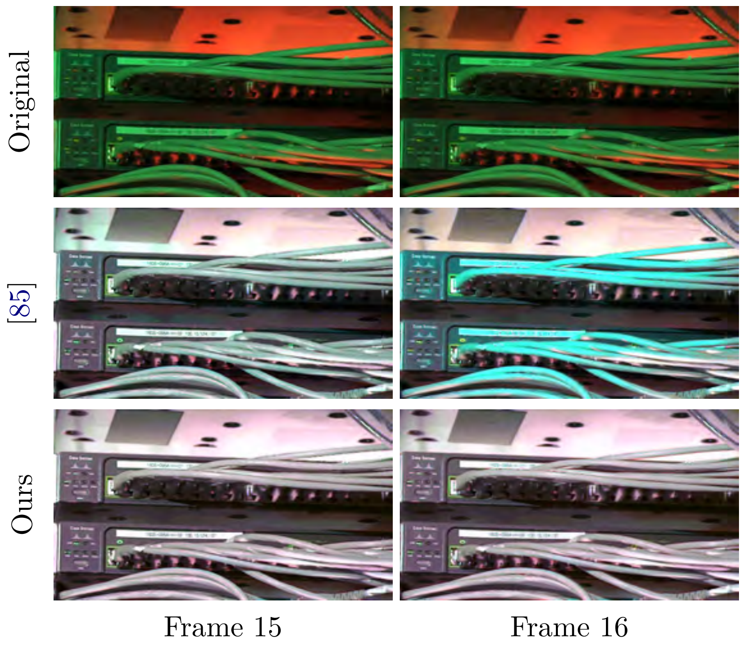


Figure 3.28: The local white balancing algorithm of Hsu et al. [85] applied to video frames is sensitive to initial parameters. Our solution regularizes the output of this algorithm.

HDR Compression We processed HDR frames with the tone mapping operators of Paris et al. [131], and Durand and Dorsey [57] on videos by Kalantari et al. [87] and Kronander et al. [98] These methods produce mostly low spatial frequency flickering. We also processed LDR frames with the "HDR Toning" filter of Adobe Photoshop (Fig. 3.29), and using Adobe Lightroom (highlights, clarity and shadows settings), and again removed the temporal inconsistencies.

[2]Zina Lahr was a fascinating artist whom I met at Siggraph 2013, and who passed away in an accident later that year. A short documentary is available at `https://vimeo.com/80973511`, part of which we used in our tests.

Dehazing We applied the algorithms of Tang et al. [179] and He et al. [81] to video frames. While we found the former more temporally stable, both benefit from our approach (Fig. 3.29).

Depth prediction Our algorithm also performs well for recovering depth from a single input image. We successfully regularize the method of Eigen et al. [58] applied per-frame (Fig. 3.29). Because of the problem difficulty, the resulting depth maps are of low resolution and only produce spatially low frequency artifacts that are easy to remove with our approach.

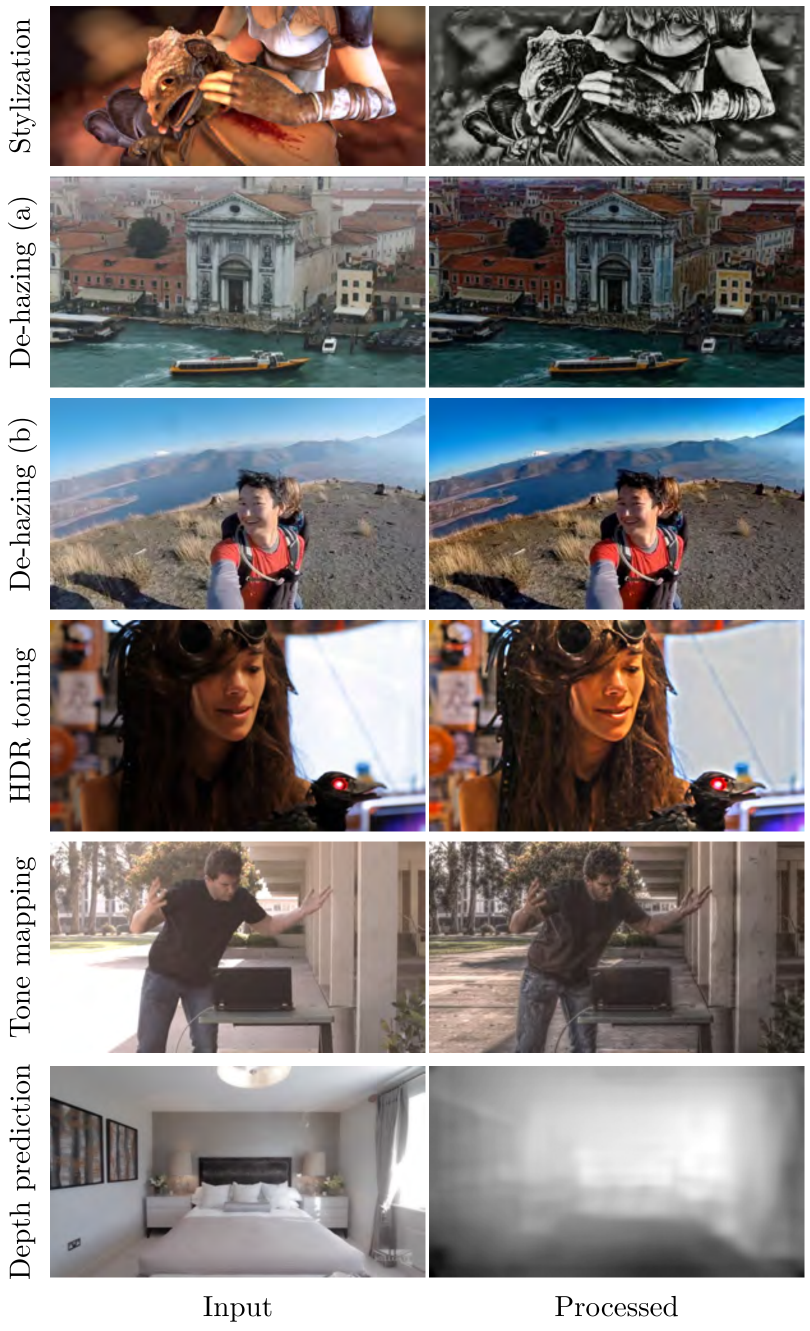


Figure 3.29: We experimented with other filters which produce a flickering too subtle for side-by-side comparison. Among those, the style transfer approach of Aubry et al. [8], two dehazing methods [81] (a) and [179] (b), Photoshop's *HDR toning* effect, the tone mapping of Paris et al. [131], and depth prediction [58]. Please appreciate temporal consistency in the video accompanying [33].

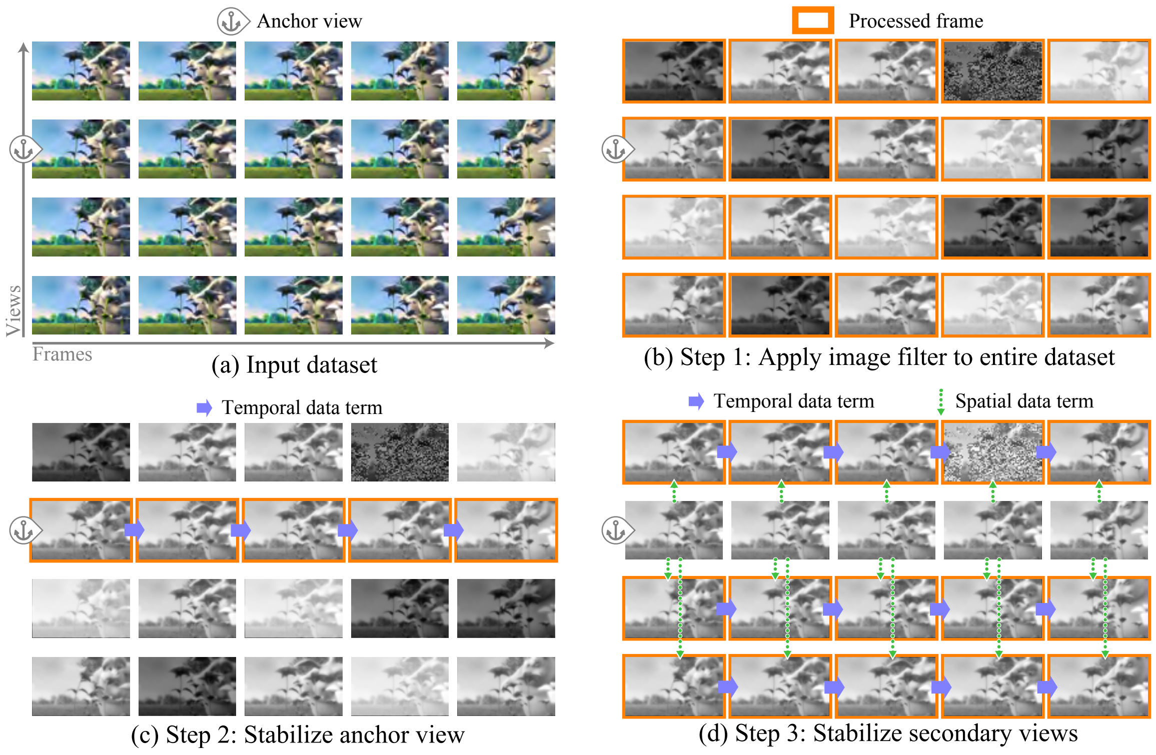

Anchor view
Processed frame
Views
Frames
(a) Input dataset
(b) Step 1: Apply image filter to entire dataset
Temporal data term
Spatial data term
(c) Step 2: Stabilize anchor view
(d) Step 3: Stabilize secondary views

Figure 3.30: **Multi-view algorithm overview.** We apply image filer f to input data $\{V_{ij}\}$ (a) to produce filtered data $\{P_{ij}\}$. We are interested in the case where these data suffer from spatial and temporal instability (b). First, we stabilize the anchor view $\{P_{at}\}$ using an existing technique to remove temporal inconsistencies from videos (c). Then, we stabilize the secondary frames $\{P_{ij}\}$ (for $i \neq a$) using the anchor frames $\{O_{at}\}$ as reference in addition to the previous frames (d). We reorder this computation to be more storage efficient: in practice, (c) and (d) are interleaved to create a streaming algorithm. Please view in color. ('Big Buck Bunny Flower': $\lambda_t = 0.1, \lambda_s = 0.1$.)

3.3.3 Restoring Multi-View Video Spatial and Temporal Consistency

We now consider a dataset with several *views* $\{V_i\}$, e.g., a stereo dataset has two views V_1 and V_2, and a 5×5 light field has 25 views. We use x to denote the position of a pixel in a view; and for dynamic sequences, e.g., a stereo video, we use t to index the frames, i.e., V_{it} is the t^{th} frame of the i^{th} view. Our algorithm takes as input an unprocessed dataset $\{V_{it}\}$ and an image filter f. By analogy with the single-view case, we name $\{P_{it}\} = \{f(V_{it})\}$ the dataset filtered frame-by-frame by f. We are interested in cases where the filtered dataset $\{P_{it}\}$ suffers from spatial and/or temporal inconsistencies. Our objective is to generate an output dataset $\{O_{it}\}$ that retains the visual appearance of $\{P_{it}\}$ while being spatially and temporally consistent.

We design our algorithm to handle large numbers of views and arbitrary video lengths. We proceed in two main steps. First, we select an *anchor view* denoted by a, and stabilize it. Second, we operate on the other views $i \neq a$, processing frames one at a time in temporal order (Fig. 3.30). One major advantage of our algorithm is that when we treat the frame P_{it}, we only need to consider its input counterpart V_{it}, its predecessor $V_{i(t-1)}$, and the corresponding output anchor frame O_{at}, resulting in little memory overhead. We formulate our problem by extending the single-view energy described in subsection 3.3.1.

Stabilizing the Anchor View

The first step of our algorithm is to select by hand an anchor view. In principle this could be any view, but as we wish to maximise potential coverage in correspondence, then this is typically the most central view in the dataset. We name a its index. $\{V_{at}\}$, $\{P_{at}\}$, and $\{O_{at}\}$ are datasets restricted to a single view. To remove any inconsistency introduced in $\{P_{at}\}$ by the image filter, we use the method described in subsection 3.3.1 with the least-square energy of Equation 3.11a. We denote $\mathcal{T}_{()}$ the warping operator warp() between consecutive frames in time. In practice, for $\mathcal{T}_{()}$, we adapt the PatchMatch algorithm [13] (§3.3.3) to compute a correspondence $\mathcal{T}_{at}$ between the input frames $V_{a(t-1)}$ and V_{at}.

Stabilizing the Secondary Views

Once we have stabilized the anchor view a, we process secondary views $i \neq a$ one by one. For each view, the process is causal and only involves data at t and $(t-1)$ similarly to Equation 3.11a. The difference is the addition of a data term that relates the considered frame P_{it} to its corresponding stabilized anchor frame O_{at}:

$$\begin{aligned}\operatorname{argmin}_{O_{it}} \int & \|\nabla O_{it} - \nabla P_{it}\|^2 \\ & + w_{\mathrm{t}}(x)\|O_{it} - \mathcal{T}_{it}(O_{i(t-1)})\|^2 \\ & + w_{\mathrm{s}}(x)\|O_{it} - \mathcal{S}_{it}(O_{at})\|^2 \mathrm{d}x \end{aligned} \tag{3.14a}$$

$$\text{with: } w_{\mathrm{t}}(x) = \lambda_{\mathrm{t}} \exp(-\alpha_{\mathrm{t}} d(V_{it}, \mathcal{T}_{it}(V_{i(t-1)}))^2) \tag{3.14b}$$

$$w_{\mathrm{s}}(x) = \lambda_{\mathrm{s}} \exp(-\alpha_{\mathrm{s}} d(V_{it}, \mathcal{S}_{it}(V_{at}))^2) \tag{3.14c}$$

where $\mathcal{S}_{it}(\cdot)$ is the spatial warp operator that puts in correspondence the pixels in view i with those in the anchor view a, $d(.,.)^2$ denotes the quality of alignment, computed as the sum of squared pixel difference over a 7×7 patch neighborhood and α_{s} and λ_{s} are parameters that controls the influence of the view consistency term. We use a modified version of PatchMatch to define the operators $\mathcal{S}_{it}$ and $\mathcal{T}_{it}$ as described in details in our paper [32], which goal is to prevent overly small patches while improving the speed of PatchMatch – optical flows being untractable for the amount of data being processed in multi-view videos. Here, λ_{t} controls the regularization strength over time and λ_{s} across views, while α_{t} and α_{s} control the confidence we give to the computed correspondences. In practice, we use $\lambda_{\mathrm{t}} \in [1, 50]$, $\lambda_{\mathrm{s}} \in [0.2, 10]$ and $\alpha_{\mathrm{t}} = \alpha_{\mathrm{s}} \in 7 \times 7 \times [0.1, 1]$.

Akin to Equation 3.11a, this process is a least-squares optimization problem that amounts to solving a Screened Poisson Equation. Most importantly, the memory requirement of our approach does not dependent on the number of views and frames in a dataset, e.g., we can process arbitrarily long sequences within a fixed amount of memory, by re-ordering computations. Instead of processing the entire anchor view first and then secondary views, we sequentially process all views at time t (see our paper [32] for details). As the process at each frame amounts to running PatchMatch at most twice (for $\mathcal{S}$ and $\mathcal{T}$) and solving a linear system of the same size and sparsity that is independent of the number of views and frames, the computational complexity of processing a dataset is linear in the number of views and frames, that is: $\mathcal{O}(N_{\mathrm{v}} N_{\mathrm{f}})$, where N_{v} is the number of views in a dataset and N_{f} is the number of frames per view.

In the appendix of our paper [32], we perform a similar Fourier analysis which shows that while the high frequencies still mostly come from the processed frame P_{it}, the low frequencies are dominated by a blend of the previous frame $O_{i(t-1)}$ and of the anchor frame O_{at}, the balance between them being controlled by w_{s} and w_{t}.

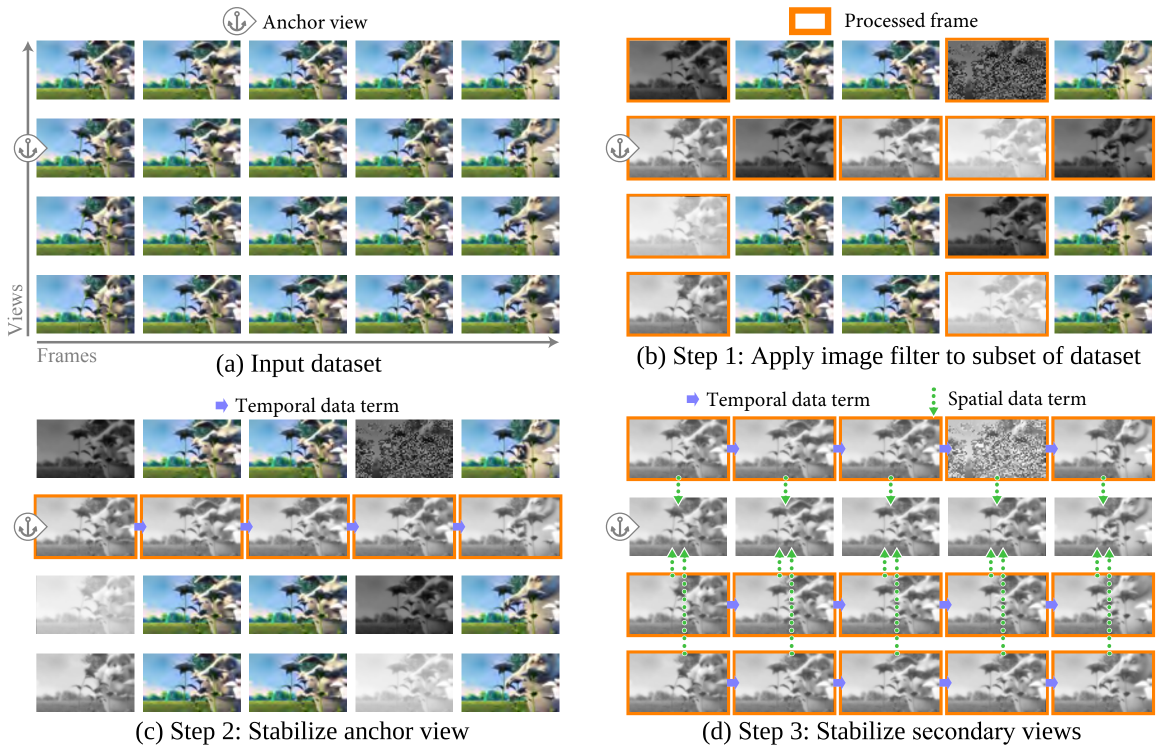

Anchor view
Views
Frames
(a) Input dataset
Processed frame
(b) Step 1: Apply image filter to subset of dataset
Temporal data term
(c) Step 2: Stabilize anchor view
Temporal data term
Spatial data term
(d) Step 3: Stabilize secondary views

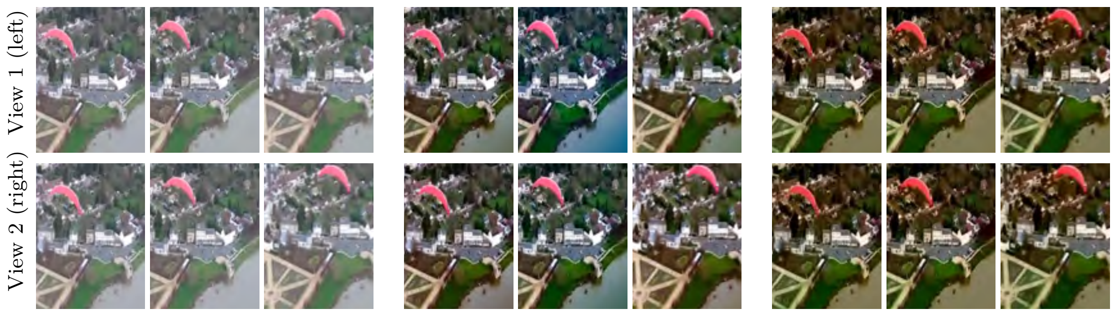
Figure 3.31: We accelerate our algorithm by applying image filter f only to a subset of the input data (a). We filter all frames at $t = 0$ and all anchor frames, and 1 in n frames for the secondary views (b). We use $n = 3$ in this example. The rest of the algorithm remains the same (c,d). This significantly speeds up the process by "hallucinating" the effect of the filter. We reorder this computation to be more storage efficient by interleaving (c) and (d) to create a streaming algorithm. Please view in color. ('Big Buck Bunny Flower': $\lambda_t = 0.1, \lambda_s = 0.1$.)

Speed-up via Filter Transfer

The approach described produces stable outputs for many datasets and filters (§3.3.4). However, applying image processing filters across multiple frames and views can be prohibitively expensive when the image filter f is costly. We propose an approximation which enables a significant speed-up to the per-frame filtering while maintaining a satisfying output. Our strategy is to filter only a subset of the frames, i.e., we compute $f(V_{it})$ only for some i and t values and let our regularization technique transfer the response from these frames (Fig. 3.31). With this strategy, we exploit the high level of redundancy between nearby frames and views to propagate the effect of the image filter f from a few filtered frames onto all the others even if they have not been filtered. In our algorithm, the anchor frames are more important because they regularize the secondary frames. Building on this observation, we filter all the anchor frames as normal, but filter only one in every n secondary frames. Because our approach is causal, we also filter the first frames of all views, i.e., we compute $f(V_{i0})$ for all i. Formally, we define the dataset $\{\tilde{P}_{it}\}$:

$$\tilde{P}_{it} = \begin{cases} f(V_{it}) & \text{if } (i = a) \text{ or } (t \bmod n \equiv 0) \\ V_{it} & \text{otherwise} \end{cases} \tag{3.15}$$

Then, the algorithm is the same, i.e., we minimize Equations 3.11a and 3.14 using $\tilde{P}$ instead of P. As we filter all anchor frames, Equation 3.11a is unchanged; only secondary view processing (Eq. 3.14) is affected. An experimental analysis of the approximation error introduced by this scheme is described in our paper [32]. We found that speed-up factors of 5× were possible while keeping a satisfying output quality.



(a) Input camera array video (stereo rig) (b) Per-frame filtered with dehaze operator (c) Our result after enforcing consistency

Figure 3.32: Our method turns inconsistent filtered video from camera arrays into *consistent* video across time and view. Applying image dehazing per frame to an input stereo video (a) produces unstable results across time and view (b). Our approach makes the resulting stereo video consistent (c). (‘Up and Down’: $\lambda_t = 0.05, \lambda_s = 0.05$.)

3.3.4 Results for Multi-View Videos

We found that our approach performs well on a broad range of image filters as shown in our experiments. First, we report run times and memory consumption, then we run experiments with a variety of filters, scenes, and camera array configurations.

Performance Our algorithm is efficient, and scales *linearly* with the number of views. In many cases, camera array sequences either have high angular and low spatial resolution, or high spatial and low angular resolution. Our algorithm scales linearly with angular resolution, and super-linearly with spatial resolution due to our multi-scale Poisson solver and PatchMatch.

On a 6-core Intel Xeon 3.5 GHz, a low resolution 320×240 15-view sequence is regularized at a rate of approximately 2.7 seconds per frame for all views, while a high resolution 1280×768 91-view sequence takes roughly 2 minutes per frame for all views, or 1.3 seconds per image. With our unoptimized implementation, this corresponds to a throughput of 0.8 mega-pixel per second.

Color mapping Automatic color adjustment filters are common, and so we tested both Adobe Photoshop and Premiere Pro on various multi-camera setups: light fields (‘Truck’, ‘Aquarium’), stereo videos (‘Bangkok Chaos’), and multi-view setups (‘Magician’). Common artifacts are flickering, which is often caused by bright object, such as the sun or a flame, entering the frame, and more subtle color casts which vary between views and over time as occlusions reveal unseen areas. In most cases, our approach is able to correct for both artifacts. However, the lack of correspondences between views in the ‘Magician’ sequence leaves some temporal inconsistencies. Similar flickering artifacts are seen with other color mapping functions such as tone mapping and contrast-preserving decoloration (color2gray), which we successfully remove.

Stylization Stylization methods often perform some kind of color mapping in combination with edge highlights, which can exhibit flickering over time and space. We show a panoramic example using the watercolor filter from Adobe Photoshop applied to each frame, and then regularized. In general, filters which add new edges which are inconsistent with input edges will lead to those new edges being smoothed out during consistency enforcement (Fig. 3.33).

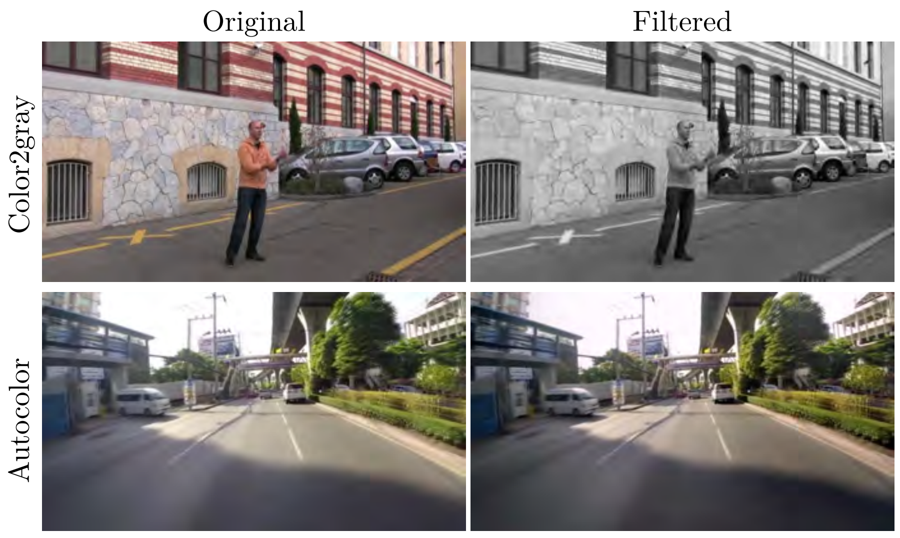


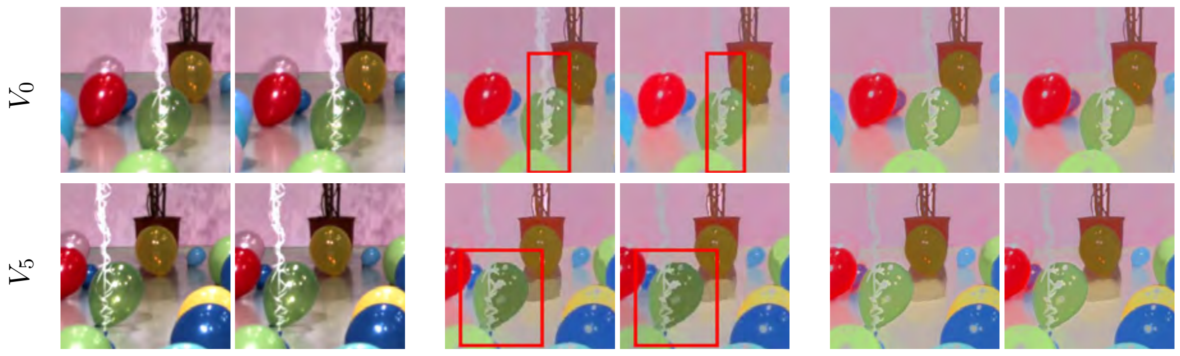
Figure 3.33: Two examples of other filters that resulted in flickering that was corrected. (‘Juggler’: $\lambda_t = 0.2, \lambda_s = 0.2$. ‘Bangkok Chaos’: ($\lambda_t = 0.01, \lambda_s = 0.01$.))

Dehazing We ran the Photoshop Dehazing filter on each frame of a stereo video, which estimates an atmospheric haze color from image statistics, and uses this to remove haze in outdoor images. In this case, small changes in this estimate led to drastic changes in the result, which we were able to make consistent (Fig. 3.32).

Intrinsic Decomposition Similarly to our single-view experiments, we computed a per-frame decomposition into reflectance and albedo information using Bell et al. [16] (Fig. 3.34). This filter shows significant flickering across time and view, but our approach is able to enforce consistency across both. In Fig. 3.35, we show a comparison with the recent (static) light field intrinsic decomposition method of Garces et al. [70]. Additional comparisons can be found in supplemental materials of our paper [32].

Non-photorealistic Rendering To help evaluate which filters are appropriate for our approach, we also consider NPR filters which create a significantly pronounced effect. As our algorithm works in the gradient domain, it implicitly assumes that the input edge structure can effectively regularize the filter. Thus, our approach is not appropriate for filters which add significant new edges to the output, e.g., Adobe Photoshop crystallize filter (Fig. 3.38). This is because when there is a new edge in the output that is missing from the input, our approach blurs it away, producing an unsatisfying result.

However, for iterative filters that introduce new edges, such as the popular neural style transfer [72] or ‘deep dream’ effects [122], we propose an iterated (filter $\rightarrow$ regularize $\rightarrow$ filter) approach to cope with this edge blurring problem. By interleaving our regularization within style transfer iterations, we are able to remove significant inconsistency at low computational cost, while retaining the desired style, as the high frequencies are added back at each iteration. We perform this iteration four times to create a stable result, as shown in the video and in Fig .3.36. One contemporaneous approach for temporally stable style transfer uses long-term features to track objects across motions in videos [154]. While effective, this approach relies on optical flow and takes roughly 3 minutes per image pair, which would be prohibitive when scaling to large numbers of views. Further, their approach modifies the underlying formulation [72], while ours is agnostic to the formulation and so applies to a class of problems.



(a) Input Videos (cropped) (b) Per-frame intrinsic albedo (c) Our result after enforcing consistency

Figure 3.34: Decomposing light field videos into albedo and reflectance information produces unstable results due to changes in content over time and view. For instance, in (b), V_0, the balloon string brightness changes over time; in V_5, the green balloon has different appearance either side of the string. Our method greatly reduces the resulting flicker, yielding a consistent result. ('Balloon': $\lambda_t = 0.2, \lambda_s = 0.02$.)

Random Filters To demonstrate the efficacy of our filter transfer approach, we applied random color transforms to all but the anchor view of the 'Treasure' static light field (Fig. 3.37). Despite the drastic frame-to-frame differences, our approach is still able to produce a mostly consistent output. While a real world per-frame filter is unlikely to produce such extreme variations, this test shows that when the gradients are kept intact, our approach can still stabilize the views in the presence of significant instability.

Other experiments and comparisons to state-of-the-art and proprietary methods are presented in our paper [32].

Discussion

Our approach addresses the problem of temporal instability introduced by applying unstable image filter to single and multi-view videos. It is designed to exploit a pair of unprocessed–processed videos and is not meant to handle the single-sequence scenario. For instance, it cannot remove flickering due to problems at capture time such as sensor noise or temporal aliasing of time-lapse videos, since, in these cases, there is no temporally-consistent input video to be used as a reference.

We found that our approach does not work well on matting because instabilities occur on fuzzy object boundaries which is also where optical flow techniques tend to fail (Fig. 3.38). Further, as discussed previously, artistic filters that create content uncorrelated with their input are also problematic, such as painterly stylization (Fig 3.38), although when filters are formulated as iterative processes, our solution interleaving filter iterations and regularization alleviate this issue. To speed-up the per-frame processing, we experimented with a fixed subsample factor, but a prior video analysis would allow automatic subsample factor adjustment based on motion or content. Our work favored a solution for which no precomputation is necessary.

In some cases, standard PatchMatch introduces posterization artifacts, but by adding rigid motion initialization, limiting the search radius and number of repeated target patches, and skipping constant source patches, we effectively increase robustness and reduce artifacts. That said, some sequences can be challenging, e.g., 'Magician', where artifacts in shadows and smooth

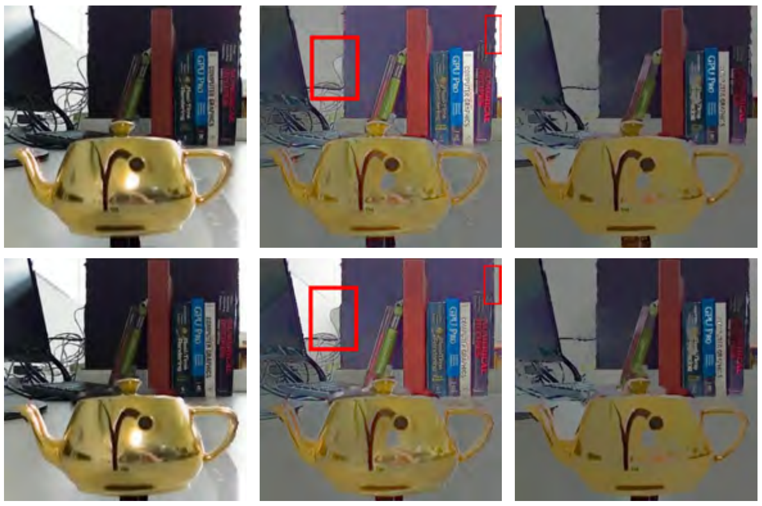

(a) Input Images (b) [70] (c) Our result

Figure 3.35: We compare our approach with that of Garces et al. [70] which operates on static light field images. We enforce consistency across views on the albedo obtained via the method of Zhao et al. [194]. The method of Garces et al. exhibits consistency artifacts across views, shown in red. ('Teapot': $\lambda_t = 0, \lambda_s = 1.$)

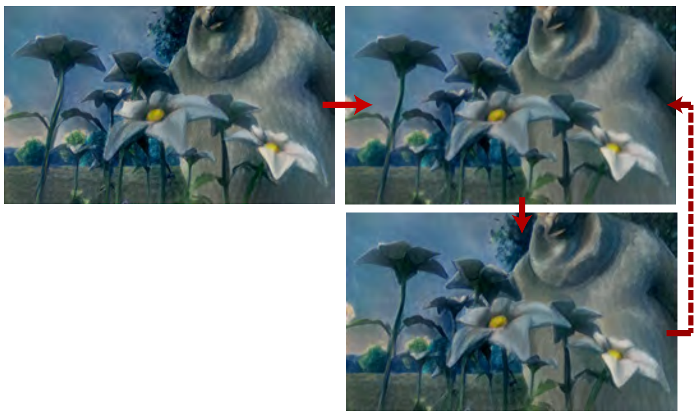
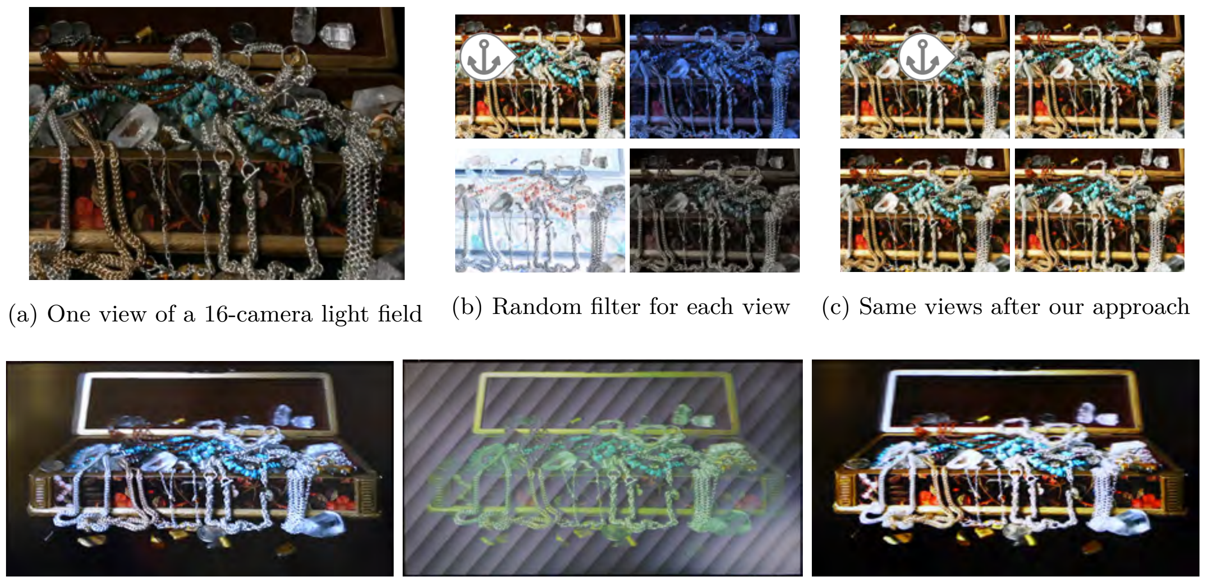


Figure 3.36: Iterative NPR filters can benefit from our regularization. We stabilize the Gatys et al. approach [72]. Our process averages out high frequencies over time but integrating it into the NPR loop reintroduces them. ('Big Buck Bunny Flower': $\lambda_t = 0.1, \lambda_s = 0.1.$)

gradients come from correspondence errors. While our method helps reduce 'one-to-many' correspondence errors over standard PatchMatch in wide-baseline or low-textured sequences, some sequences still produce artifacts due to the challenge of finding correspondence in general situations where disocclusion between views may be large. Prior information about the camera layout, or improvements in registration, will help generalize our approach.

3.3.5 Conclusion

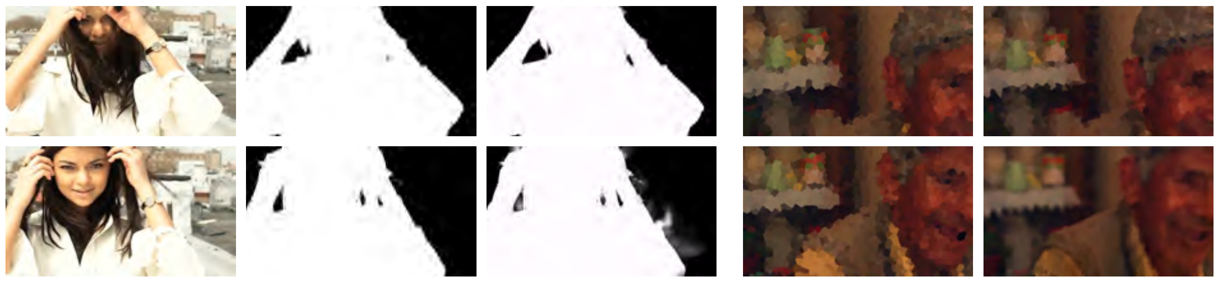
We have described a blind algorithm to stabilize the output of image processing filters applied frame by frame to videos, and static or dynamic light fields. Our approach relies on a standard least-squares energy that can be solved with a linear system that has the property of scaling linearly with the number of cameras in the array and with the duration of sequence, and of running within a fixed amount of memory independent of these parameters. We have analyzed the properties of



(a) One view of a 16-camera light field (b) Random filter for each view (c) Same views after our approach

(d) Output on an automultiscopic display (left to right: input, random filters causing view artifacts, our output).

Figure 3.37: Torture test: We chose a random filter effect for each view (b). The anchor frame (top left) stabilizes the remaining views. Even with vastly different appearances, our approach still creates a consistent output (c). When viewed on an automultiscopic display (d), view inconsistencies cause spatially-varying artifacts which change over views. Our approach removes them. (‘Treasure’: $\lambda_t = 0, \lambda_s = 10$.)



(a) Original frames (b) Matte frames (Shahrian et al. [166]) (c) Our result (d) Processed frames (e) Our result

Figure 3.38: *Left:* The matting problem (here, Shahrian et al. [166]) (b), produces temporal artifacts on object boundaries, precisely where the optical flow is unreliable due to occlusions, which leads to bleeding (c). *Right:* Some NPR effects, such as the ‘Crystallize’ tool of Adobe Photoshop, produce temporally inconsistent edges which do not gracefully blend in time (d). This leads our method to lose the NPR style (e).

our schemes in the Fourier domain and showed that despite its apparent simplicity, it performs a sophisticated differentiation between high and low frequencies that enables the stabilization of the video without degrading its content. Our experiments show that our technique performs significantly better than temporal smoothing and is able to produce high-quality results on a wide variety of applications independently of their inner workings, thereby helping bring the video processing toolbox closer to parity with that of images. For computationally expensive image filters, we have proposed an acceleration strategy that runs the filter only a subset of the data and then transfers the result to nearby frames and views. Our experiments have demonstrated that our approach enables a large class of standard filtering operations on media like stereo and light field videos that are rapidly gaining in popularity.

3.4 Conclusions on Temporal Consistency

My work on temporal consistency has been progressively built upon, in such a way that I now consider our blind temporal consistency to completely subsume our previous temporal consistency approaches tailored for color grading and intrinsic video decomposition. It works very well, it is fast, and it is easy to implement. Nevertheless, even for static images, our intrinsic video decomposition technique remains a reasonable solution mostly because the interactive process allows users to refine the decomposition until it matches their expectations. It hence remains among the best algorithms we evaluated in our state-of-the-art report [26]. Similarly, our color grading tool also remains a reasonable approach. I believe this is in part due to the combined lack of definition for a ground truth color transfer, and the very tolerant human color perception. Even drastically different color transfer results would perhaps be considered as similarly good. In contrast, temporal artifacts can be very disturbing, and being able to correct them while still benefiting from the continuing research and developments in image processing for static images should be very impactful, and should really push blind approaches. Our blind consistency paper has yet had a moderate impact in the last 3 years, but I believe it should set the ground for video filtering research and its impact should likely increase. In the meantime, a convolutional neural network approach builds upon our work [100] and emphasizes the trend for deep learning that concluded the first part of this HDR.

4 Perspectives (English)

My research has focused on two aspects – optimal transport and video processing. I wish to explore the frontiers of these subjects, and go beyond them. This section details my personal thoughts on these subjets, and exposes a few paths I want to undertake in the near future, some of which I already started investigating.

4.1 Optimal Transport

Semi-discrete barycenters

As we have seen in subsection 1.2.2, the semi-discrete optimal transport problem considers a continuous source measure to be transported towards a (weighted) sum of diracs. The *asymmetrical* nature of this problem, discrete vs. continuous, seems to make it a bad candidate to generalize to more than two probability distributions to produce Wasserstein barycenters. Nevertheless, we have started some preliminary investigations on the problem of using Power Diagrams to define barycenters between discrete measures approximating continuous measures, by first symmetrizing the problem. This finds applications in mesh interpolation, since it provides intermediate shapes in a way that is agnostic to the topology of the input meshes (e.g., it becomes possible to meaningfully interpolate between a sphere and a torus). Interesting difficulties to investigate also include the locus of discontinuities of the transport map: this is where *tearing* occur, and careful handling is necessary. This will be part of the work of a new Ph.D. student we are recruiting as of September 2018, Agathe Herrou, in joint supervision with Bruno Levy (INRIA Nancy Grand-Est) and Julie Digne (CNRS, LIRIS). Agathe started her investigations as part of her M.Sc. internship with us. Part of her work will also (or is expected to!) consist in generalizing the inverse problems we have built on continuous measures via entropy regularized mass transport to the semi-discrete framework. Her work is funded by a grant from the *Agence Nationale de la Recherche* (ANR) called ROOT (*RegressiOn with Optimal Transport, for computer graphics and vision*). This ANR also funds a second Ph.D. student, Matthieu Heitz, working inverse problems within the entropy-regularized framework who I co-supervise with Gabriel Peyré (CNRS, ENS), Marco Cuturi (ENSAE) and David Coeurjolly (CNRS, LIRIS).

Neural Networks

The popularity of optimal transport is also currently exploding, with much research directed towards machine learning, even having given rise to whole conferences dedicated to the subject [35]. In particular, neural networks have recently benefited from optimal transport, mostly as a loss between observed and predicted probability measures, whether with simple restricted Boltzmann machines [121], Autoencoders [93] or Generative Adversarial Networks [49] (both of which use our sliced Wasserstein formulation [29]) or other generative networks [73] to name a few. However, they have, to my knowledge, not used the whole geometry of optimal transport, only as their loss function. I am currently investigating how deep neural networks could benefit from the optimal transport geometry via the entropic regularized optimal transport framework, but this could be envisaged within the sliced Wasserstein framework [29], as this facilitates the handling of high-dimensional problems[1].

Unbalanced Sliced Transport

The unbalanced optimal transport in 1D for measures consisting of non weighted Diracs is much more complex than the balanced case. This problem shares large similarities with alignments problems encountered for instance in genomics, and results in algorithms of quadratic complexity (compared to linear for the balanced case, excluding the sorting of atoms that is needed in both cases). I am currently developing a fast algorithm, already currently supporting hundreds of thousands of Diracs in reasonable time.

4.2 Imaging problems

Video-Based Rendering

In the line of multiple-view imaging problems, image-based rendering has been successful in rendering realistic and geometrically complex scenes by merely performing clever interpolations and warpings between sparse sets of photographs taken from different viewpoints, possibly using a limited rough 3d reconstruction of the scene which serves as a proxy [172]. Similar techniques have seen popular applications such as *hyperlapse* videos [95], where highly unstable video sequences (e.g., a long camera motion path captured by someone walking) could be accelerated and stabilized, or PhotoTourism [173] that proposes visits of popular places largely pictured by the crowd (see Figure 4.1). Within this context, I believe this is a good time to introduce a possible next step – that of *Video Based Rendering*. This generalization of image-based rendering is expected to result in its own set of temporal consistency issues to be adressed (in particular, for dynamic scenes), and is the topic of a Ph.D student, Beatrix-Emőke Fülöp-Balogh, I am co-supervising with Julie Digne.

[1]During my defense, the jury of my HDR brought up a concern that sliced Wasserstein distances with randomized slices could suffer from a drop in convergence rate with the dimensionality as the convergence of Monte-Carlo integration depends on the variance of the integrand which itself here linearly depends on the dimension.

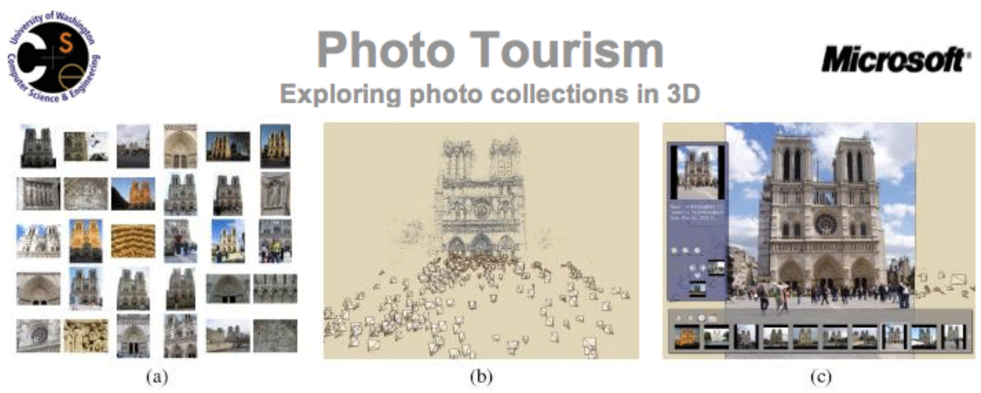


Figure 4.1: PhotoTourism allows to navigate among largely crowd-sourced photographs of popular places by clever view interpolation, a process called image-based rendering. See live demo at http://phototour.cs.washington.edu/

Computer Generated Holography

Upon investigating stabilization problems for multiple views, I worked with light field videos and other data for passive 3d displays. My curiosity towards passive displays was further aroused by an excellent Eurographics course on computational holography [112]. I am currently playing with analog holography, and envision working on computer generated holography in the relatively near future. This amounts to generalizing the multiple views light field images use to instead encode the light wave as interference patterns (combinations of light fields and holography have recently been studied [170]). Realistic and efficient rendering of images for holography remains an open problem.

5 Perspectives (Français)

Ma recherche s'est focalisée sur deux aspects – le transport optimal et le traitement vidéo. Je souhaite explorer les frontières de ces sujets, et aller au-delà. Cette section détaille mes réflexions personnelles sur ces sujets, et expose plusieurs pistes que je souhaite entreprendre dans un futur proche, dont certaines que j'ai dejq commencé à investiguer.

5.1 Transport Optimal

Barycentres Semi-discrets

Comme nous avons vu en subsection 1.2.2, le transport optimal semi-discret considère une mesure source continue transportée vers une somme (pondérée) de diracs. La nature *asymmétrique* de ce problème, discret vs. continu, semble le rendre un mauvais candidat pour une généralisation au cas de plus de deux distributions de probabilité pour produire des barycentres de Wasserstein. Néanmoins, nous avons commencé des investigations préliminaires de ce problème en utilisant des diagrammes de puissance pour définir un barycentre entre deux mesures discrètes approximant des mesures continues, en symétrisant d'abord le problème. Cela trouve des applications en interpolation de maillages, puisque cela produit des formes intermédiaires d'une manière agnostique à la topologie des maillages en entrée (par exemple, il devient possible d'interpoler d'une manière qui ait un sens entre une sphère et un tore). Une difficulté intéressante à investiguer est le lieu des discontinuités de la fonction de transport: c'est là où se produisent des *déchirements*, ce qui peut nécessiter un traitement spécial.

Cela fera partie du travail de la nouvelle doctorante que nous recrutons en Septembre 2018, Agathe Herrou, en co-supervision avec Bruno Lévy (INRIA Nancy Grand-Est) et Julie Digne (CNRS, LIRIS). Agathe a commencé ses investigations lors de son stage de Master avec nous. Une partie de son travail (du moins attendu !) consistera à généraliser les problèmes inverses que nous avions définis pour des mesures continues via le transport optimal régularisé entropiquement vers le cadre semi-discret. Son travail est financé par une subvention de l'*Agence Nationale de la Recherche* (ANR), appelé ROOT (*RegressiOn with Optimal Transport, for computer graphics and vision*). Cette ANR finance par ailleurs un second étudiant en thèse, Matthieu Heitz, qui travaille sur les problèmes inverses dans le cadre du transport régularisé par l'entropie, que je co-encadre avec Gabriel Peyré (CNRS, ENS), Marco Cuturi (ENSAE) and David Coeurjolly (CNRS, LIRIS).

Réseaux de Neurones

La popularité du transport optimal est en train d'exploser, avec beaucoup de recherche dirigée vers l'apprentissage par ordinateur, ayant même donné lieu à des conférences entières dédiées au sujet [35]. En particulier, les réseaux de neurones ont bénéficié du transport optimal, surtout en tant que fonction de perte entre les mesures de probabilités observées et prédites, soit avec de simples machines de Boltzmann restreintes [121], des autoencoders[93] ou des réseaux antagonistes génératifs [49] (les deux utilisant notre formulation du transport optimal par tranches [29]) ou d'autres réseaux génératifs [73] pour en citer quelques-uns. En revanche, à ma connaissance ils n'ont pas utilisé la géométrie du transport optimal, mais uniquement en tant que fonction perte. J'investigue actuellement comment les réseaux de neurones profonds pourraient bénéficier de la géométrie du transport optimal via le cadre du transport optimal régularisé par l'entropie, mais cela pourrait aussi être envisagé dans le cadre du transport optimal par tranches [29] puisque cela facilite la gestion de problèmes en haute dimension[1].

Transport par tranche non équilibré

Le transport optimal non équilibré en 1D pour des mesures consistant en des diracs non pondérés est beaucoup plus complexe que dans le cas équilibré. Ce problème partage des similarités avec les problèmes d'alignements rencontrés, par exemple, en génomique, et résulte en des algorithmes de complexité quadratique (comparé à linéaire pour le cas équilibré, en excluant le tri des atomes nécessaire dans les deux cas). Je développe actuellement un algorithme rapide, supportant déjà des centaines de milliers de Diracs en temps raisonnable.

5.2 Problèmes en Imagerie

Rendu Basé Vidéo

Dans la lignée des problèmes d'imagerie multi-vues, le rendu basé images a permis de rendre des scènes réalistes et géométriquement complexes, simplement en interpolant intelligemment et déformant une base de données éparse de photos prises depuis différents points de vue, en utilisant possiblement une reconstruction 3D partielle de la scène qui sert de proxy [172]. Des techniques similaires ont permis des applications populaires telles que les vidéos *hyperlapse* [95], où une séquence vidéo très instable (par exemple, une longue trajectoire de camera capturée par quelqu'un qui marche) est accélérée et stabilisée, ou le PhotoTourisme [173] qui propose des visites d'endroits populaires largement pris en photos par la foule (voir Figure 5.1). Dans ce contexte, je pense qu'il est opportun d'introduire la prochaine étape – celle du rendu basé vidéo. Je m'attends à ce que cette généralisation du rendu basé image résulte en son propre ensemble de problèmes de cohérence temporelle à résoudre (en particulier, pour les scènes dynamiques), et est le sujet de notre étudiante en thèse Beatrix-Emőke Fülöp-Balogh, que je co-encadre avec Julie Digne.

[1]Lors de ma soutenance, le jury de mon HDR s'est inquiété du fait que le transport optimal par tranches avec des tranches aléatoires pourrait souffrir d'une baisse du taux de convergence avec la dimensionalité, car la convergence de l'integration par Monte Carlo dépend de la variance de l'intégrande, qui elle-même dépend ici linéairement de la dimension.

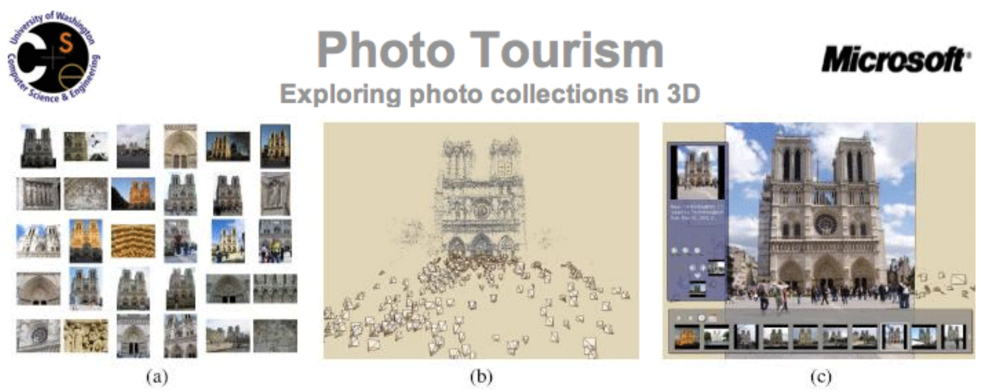


Figure 5.1: Le PhotoTourisme permet de naviguer dans une série de photographies capturées par la foule en interpolant intelligemment les vues, un processus appelé rendu basé images. Voir une démo à http://phototour.cs.washington.edu/

Holographie Générée par Ordinateur

En investiguant les problèmes de stabilisation multi-vues, j'ai pu travailler avec des vidéos champs de lumière (*light field*) et d'autres données pour les écrans 3D passifs. Ma curiosité vers les écrans passifs a été attisée par un excellent cours à Eurographics sur l'holographie computationnelle [112]. Je m'adonne actuellement à l'holographie analogique, et envisage de travailler sur l'holographie générée par ordinateur dans un futur relativement proche. Cela correspond à généraliser les images light field multi-vues pour encoder l'onde lumineuse comme figure d'interférence (des combinaisons de light fields et d'holographie ont récemment été étudiés [170]). Le rendu d'images de synthèse réalistes et de manière efficace pour l'holographie reste ouvert.

6

Bibliography

[1] M. Agueh and G. Carlier. Barycenters in the wasserstein space. *SIAM Journal on Mathematical Analysis*, 43(2):904–924, 2011. 17, 20

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993. 32

[3] N. Aifanti, C. Papachristou, and A. Delopoulos. The MUG facial expression database. In *Image Analysis for Multimedia Interactive Services (WIAMIS), 2010 11th International Workshop on*, pages 1–4. IEEE, 2010. 56, 58

[4] S.-i. Amari and H. Nagaoka. *Methods of information geometry*, volume 191. American Mathematical Soc., 2007. 18, 19, 68

[5] L. Ambrosio. Optimal transport maps in Monge-Kantorovich problem. *ArXiv Mathematics e-prints*, Apr. 2003. 14

[6] J. André, D. Attali, Q. Mérigot, and B. Thibert. Far-field reflector problem under design constraints. *Int. J. Comput. Geom. Appl.*, 25(2):143, 2015. 21

[7] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. arXiv:1701.07875, 2017. 22

[8] M. Aubry, S. Paris, S. Hasinoff, J. Kautz, and F. Durand. Fast local Laplacian filters: Theory and applications. *ACM Trans. on Graphics (SIGGRAPH)*, 2014. 88, 90

[9] F. Aurenhammer, F. Hoffmann, and B. Aronov. Minkowski-type theorems and least-squares clustering. *Algorithmica*, 20(1):61–76, 1998. 24, 25

[10] A. Averbuch, R. Coifman, D. Donoho, M. Israeli, Y. Shkolnisky, and I. Sedelnikov. A framework for discrete integral transformations: II. The 2D discrete Radon transform. *SIAM J. Sci. Comput.*, 30(2):785–803, 2008. 38

[11] T. O. Aydin, N. Stefanoski, S. Croci, M. Gross, and A. Smolic. Temporally coherent local tone mapping of hdr video. *ACM Trans. Graph.*, 33(6):196:1–196:13, Nov. 2014. 27, 82

[12] X. Bai, J. Wang, D. Simons, and G. Sapiro. Video SnapCut: robust video object cutout using localized classifiers. In *ACM Trans. on Graphics (Proc. of ACM SIGGRAPH 2009)*, page 70:1–70:11, 2009. 65, 68, 79

[13] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. on Graphics (SIGGRAPH)*, 28(3), 2009. 86, 92

[14] F. Bassetti, A. Bodini, and E. Regazzini. On minimum kantorovich distance estimators. *Statistics & probability letters*, 76(12):1298–1302, 2006. 22

[15] F. Bassetti and E. Regazzini. Asymptotic properties and robustness of minimum dissimilarity estimators of location-scale parameters. *Theory of Probability & Its Applications*, 50(2):171–186, 2006. 22

[16] S. Bell, K. Bala, and N. Snavely. Intrinsic images in the wild. *ACM Trans. on Graphics (SIGGRAPH)*, 33(4), 2014. 87, 95

[17] J. D. Benamou and Y. Brenier. A computational fluid mechanics solution to the Monge–Kantorovich mass transfer problem. *Numer. Math.*, 84(3):375–393, 2000. 22, 29, 61

[18] J.-D. Benamou, G. Carlier, M. Cuturi, M. Nenna, and G. Peyré. Iterative Bregman projections for regularized transportation problems. *SIAM J. on Sci. Computing*, 2(37), 2015. 23, 44, 45

[19] E. Bernton, P. E. Jacob, M. Gerber, and C. P. Robert. Inference in generative models using the wasserstein distance. *arXiv preprint arXiv:1701.05146*, 2017. 22

[20] D. P. Bertsekas. The auction algorithm: A distributed relaxation method for the assignment problem. *Annals of operations research*, 14(1):105–123, 1988. 22

[21] F. Besse, C. Rother, A. Fitzgibbon, and J. Kautz. PMBP: PatchMatch belief propagation for correspondence field estimation. In *BMVC - Best Industrial Impact Prize award*, 2012. 86

[22] J. Bieling, P. Peschlow, and P. Martini. An efficient GPU implementation of the revised simplex method. pages 1–8, april 2010. 36

[23] J. Bigot, R. Gouet, T. Klein, and A. López. Geodesic pca in the wasserstein space. *arXiv preprint arXiv:1307.7721*, 2013. 22

[24] A. Björck. *Numerical Methods for Least Squares Problems*. SIAM, 1996. 72

[25] N. Bonneel. Le transport optimal pour des applications en informatique graphique. *Interstices*. https://interstices.info/jcms/p_92705/le-transport-optimal-pour-des-applications-en-informatique-graphique. 18

[26] N. Bonneel, B. Kovacs, S. Paris, and K. Bala. Intrinsic Decompositions for Image Editing. *Computer Graphics Forum (Eurographics State of the Art Reports 2017)*, 36(2), 2017. 71, 76, 77, 99

[27] N. Bonneel, S. Paris, M. van de Panne, F. Durand, and G. Drettakis. Single photo estimation of hair appearance. *Computer Graphics Forum*, 28(4), 2009. 9, 11, 29

[28] N. Bonneel, G. Peyré, and M. Cuturi. Wasserstein Barycentric Coordinates: Histogram Regression Using Optimal Transport. *ACM Transactions on Graphics (SIGGRAPH 2016)*, 35(4), 2016. 46, 47, 48, 52

[29] N. Bonneel, J. Rabin, G. Peyré, and H. Pfister. Sliced and Radon Wasserstein barycenters of measures. *J. of Mathematical Imaging and Vision*, 51(1), 2015. 26, 37, 38, 39, 88, 102, 106

[30] N. Bonneel, K. Sunkavalli, S. Paris, and H. Pfister. Example-Based Video Color Grading. *ACM Trans. Graph. (SIGGRAPH)*, 32(4), 2013. 64, 68, 82, 88

[31] N. Bonneel, K. Sunkavalli, J. Tompkin, D. Sun, S. Paris, and H. Pfister. Interactive Intrinsic Video Editing. *ACM Transactions on Graphics (SIGGRAPH Asia 2014)*, 33(6), 2014. 71, 75, 77, 82, 87

[32] N. Bonneel, J. Tompkin, D. Sun, O. Wang, K. Sunkvalli, S. Paris, and H. Pfister. Consistent Video Filtering for Camera Arrays. *Computer Graphics Forum (Eurographics 2017)*, 36(2), 2017. 82, 92, 93, 95, 96

[33] N. Bonneel, J. Tompkin, K. Sunkavalli, D. Sun, S. Paris, and H. Pfister. Blind Video Temporal Consistency. *ACM Transactions on Graphics (SIGGRAPH Asia 2015)*, 34(6), 2015. 82, 85, 90

[34] N. Bonneel, M. van de Panne, S. Paris, and W. Heidrich. Displacement interpolation using lagrangian mass transport. *ACM Trans. Graph. (SIGGRAPH Asia)*, 30(6), 2011. 29, 33, 36

[35] O. Bousquet, M. Cuturi, G. Peyré, F. Sha, and J. Solomon. Nips conference on optimal transport and machine learning, 2017. 102, 106

[36] Y. Brenier. Décomposition polaire et réarrangement monotone des champs de vecteurs. *C. R. Acad. Sci. Paris Sér. I Math.*, 305(19):805–808, 1987. 14

[37] J. Cantarella and M. Piatek. Tsnnls: A solver for large sparse least squares problems with non-negative variables. *CoRR*, cs.MS/0408029, 2004. 31

[38] Y. Chang, S. Saito, and M. Nakajima. Example-based color transformation of image and video using basic color categories. *Image Processing, IEEE Trans. on*, 16(2):329–336, 2007. 27

[39] X. Chen, A. Golovinskiy, and T. Funkhouser. A benchmark for 3D mesh segmentation. *ACM Trans. Graph. (SIGGRAPH)*, 28(3), 2009. 56

[40] L. Chizat, G. Peyré, B. Schmitzer, and F.-X. Vialard. Scaling Algorithms for Unbalanced Transport Problems. *Mathematics of Computation*, July 2016. 24, 51

[41] I. CPLEX. High-performance software for mathematical programming and optimization, 2005. 32, 33

[42] W. H. Cunningham. A network simplex method. *Mathematical Programming*, 11(1):105–116, 1976. 22

[43] M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2292–2300. Curran Associates, Inc., 2013. 23

[44] M. Cuturi and A. Doucet. Fast computation of Wasserstein barycenters. In *Int. Conf. on Machine Learning (ICML)*, 2014. 39, 46

[45] M. da Silva, F. Durand, and J. Popović. Linear Bellman combination for control of character animation. *ACM Trans. Graph.*, 28(3):1–10, 2009. 35

[46] F. de Goes, K. Breeden, V. Ostromoukhov, and M. Desbrun. Blue noise through optimal transport. *ACM Trans. Graph. (SIGGRAPH Asia)*, 31, 2012. 21

[47] F. de Goes, D. Cohen-Steiner, P. Alliez, and M. Desbrun. An Optimal Transport Approach to Robust Reconstruction and Simplification of 2D Shapes. *Computer Graphics Forum*, 2011. 21

[48] J. Delon and A. Desolneux. Stabilization of flicker-like effects in image sequences through local contrast correction. *SIAM Journal on Imaging Sciences*, 3(4):703–734, 2010. 27

[49] I. Deshpande, Z. Zhang, and A. Schwing. Generative modeling using the sliced wasserstein distance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3483–3491, 2018. 102, 106

[50] A. Desolneux, L. Moisan, and S. Ronsin. A compact representation of random phase and Gaussian textures. In *Proc. the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1381–1384, 2012. 40

[51] J. Digne, D. Cohen-Steiner, P. Alliez, F. de Goes, and M. Desbrun. Feature-preserving surface reconstruction and simplification from defect-laden point sets. *Journal of Mathematical Imaging and Vision*, 48(2):369–382, 2014. 21

[52] M. P. do Carmo. *Riemannian Geometry.* Springer, Jan. 1992. 67

[53] A. Dominitz and A. Tannenbaum. Texture mapping via optimal mass transport. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):419–433, May 2010. 21, 22

[54] W. Dong. LSHKIT: A C++ locality sensitive hashing library, 2014. 75

[55] X. Dong, B. Bonev, Y. Zhu, and A. L. Yuille. Region-based temporally consistent video post-processing. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 27

[56] D. Dowson and B. Landau. The fréchet distance between multivariate normal distributions. *Journal of Multivariate Analysis*, 12(3):450 – 455, 1982. 20

[57] F. Durand and J. Dorsey. Fast bilateral filtering for the display of high-dynamic-range images. In *Proc. of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, pages 257–266. ACM, 2002. 89

[58] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NIPS'14*, pages 2366–2374, 2014. 90

[59] J. H. Elder. Are edges incomplete? *Int. J. Comput. Vision*, 34(2-3):97–122, Oct. 1999. 84

[60] Z. Farbman and D. Lischinski. Tonal stabilization of video. *ACM Trans. on Graphics (Proc. of ACM SIGGRAPH 2011)*, 30(4):89:1 – 89:9, 2011. 27, 82

[61] S. Ferradans, G.-S. Xia, G. Peyré, and J.-F. Aujol. Optimal transport mixing of gaussian texture models. In *Proc. SSVM'13*, 2013. 40, 41

[62] S. Ferradans, G.-S. Xia, G. Peyré, and J.-F. Aujol. Static and dynamic texture mixing using optimal transport. In *Proc. SSVM'13*, volume 7893 of *Lecture Notes in Computer Science*, pages 137–148. Springer Berlin Heidelberg, 2013. 20

[63] J. Filip and R. Vávra. Template-based sampling of anisotropic BRDFs. *Computer Graphics Forum (PG)*, 33(7), 2014. 54

[64] M. M. Flood. On the Hitchcock distribution problem. *Pacific J. Math.*, 3(2):369–386, 1953. 16, 31

[65] J. Foote and D. Kimber. Flycam: Practical panoramic video and automatic camera control. In *IEEE Int. Conf. on Multimedia and Expo (ICME)*, volume 3, pages 1419–1422, 2000. 83

[66] C. Frogner, C. Zhang, H. Mobahi, M. Araya, and T. A. Poggio. Learning with a wasserstein loss. In *Advances in Neural Information Processing Systems*, pages 2053–2061, 2015. 22

[67] B. Galerne, Y. Gousseau, and J.-M. Morel. Random phase textures: Theory and synthesis. *IEEE Trans. on Image Processing*, 20(1):257–267, 2011. 40

[68] T. Gallouët and Q. Mérigot. A lagrangian scheme for the incompressible euler equation using optimal transport. *Foundation of Computational Mathematics (FOCM)*, 2017. 21

[69] W. Gangbo. The mass transfer problem and its applications. *Contemp. Math.*, 2256:79–104, 1999. 15

[70] E. Garces, J. I. Echevarria, W. Zhang, H. Wu, K. Zhou, and D. Gutierrez. Intrinsic light fields. *CoRR*, abs/1608.04342, 2016. 27, 95, 97

[71] E. S. L. Gastal and M. M. Oliveira. Domain transform for edge-aware image and video processing. In *ACM Trans. on Graphics (Proc. of ACM SIGGRAPH 2011)*, page 69:1–69:12, 2011. 66

[72] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *CoRR*, abs/1508.06576, 2015. 95, 97

[73] A. Genevay, G. Peyré, and M. Cuturi. Learning generative models with sinkhorn divergences. *arXiv preprint arXiv:1706.00292*, 2017. 22, 102, 106

[74] A. Gijsenij, T. Gevers, and J. van de Weijer. Generalized gamut mapping using image derivative structures for color constancy. *Int. J. Comput. Vision*, 86(2-3):127–139, 2010. 88, 89

[75] A. Gijsenij, T. Gevers, and J. van de Weijer. Improving color constancy by photometric edge weighting. *IEEE Trans on Pattern Analysis and Machine Intelligence*, 34(5):918–929, 2012. 88

[76] R. Grosse, M. K. Johnson, E. H. Adelson, and W. T. Freeman. Ground truth dataset and baseline evaluations for intrinsic image algorithms. In *ICCV*, pages 2335–2342, 2009. 73, 74

[77] T. Günther, K. Rohmer, C. Rössl, T. Grosch, and H. Theisel. Stylized caustics: Progressive rendering of animated caustics. *Computer Graphics Forum (Proc. Eurographics)*, 35(2):243–252, 2016. 21

[78] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski. Non-rigid dense correspondence with applications for image enhancement. *ACM Trans. on Graphics (SIGGRAPH)*, 30(4):70:1–70:9, 2011. 86

[79] S. Haker, L. Zhu, A. Tannenbaum, and S. Angenent. Optimal mass transport for registration and warping. *International Journal on Computer Vision*, 60(3):225–240, 2004. 21, 22, 29

[80] V. Hartmann and D. Schuhmacher. Semi-discrete optimal transport-the case p= 1. *arXiv preprint arXiv:1706.07650*, 2017. 61

[81] K. He, J. Sun, and X. Tang. Single image haze removal using dark channel prior. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1956–1963, 2009. 90

[82] S. Helgason. *The Radon Transform.* Birkhauser, Boston, 1980. 37

[83] F. S. Hillier and G. J. Lieberman. *Introduction to Operations Research, 5th ed.* McGraw-Hill, 1990. 22, 29, 32, 36

[84] F. S. Hillier, G. J. Lieberman, F. Hillier, and G. Lieberman. *Introduction to Operations Research.* McGraw-Hill Science/Engineering/Math, July 2004. 16

[85] E. Hsu, T. Mertens, S. Paris, S. Avidan, and F. Durand. Light mixture estimation for spatially varying white balance. *ACM Trans. Graph. (SIGGRAPH)*, 2008. 72, 88, 89

[86] T. Ilmanen. Lectures on mean curvature flow and related equations. In *Lecture Notes, ICTP, Trieste*, 1995. 67

[87] N. K. Kalantari, E. Shechtman, C. Barnes, S. Darabi, D. B. Goldman, and P. Sen. Patch-based High Dynamic Range Video. *ACM Trans. Graph. (SIGGRAPH Asia)*, 32(6), 2013. 89

[88] L. V. Kantorovich. On the translocation of masses. *C. R. (Doklady) Acad. Sci. USSR*, 321:199–201, 1942. 15

[89] L. V. Kantorovich. On a problem of monge. *Uspekhi Mat. Nauk*, 3(2):225–226, 1948. 15

[90] C. Kiser, E. Reinhard, M. Tocci, and N. Tocci. Real time automated tone mapping system for HDR video. In *Proc. of the IEEE Int. Conference on Image Processing.* IEEE, September 2012. 27

[91] J. Kitagawa, Q. Mérigot, and B. Thibert. A newton algorithm for semi-discrete optimal transport. *arXiv preprint arXiv:1603.05579*, 2016. 25

[92] B. Kloeckner. A geometric study of the wasserstein space of the line. 2009. 20

[93] S. Kolouri, C. E. Martin, and G. K. Rohde. Sliced-wasserstein autoencoder: An embarrassingly simple generative model. *arXiv preprint arXiv:1804.01947*, 2018. 102, 106

[94] N. Kong, P. V. Gehler, and M. J. Black. Intrinsic video. In *Eur. Conf. Comp. Vision (ECCV)*, volume 8690, pages 360–375, 2014. 27, 82

[95] J. Kopf, M. F. Cohen, and R. Szeliski. First-person hyper-lapse videos. *ACM Transactions on Graphics (TOG)*, 33(4):78, 2014. 102, 106

[96] S. Korman and S. Avidan. Coherency sensitive hashing. In *ECCV*, 2011. 75

[97] D. Krishnan and R. Fergus. Fast image deconvolution using hyper-laplacian priors. In *NIPS*, 2009. 72

[98] J. Kronander, S. Gustavson, G. Bonnet, and J. Unger. Unified HDR reconstruction from raw CFA data. *IEEE Int. Conference on Computational Photography (ICCP)*, 2013. 89

[99] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics (NRL)*, 2(1-2):83–97, 1955. 22

[100] W.-S. Lai, J.-B. Huang, O. Wang, E. Shechtman, E. Yumer, and M.-H. Yang. Learning blind video temporal consistency. *arXiv preprint arXiv:1808.00449*, 2018. 99

[101] E. H. Land, John, and J. Mccann. Lightness and retinex theory. *Journal of the Optical Society of America*, 61:1–11, 1971. 72

[102] M. Lang, O. Wang, T. Aydin, A. Smolic, and M. Gross. Practical temporal consistency for image-based graphics applications. *ACM Trans. on Graphics (Proc. of ACM SIGGRAPH 2012)*, 31(4):34:1–34:8, July 2012. 27, 65, 82, 86, 87

[103] C. L. Lawson and R. J. Hanson. *Solving least squares problems.* 3 edition, 1995. 31

[104] LEMON. LEMON: Library for efficient modeling and optimization in networks, 2010. http://lemon.cs.elte.hu/trac/lemon. 32

[105] A. Levin, D. Lischinski, and Y. Weiss. A closed-form solution to natural image matting. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 30(2):228 –242, Feb. 2008. 65

[106] E. Levinkov, J. Tompkin, N. Bonneel, S. Kirchhoff, B. Andres, and H. Pfister. Interactive Multicut Video Segmentation. In *Pacific Graphics (short paper)*, October 2016. 69

[107] B. Lévy. A numerical algorithm for l2 semi-discrete optimal transport in 3d. *ESAIM M2AN (Mathematical Modeling and Numerical Analysis)*, 2015. 21, 25

[108] B. Lévy and E. L. Schwindt. Notions of optimal transport theory and how to implement them on a computer. *Computers & Graphics*, 72:135–148, 2018. 61

[109] A. S. Lewis and M. L. Overton. Nonsmooth optimization via quasi-newton methods. *Math. Programming*, 141(1-2), 2013. 46

[110] Y. Lipman and I. Daubechies. Conformal wasserstein distances: Comparing surfaces in polynomial time. *Advances in Mathematics*, 227(3):1047 – 1077, 2011. 21

[111] C. Liu. *Beyond pixels: exploring new representations and applications for motion analysis.* PhD thesis, MIT, 2009. 76, 86

[112] P. Lobaz. Computer generated display holography. 2017. 103, 107

[113] J. Lu, P. V. Sander, and A. Finkelstein. Interactive painterly stylization of images, videos and 3d animations. In *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*, pages 127–134. ACM, 2010. 27

[114] D. MacDonald. A C++ implementation of the transportation simplex algorithm, 2006. http://www.site.uottawa.ca/ dmacd070/emd/. 32, 33

[115] M. Mandad, D. Cohen-Steiner, L. Kobbelt, P. Alliez, and M. Desbrun. Variance-Minimizing Transport Plans for Inter-surface Mapping. *ACM Transactions on Graphics*, 36:14, 2017. 24

[116] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H.-Y. Shum. Full-frame video stabilization with motion inpainting. *IEEE Transactions on pattern analysis and Machine Intelligence*, 28(7):1150–1163, 2006. 27

[117] R. J. McCann. A convexity principle for interacting gases. *Advances in mathematics*, 128(1):153–179, 1997. 17

[118] A. Meka, M. Zollhöfer, C. Richardt, and C. Theobalt. Live intrinsic video. *ACM Transactions on Graphics (TOG)*, 35(4):109, 2016. 27

[119] Q. Mérigot. A Multiscale Approach to Optimal Transport. *Computer Graphics Forum*, 2011. 21, 25

[120] G. Monge. Mémoire sur la théorie des déblais et des remblais. *Histoire de l'Académie Royale des Sciences de Paris*, 1781. 13

[121] G. Montavon, K.-R. Müller, and M. Cuturi. Wasserstein training of restricted boltzmann machines. In *Advances in Neural Information Processing Systems*, pages 3711–3719, 2016. 22, 102, 106

[122] A. Mordvintsev, C. Olah, and M. Tyka. Inceptionism: Going deeper into neural networks, June 2015. 95

[123] J. Morovic and P.-L. Sun. Accurate 3D image colour histogram transformation. *Pattern Recognition Letters*, 24(11):1725 – 1735, 2003. 35

[124] R. D. Neidinger. Introduction to automatic differentiation and MATLAB object-oriented programming. *SIAM Rev.*, 52(3), 2010. 47

[125] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan. Light field photography with a hand-held plenoptic camera. *Computer Science Technical Report CSTR*, 2(11), 2005. 83

[126] I. Omer and M. Werman. Color lines: Image specific color representation. In *CVPR*, 2004. 72

[127] T. Oskam, A. Hornung, R. W. Sumner, and M. Gross. Fast and stable color balancing for images and augmented reality. In *2nd Int. Conf. on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, pages 49 –56, Oct. 2012. 27

[128] N. Papadakis. *Optimal Transport for Image Processing*. Habilitation à diriger des recherches, Université de Bordeaux, Dec. 2015. 21

[129] N. Papadakis, G. Peyré, and E. Oudet. Optimal transport with proximal splitting. *SIAM Journal on Imaging Sciences*, 7(1):212–238, 2014. 21, 22, 39, 61

[130] S. Paris. Edge-preserving smoothing and mean-shift segmentation of video streams. In *Proc. of the 10th European Conference on Computer Vision*, ECCV '08, page 460–473, 2008. 27, 82, 83, 85

[131] S. Paris, S. W. Hasinoff, and J. Kautz. Local Laplacian filters: Edge-aware image processing with a Laplacian pyramid. *ACM Trans. on Graphics (SIGGRAPH)*, pages 68:1–68:12, 2011. 89, 90

[132] H.-S. Park and C.-H. Jun. A simple and fast algorithm for K-medoids clustering. *Expert Systems with Applications*, 36(2, Part 2):3336–3341, Mar. 2009. 66

[133] J. Pennington, R. Socher, and C. D. Manning. Glove: Global Vectors for Word Representation. In *EMNLP*, volume 14, pages 1532–1543, 2014. 57

[134] F. Perazzi, A. Sorkine-Hornung, H. Zimmer, P. Kaufmann, O. Wang, S. Watson, and M. H. Gross. Panoramic video from unstructured camera arrays. *Comp. Graph. Forum*, 34(2):57–68, 2015. 83

[135] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Trans. on Graphics (SIGGRAPH)*, 22(3), 2003. 84

[136] G. Peyré and M. Cuturi. *Computational Optimal Transport*. arXiv:1803.00567, 2018. 13

[137] M. Pharr and G. Humphreys. *Physically Based Rendering, Second Edition: From Theory To Implementation*. 2010. 77

[138] F. Pitié, B. Kent, B. Collis, and A. Kokaram. Localised deflicker of moving images. In *IEEE European Conference on Visual Media Production*, 2006. 27

[139] F. Pitié, A. Kokaram, and R. Dahyot. Automated colour grading using colour distribution transfer. *Computer Vision and Image Understanding*, 2007. 54, 65

[140] F. Pitié, A. C. Kokaram, and R. Dahyot. N-dimensional probablility density function transfer and its application to colour transfer. In *Proceedings of the Tenth IEEE International Conference on Computer Vision - Volume 2*, ICCV '05, pages 1434–1439, Washington, DC, USA, 2005. IEEE Computer Society. 21, 26, 35, 64

[141] T. Pouli and E. Reinhard. Progressive color transfer for images of arbitrary dynamic range. *Computers & Graphics*, 35(1):67–80, Feb. 2011. 66

[142] A. Pratelli. On the equality between monge's infimum and kantorovich's minimum in optimal mass transportation. *Ann. Inst. H. Poincare' Probab. Statist.*, 2005. 15

[143] J. Rabin, J. Delon, and Y. Gousseau. Regularization of transportation maps for color and contrast transfer. In *ICIP*, pages 1933–1936. IEEE, 2010. 21, 52, 66

[144] J. Rabin, G. Peyré, and L. D. Cohen. Geodesic Shape Retrieval via Optimal Mass Transport. In *11th European Conference on Computer Vision*, volume 6315/2010 of *LNCS*, pages 771–784, Heraklion, Crete, Greece, Sept. 2010. Springer. 21

[145] J. Rabin, G. Peyré, J. Delon, and M. Bernot. Wasserstein barycenter and its application to texture mixing. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pages 435–446. Springer, 2011. 26

[146] A. Rav-Acha, P. Kohli, C. Rother, and A. Fitzgibbon. Unwrap mosaics: A new representation for video editing. In *ACM Transactions on Graphics (TOG)*, volume 27, page 17. ACM, 2008. 27

[147] T. u. Rehman, E. Haber, G. Pryor, J. Melonakos, and A. Tannenbaum. 3D nonrigid registration via optimal mass transport on the GPU. *Medical Image Analysis*, 13(6):931 – 940, 2009. 29

[148] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley. Color transfer between images. *IEEE Comput. Graph. Appl.*, 21(5), 2001. 64

[149] E. Reinhard and T. Pouli. Colour spaces for colour transfer. In *Proceedings of the Third international conference on Computational color imaging*, CCIW'11, page 1–15, Berlin, Heidelberg, 2011. Springer-Verlag. 65

[150] RE:Vision. De:flicker v.1.3.0. `http://www.revisionfx.com/products/deflicker/`, 2015. 86

[151] A. Rolet, M. Cuturi, and G. Peyré. Fast dictionary learning with a smoothed wasserstein loss. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 630–638, 2016. 22, 44

[152] R. Rubinstein, M. Zibulevsky, and M. Elad. Efficient implementation of the K-SVD algorithm using batch orthogonal matching pursuit. *Cs Technion*, 40(8):1–15, 2008. 57, 58

[153] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, Nov 2000. 15, 21

[154] M. Ruder, A. Dosovitskiy, and T. Brox. Artistic style transfer for videos. cs/1604.08610, 2016. 27, 95

[155] S. Rusinkiewicz. A new change of variables for efficient BRDF representation. In *Rendering Techniques (Proc. Eurographics Workshop on Rendering)*, June 1998. 33

[156] G. Salton and M. J. McGill. Introduction to modern information retrieval. 1986. 57

[157] R. Sandler and M. Lindenbaum. Nonnegative matrix factorization with earth mover's distance metric. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1873–1880. IEEE, 2009. 22

[158] F. Santambrogio. Optimal transport for applied mathematicians. *Birkäuser, NY*, pages 99–102, 2015. 13

[159] M. W. Schmidt, E. van den Berg, M. P. Friedlander, and K. P. Murphy. Optimizing costly functions with simple constraints: A limited-memory projected quasi-newton algorithm. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 5, 2009. 48

[160] M. A. Schmitz, M. Heitz, N. Bonneel, F. M. N. Mboula, D. Coeurjolly, M. Cuturi, G. Peyré, and J.-L. Starck. Wasserstein dictionary learning: Optimal transport-based unsupervised non-linear dictionary learning. *SIAM Journal on Imaging Sciences (to appear)*, 2018. 50, 52, 56, 57

[161] B. Schmitzer. Stabilized Sparse Scaling Algorithms for Entropy Regularized Transport Problems. working paper or preprint, Oct. 2016. 24, 51

[162] A. Schrijver. On the history of the transportation and maximum flow problems, 2002. 13

[163] Y. Schwartzburg, R. Testuz, A. Tagliasacchi, and M. Pauly. High-contrast computational caustic design. *ACM Trans. Graph.*, 33(4):74:1–74:11, July 2014. 21

[164] A. Secord, W. Heidrich, and L. Streit. Fast primitive distribution for illustration. In *Rendering Techniques (Proc. Eurographics Workshop on Rendering)*, pages 215–226, 2002. 36

[165] V. Seguy and M. Cuturi. Principal geodesic analysis for probability measures under the optimal transport metric. In *Advances in Neural Information Processing Systems*, pages 3312–3320, 2015. 22

[166] E. Shahrian, D. Rajan, B. Price, and S. Cohen. Improving image matting using comprehensive sampling sets. In *IEEE Conf. Comp. Vision and Pattern Recognition*, pages 636–643, 2013. 98

[167] J. Shen, X. Yan, L. Chen, H. Sun, and X. Li. Re-texturing by intrinsic video. *Information Sciences*, 281:726–735, 2014. 27

[168] J. Shen, X. Yang, Y. Jia, and X. Li. Intrinsic images using optimization. In *CVPR*, 2011. 76

[169] L. Shen, P. Tan, and S. Lin. Intrinsic image decomposition with non-local texture cues. In *CVPR*, 2008. 74

[170] L. Shi, F.-C. Huang, W. Lopes, W. Matusik, and D. Luebke. Near-eye light field holographic rendering with spherical waves for wide field of view interactive 3d computer graphics. *ACM Transactions on Graphics (TOG)*, 36(6):236, 2017. 103, 107

[171] R. Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *Ann. Math. Statist.*, 35, 1964. 23

[172] N. Snavely, R. Garg, S. M. Seitz, and R. Szeliski. Finding paths through the world's photos. In *ACM Transactions on Graphics (TOG)*, volume 27, page 15. ACM, 2008. 102, 106

[173] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM transactions on graphics (TOG)*, volume 25, pages 835–846. ACM, 2006. 102, 106

[174] J. Solomon, F. de Goes, G. Peyré, M. Cuturi, A. Butscher, A. Nguyen, T. Du, and L. Guibas. Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Trans. Graph.*, 34(4):66:1–66:11, July 2015. 21, 23, 39, 53

[175] J. Solomon, R. Rustamov, L. Guibas, and A. Butscher. Earth mover's distances on discrete surfaces. *ACM Trans. Graph.*, 33(4):67:1–67:12, July 2014. 21

[176] V. N. Sudakov. *Geometric problems in the theory of infinite-dimensional probability distributions.* American Mathematical Society, Providence, 1979. 16

[177] D. Sun, S. Roth, and M. J. Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. *Int. J. Comput. Vision*, 106(2):115–137, 2014. 86

[178] A. Takatsu. Wasserstein geometry of gaussian measures. *Osaka J. Math.*, 48(4):1005–1026, 12 2011. 20, 67, 68

[179] K. Tang, J. Yang, and J. Wang. Investigating haze-relevant features in a learning framework for image dehazing. In *IEEE Conf. Comp. Vision and Pattern Recognition*, pages 2995–3002, 2014. 90

[180] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016. 49

[181] M. J. Tóth and B. Csébfavi. Shape transformation of multidimensional density functions using distribution interpolation of the Radon transforms. In *Computer Graphics Theory and Applications (GRAPP), 2014 International Conference on*, pages 1–8. IEEE, 2014. 60

[182] M. Turk and A. Pentland. Eigenfaces for Recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991. 56, 58

[183] K. Venkataraman, D. Lelescu, J. Duparré, A. McMahon, G. Molina, P. Chatterjee, R. Mullis, and S. Nayar. Picam: An ultra-thin high performance monolithic camera array. *ACM Trans. Graph.*, 32(6):166:1–166:13, Nov. 2013. 83

[184] A. M. Vershik. Long history of the monge-kantorovich transportation problem. *The Mathematical Intelligencer*, 35(4):1–9, Dec 2013. 13, 15

[185] C. Villani. *Topics in optimal transportation.* American Mathematical Soc., 2003. 14, 22

[186] C. Villani. *Optimal transport: old and new*, volume 338. 2008. 13, 17, 19

[187] C.-M. Wang, Y.-H. Huang, and M.-L. Huang. An effective algorithm for image sequence color transfer. *Mathematical and Computer Modelling*, 44(7–8):608 – 627, 2006. 27

[188] J. Wang, M. F. Cohen, et al. Image and video matting: a survey. *Foundations and Trends® in Computer Graphics and Vision*, 3(2):97–175, 2008. 27

[189] R. Weinstock. *Calculus of variations : with applications to physics and engineering.* Dover books on advanced mathematics. Dover, 1974. Originally published by McGraw-Hill, in 1952. 84

[190] M. Werlberger, T. Pock, and H. Bischof. Motion estimation with non-local total variation regularization. In *IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, USA, June 2010. 86

[191] G.-S. Xia, S. Ferradans, G. Peyré, and J.-F. Aujol. Synthesizing and mixing stationary gaussian texture models. *SIAM Journal on Imaging Sciences*, 7(1):476–508, 2014. 19

[192] L. Yatziv and G. Sapiro. Fast image and video colorization using chrominance blending. *IEEE Trans. on Image Processing*, 15(5):1120 –1129, May 2006. 27, 69

[193] G. Ye, E. Garces, Y. Liu, Q. Dai, and D. Gutierrez. Intrinsic Video and Applications. *ACM Trans. Graph. (SIGGRAPH)*, 33(4), 2014. 27, 71, 77, 78, 79, 82

[194] Q. Zhao, P. Tan, Q. Dai, L. Shen, E. Wu, and S. Lin. A closed-form solution to retinex with nonlocal texture constraints. *PAMI*, 34(7):1437–1444, 2012. 74, 76, 77, 78, 87, 97

7 Curriculum Vitae

Nicolas Bonneel
LIRIS
23-25, Avenue Pierre de Coubertin
69622 Villeurbanne Cedex (France)
nicolas.bonneel@liris.cnrs.fr
`https://perso.liris.cnrs.fr/nbonneel/`

Junior 35-year-old CNRS researcher in computer graphics, interested in optimal mass transportation and its applications to graphics, vision and imaging problems. Published seven papers in A* venues since 2006, including ICCV and six first authored Siggraph/Siggraph Asia papers. Obtained and coordinates a 250k€ ANR JCJC grant on optimal transport.

Experience

2014-	**CNRS, Junior researcher (CR2) at LIRIS (Lyon).** Research on geometry and video processing.
2012-2014	**Harvard University (Cambridge, USA), Post-Doctoral Researcher.** With Hanspeter Pfister. Research on image and video processing using Optimal Transport theory, and teaching assistant.
2011	**INRIA Nancy - Grand Est (France), Post-Doctoral Researcher.** With Bruno Levy. Research on geometry processing.
2010	**University of British Columbia (Vancouver, Canada), Post-Doctoral Researcher.** With Michiel van de Panne. Research on texture synthesis and Optimal Transport theory.
2006-2009	**INRIA Sophia Antipolis (France), Ph.D. student** *Audio and Visual Rendering with Perceptual Foundations.* Advisor: George Drettakis.
July 2008	**Massachusetts Institute of Technology (MIT) (Cambridge, USA), Invited graduate student** With Frédo Durand. Work on reflectance fitting via Optimal Transport.
2006	**Université de Toulouse Paul Sabatier, M.Sc. student** *Rendu, illumination et modèles à base de points.* Advisors: Mathias Paulin, Gaël Guennebaud.

Education

2009	**Ph.D. in Computer Science** Université de Nice - Sophia Antipolis.
2006	**Master in Computer Science** Université Paul Sabatier, Toulouse.
2006	**Engineering Diploma in Applied Mathematics** Institut National des Sciences Appliquées (INSA), Toulouse.

Teaching Experience

Lectures	**Numerical methods for Image Synthesis**, École Normale Supérieure Lyon, Master's students (2016, 2017) **Visualization**, Université Lyon 1, Master's students (2016, 2017) **Computer Graphics**, École Centrale Lyon, Master's students (2014–2017) **Computer Graphics**, Université Lyon 2, Master's students (2015) **Geometry Processing** (occasional lectures), Université Lyon 1, Master's students (2014, 2015)
Teaching Assistant	**CS109 Data Science**, Harvard University, Undergrads (2013) **CS171 Visualization**, Harvard University, Undergrads (2012, 2013) **CS205 Computational Sciences**, Harvard University, Master's students (2012)
Tutorials/ Labs	**Algorithmic, procedural programming**, Université Lyon 1, Undergrads (2014, 2015) **Operations Research**, Université Lyon 1, Undergrads (2015,2016) **Computer Graphics**, Université Lyon 1, Undergrads (2014) **Programming in C**, Université de Nancy, Undergrads (2011) **Computer Graphics**, Polytech'Nice, Master's students (2009)
Ph.D Supervision	Agathe Herrou (2018-). Beatrix-Emőke Fülöp-Balogh (2017-) Matthieu Heitz (2016-). Vincent Léon (2014–2016).
M.Sc Supervision	Abir Zendagui (2018), 2nd year Martin Guy (2018), 2nd year Yoann Coudert-Osmont (2018), 1st year Agathe Herrou (2017), 2nd year Emmanuelle Chapoulie (2009), 1st year

Professional Activities

Associate Editor

The Visual Computer (Springer) – since 2016.

International Program Committee

Graphics Interface (2018), ACM SIGGRAPH Asia Posters and Shorts (2016, 2017), IEEE International Conference on Computational Photography (2016, 2017), Pacific Graphics (2013–2017), Workshop on Self-Organizing Maps and Learning Vector Quantization, Clustering and Data Visualization (2017), CAD/Graphics (2015), ACM Web3D (2012), Eurographics Symposium on Rendering (2011)

PhD committee

Hristina Hristova (2017): *Example-guided image editing* (Advisors: Olivier Le Meur, Rémi Cozot)

Guillaume Tartavel (2015): *Modèles variationnels pour les textures : applications à la synthèse et à la restauration* (Advisors: Gabriel Peyré and Yann Gousseau)

Organizing committees

Journées AFIG (2015): Local organizing committee. National conference in computer graphics, to be held in Lyon in Nov. 2015

Eurographics (2017): Local organizing committee of Eurographics, the third major computer graphics conference (after SIGGRAPH and SIGGRAPH Asia), to be held in Lyon in 2017. Co-chair of the Eurographics Doctoral Consortium, a co-located event which promotes interactions between young PhD students and renown researchers.

Invited talks

2018	SIAM Conference on Imaging Sciences in Bologna (Italy)
2016	Fédération Informatique Fondamentale in Lyon (France)
2016	CAVALIERI workshop on optimal transport, Paris (France)
2016	PICOF workshop on inverse problems in Autran (France)
2015	SMATI seminar at the applied math lab MAP5 in Paris (France)
2015	Lecture on video processing at Kyoto University (Japan)
2014	Workshop on optimal transport at Paris-Dauphine (France).
2014	Workshop on optimal transport at the University of Toulouse (France).
2014	Seminar at the Max Planck Institute, Saarbrucken (Germany)
2014	Seminar at MIT, Cambridge (Massachusetts)
2011	Seminar at Disney Research, Zurich (Switzerland)
2011	Seminar at INRIA, Grenoble (France)
2011	Seminar at INRIA, Bordeaux (France)
2011	Workshop on computer graphics (LIMA project) at the University of Lyon (France).
2010	Seminar at Photometria/TAAZ (virtual makeover company by David Kriegman), San Diego (USA).
2010	Seminar at Disney Research, Glendale (California).
2009	Seminar at INRIA, Grenoble (France)

Awards and Grants

Awarded an ANR JCJC Grant of 250k€ on inverse problems in optimal transport ("ROOT" project, 2016-2021).

Local coordinator of an ANR Générique Grant of 120k€ (locally) on the study of the space of light paths for rendering ("CALiTrOp" project, 2017-2022).

Best paper award at the Vision Modeling and Visualization workshop, out of 43 accepted papers and 82 submissions ([25], VMV 2010)

IEEE BioVis Data contest honorable mention, for a protein visualization tool ([37], 2013)

Publications

Journal/Conference Name	Impact Factor	CORE Ranking	SCImago SJR
ACM Trans. on Graphics	4.384	A*	Q1
IEEE Trans. on Visu. and Comp. Graph.	3.08	A	Q1
SIAM J. Imaging Sciences	2.36		Q1
Computer Graphics Forum	2.05	B	Q1/Q2
J. of Mathematical Imaging and Vision	1.93	B	Q1/Q2
Experimental Brain Research	1.81		Q2
Computer and Graphics	1.20	B	Q2
ACM Trans. on Applied Perception	0.96	B	Q2–Q4
ICCV		A*	
SIGGRAPH		A*	
SIGGRAPH Asia		"journal published"	
Pacific Graphics		A	
ACM Symp. Interactive 3D Graph. and Games		B	

International Journals

[1] **N. Bonneel**, D. Coeurjolly, P. Gueth, and J.-O. Lachaud, "Mumford-Shah Mesh Processing using the Ambrosio-Tortorelli Functional," *Computer Graphics Forum (Pacific Graphics)*, vol. 37, no. 7, 2018.

[2] M. A. Schmitz, M. Heitz, **N. Bonneel**, F. M. N. Mboula, D. Coeurjolly, M. Cuturi, G. Peyré, and J.-L. Starck, "Wasserstein Dictionary Learning: Optimal Transport-based unsupervised non-linear dictionary learning," *SIAM Journal on Imaging Sciences*, vol. 11, no. 1, 2018. 50, 52, 56, 57

[3] **N. Bonneel**, J. Tompkin, D. Sun, O. Wang, K. Sunkvalli, S. Paris, and H. Pfister, "Consistent Video Filtering for Camera Arrays," *Computer Graphics Forum (Eurographics 2017)*, vol. 36, no. 2, 2017. 82, 92, 93, 95, 96

[4] **N. Bonneel**, B. Kovacs, S. Paris, and K. Bala, "Intrinsic Decompositions for Image Editing," *Computer Graphics Forum (Eurographics State of the Art Reports 2017)*, vol. 36, no. 2, 2017. 71, 76, 77, 99

[5] **N. Bonneel**, G. Peyré, and M. Cuturi, "Wasserstein Barycentric Coordinates: Histogram Regression Using Optimal Transport," *ACM Trans. on Graphics (SIGGRAPH)*, vol. 35, no. 4, 2016.

[6] **N. Bonneel**, J. Tompkin, K. Sunkavalli, D. Sun, S. Paris, and H. Pfister, "Blind Video Temporal Consistency," *ACM Trans. on Graphics (SIGGRAPH Asia)*, vol. 34, no. 6, 2015. 82, 85, 90

[7] V. Léon, **N. Bonneel**, L. Guillaume, and J.-P. Vandeborre, "Continuous Semantic Description of 3D Meshes," *Computer and Graphics (Proc. of CAD/Graphics 2015).*, vol. 54, pp. 47–56, 2016.

[8] **N. Bonneel**, J. Rabin, G. Peyré, and H. Pfister, "Sliced and Radon Wasserstein Barycenters of Measures," *Journal of Mathematical Imaging and Vision*, vol. 51, no. 1, 2015.

[9] **N. Bonneel**, K. Sunkavalli, J. Tompkin, D. Sun, S. Paris, and H. Pfister, "Interactive Intrinsic Video Editing," *ACM Trans. on Graphics (SIGGRAPH Asia)*, vol. 33, no. 6, 2014. 71, 75, 77, 82, 87

[10] F. Shen, K. Sunkavalli, **N. Bonneel**, S. Rusinkiewicz, H. Pfister, and X. Tong, "Time-lapse Photometric Stereo and Applications," *Computer Graphics Forum (Pacific Graphics)*, vol. 33, no. 7, 2014.

[11] J. Mercer, B. Pandian, A. Lex, **N. Bonneel**, and H. Pfister, "Mu-8: Visualizing Differences between Proteins and their Families," *BMC Proceedings*, vol. 8, no. Suppl 2, 2014.

[12] **N. Bonneel**, K. Sunkavalli, S. Paris, and H. Pfister, "Example-Based Video Color Grading," *ACM Trans. on Graphics (SIGGRAPH)*, vol. 32, no. 4, 2013.

[13] **N. Bonneel**, M. van de Panne, S. Paris, and W. Heidrich, "Displacement Interpolation Using Lagrangian Mass Transport," *ACM Trans. on Graphics (SIGGRAPH Asia)*, vol. 30, no. 6, 2011.

[14] M. Cabral, **N. Bonneel**, S. Lefebvre, and G. Drettakis, "Relighting Photographs of Tree Canopies," *IEEE Trans. on Visualization and Computer Graphics*, vol. 17, no. 10, 2011.

[15] **N. Bonneel**, C. Suied, I. Viaud-Delmon, and G. Drettakis, "Bimodal perception of audio-visual material properties for virtual environments," *ACM Trans. on Applied Perception*, vol. 7, no. 1, 2010.

[16] **N. Bonneel**, S. Paris, M. van de Panne, F. Durand, and G. Drettakis, "Single Photo Estimation of Hair Appearance," *Computer Graphics Forum*, vol. 28, no. 4, 2009.

[17] C. Suied, **N. Bonneel**, and I. Viaud-Delmon, "Integration of auditory and visual information in the recognition of realistic objects," *Experimental Brain Research*, vol. 194, no. 1, 2009.

[18] **N. Bonneel**, G. Drettakis, N. Tsingos, I. Viaud-Delmon, and D. James, "Fast Modal Sounds with Scalable Frequency-Domain Synthesis," *ACM Trans. on Graphics (SIGGRAPH)*, vol. 27, no. 3, 2008.

International Conferences with Committee

[19] B.-E. Fülöp-Balogh, **N. Bonneel**, and J. Digne, "Correcting Motion Distortions in Time-of-Flight Imaging," in *ACM Siggraph Conference on Motion, Interaction and Games (Short presentation).*

[20] M. A. Schmitz, M. Heitz, **N. Bonneel**, F. M. N. Mboula, D. Coeurjolly, M. Cuturi, G. Peyré, and J.-L. Starck, "Optimal transport-based dictionary learning and its application to euclid-like point spread function representation," in *Proceedings SPIE Wavelets and Sparsity*, 2017.

[21] V. Léon, V. Itier, **N. Bonneel**, G. Lavoué, and J.-P. Vandeborre, "Semantic correspondence across 3D models for example-based modeling," in *10th Eurographics Workshop on 3D Object Retrieval (3DOR)*, April 2017.

[22] E. Levinkov, J. Tompkin, **N. Bonneel**, S. Kirchhoff, B. Andres, and H. Pfister, "Interactive Multicut Video Segmentation," in *Pacific Graphics (short paper)*, October 2016.

[23] M. Keuper, E. Levinkov, **N. Bonneel**, G. Lavoué, T. Brox, and B. Andres, "Efficient Decomposition of Image and Mesh Graphs by Lifted Multicuts," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[24] B. Levy and **N. Bonneel**, "Variational Anisotropic Surface Meshing with Voronoi Parallel Linear Enumeration," in *Proceedings of the 21st International Meshing Roundtable (IMR'12)*, October 2012.

[25] **N. Bonneel**, M. van de Panne, S. Lefebvre, and G. Drettakis, "Proxy-Guided Texture Synthesis for Rendering Natural Scenes," in *Proceedings of Vision, Modeling, and Visualization 2010 (VMV'10)*, November 2010. 124

[26] D. Grelaud, **N. Bonneel**, M. Wimmer, M. Asselot, and G. Drettakis, "Efficient and practical audio-visual rendering for games using crossmodal perception," in *I3D '09: Proceedings of the 2009 symposium on Interactive 3D graphics and games*, ACM, February 2009.

[27] I. Viaud-Delmon, F. Znaïdi, **N. Bonneel**, C. Suied, O. Warusfel, K.-V. N'Guyen, and G. Drettakis, "Auditory-visual virtual environments to treat dog phobia," in *Proceedings of the 7th ICDVRAT-International Conference on Disability, Virtual Reality and Associated Technologies*, 2008.

[28] G. Drettakis, **N. Bonneel**, C. Dachsbacher, S. Lefebvre, M. Schwarz, and I. Viaud-Delmon, "An Interactive Perceptual Rendering Pipeline using Contrast and Spatial Masking," in *Rendering Techniques (Proceedings of the Eurographics Symposium on Rendering)*, Eurographics, June 2007.

[29] T. Moeck, **N. Bonneel**, N. Tsingos, G. Drettakis, I. Viaud-Delmon, and D. Alloza, "Progressive perceptual audio rendering of complex scenes," in *I3D '07: Proceedings of the 2007 symposium on Interactive 3D graphics and games*, ACM, April 2007.

Education

[30] **N. Bonneel**, *Audio and Visual Rendering with Perceptual Foundations*. PhD thesis, Université de Nice - Sophia Antipolis, September 2009.

[31] **N. Bonneel**, "Rendu, Illumination et Modèles à Base de Points," Master's thesis, Université de Toulouse - Paul Sabatier, June 2006.

Other (Workshops, national conferences, technical reports, posters)

[32] M. A. Schmitz, M. Heitz, **N. Bonneel**, F. M. N. Mboula, D. Coeurjolly, M. Cuturi, G. Peyré, and J.-L. Starck, "Wasserstein Dictionary Learning: Optimal Transport-based unsupervised non-linear dictionary learning." Arxiv preprint 1708.01955, August 2017.

[33] **N. Bonneel**, "Le transport optimal pour des applications en informatique graphique," in *Interstices*, May 2017.

[34] M. Kurt, G. Ward, and **N. Bonneel**, "A Data-Driven BSDF Framework." ACM SIGGRAPH (Poster), July 2016.

[35] V. Léon, **N. Bonneel**, G. Lavoué, and J.-P. Vandeborre, "Géodésiques sémantiques pour la description et la labélisation automatique de parties," in *Journées de l'Association Française d'Informatique Graphique*, November 2014.

[36] G. Ward, M. Kurt, and **N. Bonneel**, "Reducing Anisotropic BSDF Measurement to Common Practice," in *Proceedings of the 2nd Eurographics Workshop on Material Appearance Modeling: Issues and Acquisition*, MAM'14, June 2014.

[37] J. Mercer, B. Pandian, **N. Bonneel**, A. Lex, and H. Pfister, "Mu-8: Visualizing Differences between a Protein and its Family." IEEE BioVis Data Contest entry, October 2013. 124

[38] G. Ward, M. Kurt, and **N. Bonneel**, "A Practical Framework for Sharing and Rendering Real-World Bidirectional Scattering Distribution Functions," Tech. Rep. LBNL-5954E, Lawrence Berkeley National Laboratory technical report, October 2012.

[39] C. Suied, **N. Bonneel**, and I. Viaud-Delmon, "The role of auditory-visual integration in object recognition," *Journal of the Acoustical Society of America (Abstract)*, vol. 123, no. 5, 2008.

[40] C. Suied, **N. Bonneel**, and I. Viaud-Delmon, "Integration of auditory and visual information in fast recognition of realistic objects." Frontiers in Neuroscience (Poster), 2008.

[41] C. Suied, **N. Bonneel**, and I. Viaud-Delmon, "Role of semantic vs spatial congruency in a bimodal go/no-go task." 8th Internation Multisensory Research Forum (Poster), 2007.