Universitat de les Illes Balears

MASTER'S THESIS

Primal-dual Unfolding for Underwater Digital Images

Frank William Hammond Espinosa

Master's Degree in Intelligent Systems (MUSI)
Specialization in Artificial Intelligence.

Centre for Postgraduate Studies

Academic year 2024/25

Primal-dual Unfolding for Underwater Digital Images

Frank William Hammond Espinosa

MASTER'S THESIS

Centre for Postgraduate Studies

University of the Balearic Islands

Academic year 2024/25

Keywords:

Underwater imaging, Applied mathematics

Tutor(s): Julia Navarro Oliver and Ana Belén Petro Balaguer.

Primal-dual Unfolding for Underwater Digital Images

Frank William Hammond Espinosa
Tutor(s): Julia Navarro Oliver and Ana Belén Petro Balaguer
Thesis of the Master's degree in Intelligent Systems (MUSI)
University of the Balearic Islands, 07122 Palma, Illes Balears, Spain
frank.william.hammond@gmail.com

***Abstract*—Lorem ipsum etc.**

I. Introduction and related works

Underwater image processing is of great interest for its applications on marine robotics [10] or marine biology [25]. It presents a unique set of challenges due to two physical phenomena that govern the image formation process in aquatic media. As light propagates through the water, it loses its intensity in a process called *absorption*, at a significant rate that is dependent both on the condition of the water and the light wavelength. Specifically, longer wavelengths tend to be absorbed more rapidly, resulting in underwater images often appearing bluish or greenish to the human eye. In addition to losing intensity, light can change directions when colliding with suspended particles in the medium. This process, known as *scattering*, results in a loss of contrast and hazy appearance. Inverting these two processes allows the recovery of a much richer image that is better suited for subsequent imaging tasks such as annotation [23] or classification [28]. Such inversion is particularly challenging and therefore traditional, general approaches like histogram equalization or Retinex models are insufficient without modifications specific to the underwater environment [30].

McGlamery [26] and Jaffe [17] introduced what is now the most widely adopted mathematical framework for this problem, now known as the Undewater Image Formation Model (UIFM). Modern approaches often rephrase a simplified version of the original models as *dehazing* problems [11]. This simplified version of the UIFM does not take into account its *forward scattering* component, but still produces acceptable results. Most importantly, it yields equations very similar to those of the Atmospheric Scattering Model (ASM), which is mostly associated to dehazing problems, and therefore many techniques used for atmospheric dehazing can be adapted for the aquatic environment.

In order to solve these dehazing problems, many approaches first try to estimate the *background light* (a.k.a. *atmospheric light* in dehazing settings) of the scene via some heuristic, which often uses some sort of statistical measure of the intensities in one or several color channels of the image. A simple approach consists of using the brightest pixel in the image, but due to its brittleness it has been superseded by more sophisticated techniques. Popular approaches are based on the dark channel intensity histogram [15], a quad-tree algorithm that searches the brightest areas in the image [18, 20, 30], on the brightest local intensity minimum [6], or modifications of the above and combinations thereof.

Once the background light has been approximated, an array of techniques are often used to estimate the *transmission map*. Many traditional approaches ([9, 11, 30]) are based on the classical Dark Channel Prior [15] or versions of it, such as the Red Channel Prior [11] or the Underwater Dark Channel Prior [9]. Other techniques are possible too, such as the one presented in [21], which minimizes information loss due to RGB encoding. Having estimated the background light and transmission map, it is then possible to analytically invert the simplified UIFM and obtain an estimate for the scene radiance, which effectively corresponds to the undegraded image.

Machine Learning (ML) approaches have been successfully used for these tasks too. A classification of different techniques used for dehazing is presented in [13]. Among the wide variety of classes collected, two follow the paradigm described above: those that learn the transmission map only, and those that jointly learn the transmission map and the background light.

In [5], authors estimate the background light by selecting the brightest pixel below a certain threshold, and then feed the unaltered input into a small neural network they coined *DehazeNet*, which then outputs a coarse transmission map. After refining this map with a guided filter, they invert the ASM to obtain the scene radiance. A similar technique is used in [24]. The algorithm in [29] is also alike, but instead of yielding the tranmission map directly, they use the output of the penultimate layer of their nework as the transmission map, and then pass the resulting radiance estimation through a final, linear layer. A "neural-network-first" approach that is still based on the ASM can be found in [14]. Authors propose two similar encoder-decoder architectures in which a single encoder feeds either two or three decoders with different purposes: one estimates the atmospheric light, another the transmission map and, when present, the third approximates the scene radiance directly. In both configurations, the final output is computed via the ASM solely based on the atmospheric light and transmission map estimates. However, in the three-decoder architecture, the output of the third decoder is used as assistance during training, since, according to the authors, this improves performance for severely degraded images.

Not all methods are based directly on inverting a physical model. These are sometimes called *model-free* algorithms. Examples of this are fusion-based algorithms [2, 3], which compute several auxiliary images from the input - each intended to enhance a certain aspect of it - and then combine those to form a definitive output. The way of computing these auxiliary images often does incorporate aspects of the UIFM. Most ML-based models presented in [13] that do not fall under the two previously mentioned categories can also be considered *model-free*. Some draw inspiration from either the ASM [19] or from Retinex theory [22] to justify their models' architectures, but ultimately do not leverage a physical model that explains image formation. However, their lack of interpretability is often somewhat compensated by exhibiting excellent performance.

Algorithm *unfolding*, also called algorithm *unrolling*, presents a way of leveraging physical models while using neural networks in a way that allows them to be trained end-to-end, yielding fully explainable models that can still achieve competitive results. An excellent exposition of the core ideas about unfolding and a collection of illustrative approaches is presented in [27]. Briefly, it consists in first solving an optimization problem via an iterative approach, and then replacing some part of the iterative scheme by a neural network. The seminal work that introduced the technique [12] illustrates it quite well. It was performed in the context of sparse coding, which can be translated into minimizing the functional

$$\frac{1}{2}\|X - W_d Z\|_2^2 + \alpha\|Z\|_1$$

with respect to $Z \in \mathbb{R}^m$ for a given $X \in \mathbb{R}^n$ and a maximally ranked matrix of choice $W_d \in \mathcal{M}_{n\times m}(\mathbb{R})$, usually with $m \gg n$. A popular iterative scheme for solving this problem, the Iterative Shrinking and Tresholding Algorithm (ISTA), consists of the following iterates:

$$z_0 = 0, \qquad z_{n+1} = S_\lambda\left(W_t z_n + W_e X\right),$$

where W_t and W_e are fixed matrices that can be analytically computed from W_d, and S_λ is a function also with an analytically closed form that depends only on the parameter $\lambda > 0$ (said closed forms are not included here for brevity). One of the main contributions of [12] was the Learned ISTA. In that algorithm, the parameters λ, W_t and W_e are the output of a neural network that takes as inputs different values of X, and the iterative scheme above is repeated for a fixed number of iterations. During training, the output is evaluated with the target functional for a pre-fixed matrix W_d, enabling end-to-end training of the network for optimizing the parameters. This resulted in a fully explainable model that achieved impressive performance at the time, which in this case meant faster convergence to an optimum value of Z.

Since then, many similar approaches have been developed in a variety of fields. Iterative schemes containing some sort of proximity operator are a natural fit for unfolding, since the proximity operator itself can be replaced by a neural network in-place. This is particularly useful when the proximity operator derives from a regularization function, since those are often chosen somewhat arbitrarily in inverse imaging problems, and learning the regularization prior from actual data is both practical and mathematically elegant. The Chambolle-Pock algorithm, also known as primal-dual hybrid gradient algorithm, offers a uniquely direct way of achieving this structure, so much so that it is possible to perform this substitution in a fully general manner, as authors in [1] suggest.

In this work, heuristics from the traditional approaches outlined above are used to estimate the background light and transmission map - concretely, the quad-tree and RCP methodologies. Then, a variational problem is posed for the full UIFM, as opposed to most traditional approaches which use only the simplified version. The problem is theoretically studied, ensuring partial well-posedness for a wide family of regularizations, and an iterative scheme is drawn with the Chambolle-Pock algorithm. Finally, said scheme is unfolded by substituting part of the regularization by a neural network, which is trained end-to-end with the UIEB dataset. Two neural network architectures are used to this end: a classical UNet, and a novel one baptized as AUNet (Additive UNet), which is a simple modification to the UNet that changes concatenation on the skip connections by additions.

II. Mathematical model

In this section, all of the mathematical machinery and notation needed to understand and formalize the problem is presented (section II-A), the physical equations are introduced and manipulated for convenience (section II-B), and the problem is formalized and theoretically studied (section II-C).

A. Theoretical grounding and notation

Although the mathematical exposition in this section is, for the most part, either self-contained or specifically referenced, a certain degree of familiarity with modern mathematical analysis is assumed. A minimum familiarity with digital imaging is also assumed.

Although not recommended, the reader that is either already very familiar with higher mathematical analysis and convex optimization, or too unfamiliar with it, may skip this section and read section II-B and below.

1) Image representations: Two representations of digital images will be used in this work, the equivalences between which will be described below.

The most common representation among computing frameworks is that of multidimensional arrays, sometimes also called *tensors*. We shall refer to this one as the **discrete** representation. Since *pytorch* will be used to implement the algorithms described in this text, the standard 4D representation will be used; that is, images will always have the *Shape* (B, C, H, W), where B is the batch size (1 for a single image), C is the number of channels (3 for an RGB image, 1 for a grayscale one), and H and W are the height and width of the image, respectively.

The other representation used will be called **continuous**[1]. It will be the preferred one in the mathematical formulae of

[1] The term *continuous* here refers to the domain of the representation. In fact, the functions involved will almost never be continuous.

this text. The idea is to think of images as either scalar or three-dimensional vector fields for grayscale or RGB images, respectively, defined on a rectangular domain $\Omega \subseteq \mathbb{R}^2$. Said fields are always piecewise constant on right-semiclosed unit squares with sides parallel to the coordinate axes.

In order to be able to comfortably extend continuous operations for images, it is useful to center the domain Ω with respect to the origin. Concretely, for a batched image I with shape (B, C, H, W), B fields are defined on the same domain Ω. Namely, $I^1, \dots, I^B \colon \Omega \to \mathbb{R}^C$, where the domain Ω is given by

$$\Omega = (-\lceil \tilde{w} \rceil - 1, \lfloor \tilde{w} \rfloor] \times (-\lceil \tilde{h} \rceil - 1, \lfloor \tilde{h} \rfloor],$$

with $\overline{w} = \frac{W-1}{2}$ and $\overline{h} = \frac{H-1}{2}$. The relationship between the two representations is given by

$$I[b][c][h][w] = I^{b+1}_{c+1}(w - \lceil \tilde{w} \rceil, h - \lceil \tilde{h} \rceil) \tag{1}$$

where the left term follows a standard 0-indexed array notation, and the right is standard mathematical function notation. Then, the functions I^b_c are extended to the rest of Ω by imposing that they be constant on all right-semiclosed squares $(x-1, x] \times (y-1, y]$.

It is also possible to extend the domain of the continuous representations to all of $\mathbb{R}^2$ by simply setting them to 0 outside of Ω. Reciprocally, for a given set of functions $I^1, \dots, I^B \colon \mathbb{R}^2 \to \mathbb{R}^C$, it is possible to obtain a discrete representation of a digital image by fixing the desired dimensions H and W and sampling as in eq. (1).

It is immediate to see that this identification between the two representations respects pointwise operations (in particular, function sums and products). It is also possible to see that, for an appropriate translation, the convolution defined in *pytorch* with a kernel g coincides with the continuous convolution with (the continuous version of) the inverted kernel $\overline{g}(x) = g(-x)$.

2) L^p spaces: It will be useful to utilize mathematical terminology and machinery that is best exposed in function spaces. The most important and suitable function spaces that will be employed are the L^p spaces.

Definition 1. *Let Ω be a Borel measurable subset of $\mathbb{R}^n$ for some $n \in \mathbb{Z}^+$, and μ the Lebesgue measure on the Borel sets of $\mathbb{R}^n$.*[2] *For a given measurable function $f \colon \Omega \to \overline{\mathbb{R}}$, where $\overline{\mathbb{R}}$ is the extended real line equipped with its usual topology, define the quantity*

$$\|f\|_p = \left(\int_\Omega |f|^p \, d\mu \right)^{\frac{1}{p}}$$

for each $p \in [1, +\infty)$. Define the set of p-integrable functions as

$$\mathcal{L}^p(\Omega) = \left\{ f \colon \Omega \to \overline{\mathbb{R}} \mid \|f\|_p < +\infty \right\}.$$

It can be shown that $\mathcal{L}^p(\Omega)$ is a normed vector space when equipped with $\|\cdot\|_p$ as a norm.

Finally, define $\boldsymbol{L^p(\Omega)}$ by identifying functions in $\mathcal{L}^p(\Omega)$ that coincide almost everywhere with respect to μ and equipping the resulting equivalence classes with the norm of any of their elements. It is possible to show that $L^p(\Omega)$ is a separable Banach space for any p, and that it is a Hilbert space for $p = 2$.

It is trivial to see that the continuous representation of a digital image I with domain Ω is always an element of $L^p(\Omega)$ for any $p \in [1, +\infty)$.

3) Functional and convex analysis: It is possible to develop in a surprising generality a number of optimization algorithms which can later be applied for image processing problems. The continuous representation of digital images described above will allow the work with operators defined on function spaces in a more abstract fashion than with multidimensionals arrays. By doing so, the involved operators and formulae obtained often have much cleaner closed forms.

In this section, a few relevant definitions are presented. Most importantly, the direct method is introduced and proved. This theorem will be employed later to show the well-posedness of the presented minimization problems. Some results are given without proof; the interested reader can consult chapters 1 to 3 of [7].

Definition 2. *A normed vector space X is said to be a* ***Banach space*** *if it is complete under the metric induced by its norm. A* ***functional*** *on X is a function $F \colon X \to \mathbb{R} \cup \{+\infty\}$.*

Definition 3. *The* ***dual space*** *of X is defined as the set of all continuous linear functionals on X with codomain on $\mathbb{R}$, where continuity is to be interpreted with respect to the topology induced on X by its norm and the usual topology in $\mathbb{R}$. We denote it as X^*. It can be shown that X^* is a Banach space when equipped with the operator norm*

$$\|x^*\|_{X^*} = \sup_{x \in X, \|x\| = 1} \|x^*(x)\|.$$

We say that a given sequence $\{x_n\}_{n \in \mathbb{N}} \subseteq X$ ***converges weakly*** *to a given $x \in X$ if the sequence $\{f(x_n)\}_{n \in \mathbb{R}}$ converges (in $\mathbb{R}$) to $f(x)$ for every $f \in X^*$.*

It is immediate to check that the usual convergence of a sequence in X implies its weak convergence to the same limit.

Definition 4. *A given functional F on a Banach space X is said to be* ***weakly lower semicontinuous*** *if, for each weakly converging sequence $\{x_n\} \subseteq X$ with weak limit $\lim_n x_n \in X$,*

$$F(\lim_n x_n) \leq \liminf_n F(x_n).$$

Importantly, the norm of X is a weakly lower semicontinuous functional.

Definition 5. *A given functional F on X is said to be* ***convex*** *whenever the condition*

$$F(x + (1-\lambda)y) \leq \lambda F(x) + (1-\lambda) F(y)$$

holds for any $x, y \in X$ and any $\lambda \in (0, 1)$. If, additionally, the inequality is strict whenever $x \neq y$, F is said to be ***strictly convex****.*

[2] Properly defining the Borel sets and the Lebesgue measure is out of the scope of this work. All rectangles, either open, closed or semiclosed are Borel measurable subsets of $\mathbb{R}^2$. All integrals with respect to the Lebesgue measure coincide with their classical Riemann integral counterpart, whenever the latter is defined.

Definition 6. *A functional F is said to be* ***coercive*** *if, for every sequence $\{x_n\} \subseteq X$ such that $\lim_n \|x_n\| = +\infty$, then $\lim_n F(x_n) = +\infty$. It is said to be* ***proper*** *if there exists some $x \in X$ such that $F(x) < +\infty$.*

Definition 7. *Let X be a Banach space. The* ***bidual*** *space of X, denoted by X^{**}, is the dual space of its dual space, that is, $X^{**} = (X^*)^*$. There is a canonical linear mapping from X to X^{**} given by*

$$x \mapsto (x^* \mapsto x^*(x))$$

which can be shown to always be isometric and therefore injective. X is said to be ***reflexive*** *whenever the canonical mapping above is also surjective.*

It is a direct consequence of the Banach-Alaoglu theorem (which is omitted here for brevity) that every bounded sequence in a reflexive Banach space has a weakly converging subsequence.

Theorem 1 (The direct method)**.** *Let X be a reflexive Banach space and let F be a proper, coercive and weakly lower semicontinuous functional on X. Then, the minimization problem*

$$\min_{x \in X} F(x)$$

admits a solution. Moreover, if F is strictly convex, the solution is unique.

Proof. For the existence, note that since F is proper, $\inf_{x\in X} F(x) < +\infty$. It is therefore possible to construct a sequence $\{x_n\}_n \subseteq X$ such that $F(x_n) < +\infty$ for all n and $F(x_n) \to \inf_{x\in X} F(x)$. Since $\{F(x_n)\}_n$ is a converging sequence of real numbers, it is bounded. This, together with the coerciveness of F, implies that $\{x_n\}_n$ is bounded in norm. Since X is reflexive, it has a weakly converging subsequence. Denote the subsequence by $\{\overline{x}_n\}_n$ and its limit by $\overline{x} \in X$. Then, utilizing the weak lower semicontinuity of F:

$$\inf_{x\in X} F(x) \leq F(\overline{x}) = F(\lim_n \overline{x}_n) \leq \liminf_n F(\overline{x}_n) = \inf_{x\in X} F(x).$$

This shows that $\overline{x}$ is a minimizer of F. The uniqueness for the strict convexity is immediate by contradiction. □

Finally, a series of "bespoke" lemmas with simple proofs will be useful further ahead.

Lemma 1. *The sum of two weakly lower semicontinuous functionals is weakly lower semicontinuous.*

Proof. Immediate using the definition and the fact that $\liminf_n a_n + \liminf_n b_n \leq \liminf_n a_n + b_n$. □

Lemma 2. *Let X and Y be Banach spaces. Let $F\colon Y \to \mathbb{R} \cup \{+\infty\}$ be a weakly lower semicontinuous functional, and $K\colon X \to Y$ a continuous function. Then, $F \circ K$ is weakly lower semicontinous.*

Proof. Since K is continuous, $K(\lim_n x_n) = \lim_n K(x_n)$ for any converging sequence $\{x_n\}_n \subseteq X$. The result is immediate by applying the definition of weak lower semicontinuity. □

4) Convex optimization: Although necessary to establish the existence of solution, the direct method offers no way to actually compute the minimum of a posed problem. Other techniques, which often involve iterative algorithms, are useful for this. Notably, the Chambolle-Pock primal-dual algorithm will be used in this work. The rest of this section is devoted to briefly present the concepts necessary to enunciate it. For a more detailed explanation, the reader is referred to chapters 3 to 7 and section 8.4 of [7].

Definition 8. *Let X be a Banach space and F be a proper functional on X. Define $F^*\colon X^* \to \mathbb{R} \cup \{+\infty\}$ by*

$$F^*(x^*) = \sup_{x\in X} x^*(x) - F(x).$$

This function is called the ***Fenchel conjugate*** *or* ***convex conjugate*** *of F.*

If Y is another Banach space, and $A\colon X \to Y$ is a continuous linear operator, then a linear operator $A^*\colon Y^* \to X^*$ is defined by

$$(A^*y^*)(x) = y^*(Ax).$$

This is called the ***conjugate*** *operator of A*[3].

Definition 9. *Let X be a Banach space and F be a convex, weakly lower semicontinuous functional on X that is either bounded below or proper. For a given $\tau > 0$, define the* ***proximity operator*** *of F with step size τ as*

$$\operatorname{prox}_{\tau F}(x) = \operatorname*{argmin}_{y\in X} F(y) + \frac{\|x-y\|^2}{2\tau}.$$

This operator is well defined by theorem 1[4].

A remark to be made here is that, for separable Hilbert spaces (Banach spaces in which the norm is induced by an inner product), the dual space X^* is, in a very strong sense, completely equivalent to the space itself, X. Namely, the mapping $x \mapsto (y \mapsto \langle x, y\rangle)$ (where $\langle \cdot, \cdot \rangle$ is the inner product in X) defines an isometric isomorphism between X and X^*[5]. For this reason, dual spaces will be tacitly assumed to the base spaces in the following theorem.

Theorem 2 (Chambolle-Pock algorithm)**.** *Let X and Y be separable Hilbert spaces. Let $F\colon X \to \overline{\mathbb{R}}$, $G\colon Y \to \overline{\mathbb{R}}$ be proper, convex and lower semicontinuous functionals, and let $A\colon X \to Y$ be linear and continuous. If the problem*

$$\min_{x\in X} F(x) + G(Ax)$$

[3] Although the notation and name used are the same, these conjugates are essentially different concepts. In this text there will be no ambiguity with these notations because the codomain of a linear operator whose conjugate is needed will never be the real or extended real numbers. Therefore, there is no ambiguity in always considering the conjugate of a given operator as the linear conjugate when the operator is linear and the convex conjugate otherwise.

[4] It is possible to define this operator in greater generality, but doing so involves theory about subdifferentials and monotone operators that is out of the scope of this work.

[5] The surjectivity of this mapping is sometimes called the *Riesz-Fréchet* representation theorem.

has a solution, then the sequence $\{x_n\}_n \subseteq X$ defined by the following iterates converges to it:

$$\begin{cases} x_{n+1} = prox_{\tau F}(x_n - \tau A^* z_n) \\ \tilde{x}_{n+1} = 2x_{n+1} - x_n \\ z_{n+1} = prox_{\sigma G^*}(z_n + \sigma A\tilde{x}_{n+1}) \end{cases} \tag{2}$$

B. Physical model and formalization of the problem

1) Underwater Image Formation Model: When light travels in a water medium, two relevant physical phenomena different from those found in an aerial medium appear: first, the intensity in this light is absorbed by the water molecules in a significant manner. Such absorption is also dependent on the wavelength of the light, and is perceived by the human vision system (HVS) as the red color being underrepresented. The second phenomenon is *scattering*, whereby light does not travel in a straight line from the object to the camera forming an infinitely thin beam, but it is reflected on water molecules instead, forming a cone. This is perceived by the HVS as the image presenting a glow-like or hazy appearance.

Classical works in underwater imaging by McGlamery [26] and Jaffe [17] effectively quantify these phenomena and model the formation of an underwater image I as a linear superposition of three components, in what is often referred to as the Underwater Image Formation Model (UIFM) [30]:

$$I = I_d + I_{fs} + I_{bs}.$$

The I_d (*direct*) term corresponds to the light that travels in a straight line from the object to the camera; the I_{fs} (*forward scattering*) component corresponds to light that travels from the object to the camera but is scattered in a small angle; and the I_{bs} (*backward scattering*) term corresponds to light that travels directly from the light source to the camera by being scattered in a large angle.

Importantly, the forward-scattering component can be obtained from the direct component via a convolution with a *point-spread function* g.

These classical models are based purely on physical properties and depend heavily on the conditions of the water and camera, presenting several constants that need to be estimated experimentally. This poses a disadvantage when working with underwater images taken in unknown conditions, as is often the case in general-purpose underwater image processing algorithms.

For this reason, some modifications and simplifications of the original models have been proposed over the years. Notably, similar notation to the following is used commonly in modern literature [3, 30]: let I be the captured image (i.e., the input), J the scene radiance (i.e., the desired image), t the transmission map, g the point-spread function and B the background light. The transmission map decays exponentially with the distance between the captured object and the camera, in a rate that is dependent on the light wavelength.

For an RGB image, the UIFM can then be expressed as follows, for each color channel $c \in \{r, g, b\}$ and pixel x,

$$I^c(x) = J^c t^c(x) + (g^c * (J^c t^c))(x) + B^c(1 - t^c(x)). \tag{3}$$

Each addend in the above formula corresponds to one component of the original UIFM, in order of appearance both in the equation and the text.

In this work, a further modification of the model is made. By noting that summing the term $J^c t^c$ is equivalent to summing $\delta * (J^c t^c)$ (where δ denotes a Dirac delta), and using the linearity of the convolution, it is possible to combine the first two addends into one without losing generality. Moreover, a noise component is summed, which is common in the literature [30]. The resulting equation is as follows:

$$I^c(x) = B^c(1 - t^c(x)) + (g^c * (J^c t^c))(x) + \xi^c(x), \tag{4}$$

where $\xi(x)$ is random noise on the pixel x, which will be modelled as white noise (i.i.d. centered gaussians). The function g is no longer a point-spread function but a more abstract convolution kernel instead.

From an image processing standpoint, the desired image J is ideally obtained by inverting one of the formulae above from the captured image I. However, due to the ill-posedness and analytical untractability of the deconvolution problem involved, it is convenient to simplify the formula by assuming g^c to be a Dirac delta for each channel c. The resulting equation is

$$I^c(x) = B^c(1 - t^c(x)) + (J^c t^c)(x) + \xi^c(x), \tag{5}$$

This is equivalent to ignoring the forward-scattering component and a common practice in the literature [3]. It is often called "simplified UIFM", and can still produce good results, but is nevertheless a biased model which completely neglects forward scattering.

C. Formalization of the problem

Our approach consists of firstly estimating B, t and a preliminary version of J with traditional methods by assuming the simplified equation eq. (5) holds. Then, we utilize this preliminary version as initialization of an unfolded variational approach to invert the full eq. (4).

More concretely, since t is fixed, we perform a substitution $Jt \to J$ (which is to be inverted later) and pose the following minimization problem, separately for each channel $c \in \{r, g, b\}$:

$$\operatorname*{argmin}_{(J,g)\in D} E(J, g) = \frac{1}{2}\|J * g - R^c\|_2^2 + \mathcal{R}_1(J) + \mathcal{R}_2(g), \tag{6}$$

where $D = L^2(\Omega) \times L^2(\Omega)$, $R = I - B(1 - t)$ and $\mathcal{R}_1$ and $\mathcal{R}_2$ are some continuous regularization functions from $L^2(\Omega)$ to $\mathbb{R}$.

This problem is still mathematically intractable, since the resulting functional is not convex due to the convolution being bilinear w.r.t. the variable functions. A further simplification is made, whereby the functional is alternatively minimized by each of the two variables separately instead of jointly. To compensate for this simplification, the process is repeated a few times:

$$\begin{cases} g^c_{m+1} = \operatorname*{argmin}_{g\in L^2(\Omega)} E(J^c_m, g) \\ J^c_{m+1} = \operatorname*{argmin}_{J\in L^2(\Omega)} E(J, g^c_{m+1}) \end{cases} \tag{7}$$

Henceforth, the iterations resulting from this simplification will be indexed with the letter m and referred to as **greedy iterations**.

Due to the symmetry of the target functional E, both subproblems can be studied under the same scheme. To see this, consider the following minimization problem: let $s, y \in L^2(\Omega)$ and $R\colon L^2(\Omega) \to \overline{\mathbb{R}}$. Fixed those symbols, the idea is to solve for

$$\operatorname*{argmin}_{x \in L^2(\Omega)} F(x) = \frac{1}{2}\|x * y - s\|_2^2 + R(x).$$

Both problems in eq. (7) can be trivially reduced to one of this form by appropriately setting s to I_2, and y and R to either J_m and $\mathcal{R}_2$ or g_{m+1} and $\mathcal{R}_1$. Thus, studying this problem will yield solutions for both subproblems above.

First, it is necessary to show that it is well-posed, meaning that the operations taking place are well-defined and a solution exists. Define the auxiliary linear operator $K\colon L^2(\Omega) \to L^2(\Omega)$ given by

$$Kx = x * y..$$

To see that the codomain of this function is, in fact, L^2, we employ the Young inequality together with the well-known fact that $L^2(\Omega) \subseteq L^1(\Omega)$ for a finitely-measured Ω:

$$\|x * y\|_2 \leq \|x\|_2 \|y\|_1 < +\infty.$$

This also implies the boundedness of K.

Also define $N\colon L^2(\Omega) \to \mathbb{R}$ by $N(a) = \|a - s\|_2^2$, which is clearly strictly convex. In this way, we can express

$$F(x) = N(Kx) + R(x),$$

which is a problem of the same shape of that of theorem 2. Note that N is coercive, convex, lower semicontiuous and, in this case, always finite-valued. Convexity and lower semicontinuity imply weak lower semicontinuity.

Therefore, for a proper and weakly lower semicontinuous regularization R that is either coercive or bounded below, $F = N \circ K + R$ is coercive, proper and weakly lower semicontinuous by lemmas 1 and 2, and the problem admits a solution by theorem 1. For a non-null value of y, the function K is injective (cf. proposition 2 in the appendix), and therefore the strict convexity of N implies that of $N \circ K$, and if R is convex, then F is strictly convex. Therefore, the solution is unique. Moreover, the hypotheses of the Chambolle-Pock algorithm are satisfied. It yields the following iterates:

$$\begin{cases} x_{n+1} = \operatorname{prox}_{\tau R}(x_n - \tau K^* z_n) \\ \tilde{x}_{n+1} = 2x_{n+1} - x_n \\ z_{n+1} = \operatorname{prox}_{\sigma N^*}(z_n + \sigma K \tilde{x}_{n+1}) \end{cases} \tag{8}$$

Henceforth, the iterations in a primal-dual scheme will be indexed with the letter n and referred to as **stages**.

Closed-form expressions for K^* and $\operatorname{prox}_{\sigma N^*}$ can be derived analytically (cf. propositions 3 and 4 in the appendix). They are given by

$$\operatorname{prox}_{\sigma N^*}(x) = \frac{x - \sigma s}{\sigma + 1} \quad \text{and} \quad K^* z = z * \overline{y}, \tag{9}$$

where $\overline{y}(x) = y(-x)$.

The only “missing” symbol is $\operatorname{prox}_{\tau R}$, but this depends entirely on the regularization function used and has no general closed form.

We now discuss precisely which regularization is to be used for each subproblem. Going forward, the variable g will be called *kernel* and the variable J will be called *image*[6].

1) Kernel regularization: For the kernel g, it is desirable to impose three constraints:

1) That it is a convolution kernel, i.e., nonnegative and $\int_\Omega g \; d\mu = 1$. The assumption that the intensity of g be 1 is motivated by the empirical fact that the simplified UIFM, which is equivalent to setting $g = \delta$, often yields acceptable results. Convolving by a kernel with a different total intensity (in a practical sense, $\int_\Omega \delta \; d\mu = 1$) would alter the intensity of the output image J. This was the case in early experiments where this condition was not imposed.
2) That it vanishes outside a square much smaller than Ω. This condition is ubiquitous for convolution kernels in computer vision, as it corresponds to their discrete representations having much smaller sizes than the images'.
3) That it is a decreasing function of its radius. That is, that there exist a decreasing function $\varphi\colon [0, +\infty) \to \mathbb{R}$ such that $g(x) = \varphi(\|x\|)$ for all $x \in \Omega$. This assumption is motivated by two intuitions: first, that there are no privileged directions in an underwater setting, i.e. g is rotationally invariant, or, equivalently, a function of its radius; second, that the larger the angle of the scattering, the larger the distance that light must travel from the object to the camera, and therefore the more intensity is absorbed from it.

These constraints can all be expressed via a regularization function as follows: let

$$\mathcal{K} = \{f \in L^2(\Omega) | f \geq 0, \|f\|_1 = 1\}.$$

Then, condition 1 is equivalent to $g \in \mathcal{K}$. Additionally, for a fixed $r \in \mathbb{Z}^+$ such that $[-r, r]^2 \subseteq \Omega$, define

$$\mathcal{S}_r = \{f \in L^2(\Omega) | f \equiv 0 \text{ in } \Omega \setminus [-r, r]^2\}.$$

Condition 2 is equivalent to $g \in \mathcal{S}_r$ for some sufficiently small r. Finally, define

$$\mathcal{D} = \{\varphi \circ \|\cdot\|_2 \mid \varphi\colon [0, +\infty) \to \mathbb{R} \text{ is decreasing}\},$$

so that condition 3 is equivalent to $g \in \mathcal{D}$.

It is trivial to check that the sets $\mathcal{K}$, $\mathcal{S}_r$ and $\mathcal{D}$ are convex. It is also possible to see that they are closed in $L^2(\Omega)$ (cf. proposition 1), and hence their intersection is closed and convex. Thus, the indicator function given by

$$\delta_{\mathcal{K} \cap \mathcal{S}_r \cap \mathcal{D}}(g) = \begin{cases} +\infty, & \text{if } g \in \mathcal{K} \cap \mathcal{S}_r \cap \mathcal{D}, \\ 0, & \text{otherwise}, \end{cases}$$

[6] Here, J does not correspond to the radiance (i.e. “recovered *image*”), but J/t does. This is just convenient notation.

is convex and weakly lower semicontinuous by Lemma 2.5.ii) of [7]. It is bounded below, and thus, taking $R = \delta_{\mathcal{K}\cap\mathcal{S}_r\cap\mathcal{D}}$ would guarantee existence of solution to the problem.

However, computing the proximity operator resulting would be challenging, so a final simplification is made. Specifically, g is taken as a centered gaussian density truncated to $[-r,r]^2$ (that is, set to 0 outside of $[-r,r]^2$) and normalized so that its integral is 1. This is a sufficient condition for those outlined above, and allows for an important simplification of the optimization scheme: instead of performing primal-dual iterations, a simple gradient descent will be used to optimize the standard deviation of the gaussian density.

Such simplification may not reach the optimum guaranteed to exist with the conditions above, but will be empirically sufficient, likely faster to converge and much simpler from an implementation standpoint.

2) Image regularization: Instead of fixing a regularization for the image, the iterative scheme eq. (8) will be unfolded.

Concretely, the proximity operator $\text{prox}_{\tau R}$ will be replaced by neural networks that will be then trained end-to-end. Mathematically, the resulting scheme is

$$\begin{cases} x_{n+1}^m = \mathcal{N}_n^m(x_n^m - \tau K^* z_n^m) \\ \tilde{x}_{n+1}^m = 2x_{n+1}^m - x_n^m \\ z_{n+1}^m = \text{prox}_{\sigma N^*}(z_n^m + \sigma K \tilde{x}_{n+1}^m) \end{cases}, \qquad (10)$$

where $\mathcal{N}_n^m$ is a neural network specific for stage n in the greedy iteration m. Although this deviates more from the original primal-dual scheme than using a fixed neural network would, it has been shown to yield significantly better results [1].

Instead of iterating until convergence, the number of stages and greedy iterations will be fixed beforehand.

III. Algorithmic approach

From a high-level perspective, the approach followed to obtain an estimate of the radiance J from the image I is as follows:

1) Use a score-based quad-tree algorithm to estimate the background light B.
2) Assume the simplified model, eq. (5), holds and use a DPC prior to obtain a coarse estimate of the red channel of the transmission map t^r. Refine this estimate by passing it through a guided filter with I as a guide.
3) Obtain the green and blue channels of the transmission map by exponentiating the red one. Then, use the obtained estimate of t to compute an initial estimate for J, J_0.
4) Pose a minimization problem with the obtained estimates for B and t and the full model, eq. (4) with a formal substitution $Jt \to J$. Solve this problem with an alternating iterative approach using $J_0 t$ as initialization, and obtain a final estimate for J from the minimum J^* by inverting the substitution: $J = J^* / \max(t, 0.1)$. The kernel is optimized with simple gradient updates, and the radiance is optimized via eq. (10).

In the following, steps 1 to 3 will be referred to as *deterministic steps* and step 4 will be called *variational step*.

A. Determinisitc steps

For step 1, the background light is estimated by iteratively splitting the image into four quadrants. Out of those regions, the one with the highest score, given by the average minus the standard deviation of the pixel values in the region, is selected and split again. The process stops once the selected region is smaller than a given threshold (16 colored pixels in this implementation). Then, the background light of the image is chosen as the pixel closest to white (in euclidean norm) in that small patch. This process is summarized in Algorithm 1.

For step 2, a Red Channel Prior is utilized [11]. This is a modification of the classical Dark Channel Prior [15]. The approach is based on the assumption that, for every pixel $x \in \Omega$, the following equality holds for a square patch of fixed radius centered at x, Ω_x:

$$\min_{y\in\Omega_x} \min(1 - I^r(y), I^g(y), I^b(y), S(y)) = 0,$$

where the *saturation map* S is defined as 0 for black pixels and $S = 1 - \frac{\min(I^r,I^g,I^b)}{\max(I^r,I^g,I^b)}$ otherwise. Additionally, t_0 is imposed to be constant across channels and in Ω_x neighborhoods. From this and eq. (5), authors in [11] derive the following equation for the transmission map of the red channel, t_0^r, for a fixed multiplier $\lambda \in [0,1]$, which in this work is set to 0.75:

$$t_0^r(x) = 1 - \min_{y\in\Omega_x} \left(\min \left(\frac{1 - I^r(y)}{1 - B^r}, \frac{I^g(y)}{B^g}, \frac{I^b(y)}{B^b}, \lambda S(y) \right) \right).$$

Nota: he estat incapaç de reproduir aquest raonament amb la saturació. És a dir, de s'RCP "estàndard" (llevant es $S(y)$ des mínim) sí que se dedueix la fórmula (també sense $S(y)$) amb les hipòtesis que explic, però no he estat capaç d'introduir-li la saturació. A l'article no donen massa pistes llevat que és una adaptació "straight-forward" de s'altre raonament. Si vos sembla bé, jo ho deixaria així com està escrit.

For step 3, a fine version of the transmission map, t^r, is obtained by passing t_0^r through a guided filter [16] with I as the guide. The transmission map estimates for the other two channels are obtained by exponentiating the red transmission map by appropriate coefficients. This is sound since, physically, t decays exponentially with the distance from the object to the camera, and the base of the exponent depends on the color channel. Thus, the transmission maps for the different color channels are related via exponentiation. In this work, the exact coefficients presented in [21] are utilized:

$$t^c = (t^r)^{\frac{\alpha_c}{\alpha_r}},$$

where $c \in \{r,g,b\}$ and α_c depends on the color's wavelength λ_c in nanometers (respectively: 620, 540 and 450 for red, green and blue) and the background light B^c:

$$\alpha_c = \frac{-0.00113\lambda_c + 1.62517}{B^c}.$$

Finally, we estimate a first version of $J_0 t$ by inverting eq. (5). For each color channel $c \in \{r,g,b\}$:

$$(J_0^c t^c)(x) = I^c(x) - (1 - t^c(x))B^c \qquad (11)$$

Of course, it would be possible to compute J_0 directly from the previous equation, but since it is numerically unstable (t can be and often is close to 0) and a formal substitution $Jt \to J$ will be performed for the subsequent variational problem anyway, this is unnecessary.

Steps 2 and 3 are summarized in Algorithm 2.

Algorithm 1: Estimate background light.

```
Data: $I$
Result: $J$
img ← $I$;
while $Size(img) > min\_size$ do
    quadrants ← Split(img);
    max_score ← 0;
    for $q$ in quadrants do
        $\mu$ ← Mean($q$);
        $\sigma$ ← Std($q$);
        score ← $\mu - \sigma$;
        if $score > max\_score$ then
            max_score ← score;
            img ← $q$ ;
        end
    end
end
```

Algorithm 2: Estimate TM and initial radiance.

```
Data: Input image $I$, background light $B$, TM patch radius $r_t$, guided filter patch radius $r_g$
Result: TM estimation $t$, multiplied radiance estimation $J_0t$
; /* Compute coarse TM */
$S$ ← Saturation($I$);
Initialize $t^r$ with the width and height of $I$;
for pixel $x$ do
    ; /* $\Omega_x$ patch around $x$, radius $r_t$ */
    r_min ← $\min_{y \in \Omega_x}(1 - I^r[y])/(1 - B^r)$;
    g_min ← $\min_{y \in \Omega_x}(I^g[y])/B^g$;
    b_min ← $\min_{y \in \Omega_x}(I^b[y])/B^b$;
    s_min ← $\min_{y \in \Omega_x}(S[y]) \cdot \lambda$;
    $t^r[x]$ ← 1 − min(r_min, g_min, b_min, s_min)
end
; /* Refine transmission map */
$t^r$ ← $GuidedFilter(I, t^r; r_g)$;
Obtain $t$ by exponentiating $t^r$;
; /* Estimate $J_0t$ */
for channel $c$ do
    $(J_0^c t^c) \leftarrow I^c - (1 - t^c)B^c$
end
return $t$, $J_0t$
```

B. Variational step

For step 4, the minimization subproblems in eq. (7) will be solved for a fixed, small number of greedy iterations (concretely, 3). The iterative scheme in eq. (8) with the formulae in eq. (9) is used, with the aforementioned symbolic substitutions. The missing operator $\text{prox}_{\tau R}$ is unfolded, meaning that it is substituted by a neural network instead of explicitly derived. The architecture and training of said neural network will be discussed later, in the implementation section.

The final approach used for obtaining an approximate solution to eq. (6) is summarized in Algorithm 3. Instead of analytically computing the derivative of $\|I - J * g\|_2^2$ w.r.t. the standard deviation of g, *pytorch*'s *autograd* engine is leveraged. An ADAM optimizer is used with a step size of σ.

Algorithm 3: Solve variational problem.

```
Data: $I, B, t, J_0t, \sigma, \tau$
Result: $J$
$J \leftarrow J_0t$;
$R \leftarrow I - (1 - t)B$;
Initialize $g$ as a centered gaussian kernel with std. of 1;
Initialize dual variables $\tilde{g}$, $\tilde{J}$ to 0;
for $n = 1$ to $G$ do
    ; /* Fix $J$, estimate $g$. */
    For a fixed, small number of iterations, minimize $\|R - J * g\|_2^2$ via gradient updates on the std.
    ; /* Fix $g$, estimate $J$. */
    $\overline{g} \leftarrow DoubleFlip(g)$;
    for $s = 1$ to $S$ do
        $tmp \leftarrow NeuralNet^J_{ns}\left(J - \tau \cdot \left(\tilde{J} * \overline{g}\right)\right)$;
        $J \leftarrow 2 \cdot tmp - J$;
        $\tilde{J} \leftarrow (\tilde{J} + \sigma \cdot (J * g) - \sigma \cdot R)/(\sigma + 1)$;
    end
end
return J / t
```

C. Implementation

Xerrar de U2Fold, la implementació en pytorch dels algoritmes d'abans i de la UNET additiva, etc.

1) u2fold: The algorithms detailed above were implemented in the Python programming language, version 3.12. The program is called u2fold (from Underwater UnFOLDing). The source code for the program is freely available at https://github.com/Frankwii/u2fold. The major dependency used was *pytorch*, which provides a nice programming interface for manipulating multidimensional arrays via the *Tensor* abstraction, automatically performing simple optimization algorithms via its *autograd* engine, and, what it is most popular for, implementing neural networks with the *torch.nn.Module* class and a variety of related utilities. A secondary dependency is *pydantic*, for validating input parameters specified in a JSON file such as network hyperparameters or the dataset path.

The program is controlled via a command-line interface (CLI) and is intended to be run on a Linux machine (it has not been tested on other operating systems). The main subcommand of the program is "run", which takes a required argument "–spec" that should indicate the path to a JSON file. This file will be referred to as the *configuration file*, since the behavior of the program is determined by its contents. Example files for this can be found at the root of the repository: `spec_train.json` for training a neural networ, and `spec_exec.json` for executing that previously trained network. Additionally, a subcommand "docs" is provided. This opens a simple website in the system's default browser (as specified in the "$BROWSER" environment variable) with a visual documentation of the fields that can or should be included in the configuration file. The page is very similar to those that are often generated via tools like Swagger or fastAPI for RESTful APIs. A screenshot of the generated configuration is included in section VII-B.

D. Algorithms

Algorithms 1, 2 and 3 were implemented using *pytorch* primitives, including the guided filter. Although not explicited in the figures, the implementations work for batched images.

Algorithm 1 is implemented in the source file *src/u2fold/math/background_light_estimation.py*. The code is structured slightly differently than in 1 since the image batching required the usage of the *torch.gather* method.

Algorithm 2 is implemented in the *src/u2fold/math/transmission_map_estimation.py* module. The implementation efficiently computes patch minima in parallel by leveraging *pytorch*'s *torch.max_pool2d* builtin.

Algorithm 3 is implemented in the method *forward_pass* of the *Orchestrator* class in the *src/u2fold/orchestrate/generic.py* module. The reason for using an object-oriented approach instead of a sequential one was that the logic required for secondary tasks needed in the program (logging results to visualize training, or loading model weights from disk into an array of the dimensions required by the number of greedy iterations and stages) was relatively coupled to the algorithm execution and more comfortable to implement in a stateful class than via dependency injection. A relevant implementation detail is that the gradient updates done to the kernel g were done with an ADAM optimizer which took 20 steps for the first greedy iteration and 10 for the subsequent ones. This empirically seemed to be a good criterion; performing more updates, especially during the first greedy iteration, would result in g converging to a Dirac delta (i.e. the standard deviation converged to 0) since J_0 is first initialized following eq. (5), which is equivalent to setting g to be a Dirac delta.

E. Neural networks

Two neural network architectures were trained to optimize the output of algorithm 3, both of which are relatively simple modifications of the original UNet.

IV. Experimentation

A. Metric calibration

Given the high computational costs of training neural networks, it is unfeasible to perform a grid search on a substantial set of hyperparameter choices and train a network from scratch for each choice.

As an attempt to overcome this limitation, the following approach is proposed. For a given neural network architecture:

1) Determine, through a manual process, a sensible hyperparameter combination as a starting point. This includes the choice of **training-related** hyperparameters, such as the optimizer, learning rate scheduler, etc., and **architectural** hyperparameters, such as the number of layers, their sizes, activation functions...
2) Fix the training-related hyperparameters of the starting point, and for a given set of choices for the architectural hyperparemters (which should include the starting point), train a model on each choice for a small number of epochs. Select the "best" one by minimizing a target metric.
3) Fix the architectural parameters of the best neural network so far, and, similarly, select the best choice of training-related hyperparameters by training a model on each choice. This time, however, the number of epochs should be larger, since some learning rate schedulers show performances greatly dependent on the number of epochs.
4) Finally, train the best model so far during many epochs as an attempt to get the best performance.

Nota: això ha anat bastant malament. Crec que hauré de tornar a sa manera manual de simplement anar provant combinacions i au. Una llàstima. Of course, the target metric choice plays a central role in determining the output. Several metrics will be used to evaluate the quality of the model (cf. [TODO: AÑADIR SECCIÓN DE MÉTRICAS]), but it would be desirable to obtain a single number that will allow the automatic comparison of models. An obvious choice is a simple addition of the metrics, but this presents the problem of different metrics having different ranges of values. A simple solution to this is dividing each value by its average computed over the dataset.

Table I
CALIBRATION RESULTS FOR DIFFERENT METRICS AND LOSS TERMS WITH THE UIEB DATASET.

Metric or loss	Average	Standard deviation
UCIQEm	10.59041	2.16975
TV	0.05879	0.03238
PSNRm	0.05977	0.02899
DSSIM	0.11068	0.06204
MSE	0.02561	0.02265
CCSm	0.03598	0.03860
Fidelity loss	0.03710	0.02991

V. Conclusions

VI. Student's comments

This section is not strictly part of the work but instead it is intended to offer some insight on its writing to reviewers of the thesis or potential future students that use it as a reference for similar works.

A. Limitations and ideas for future work

Ideas:

1) El algoritmo actual está limitado por el hecho de que la manera de estimar B, t, etc. es relativamente rudimentaria y, más importante, fija. Si está mal estimada la t al principio va mal todo lo demás.
2) Probar de utilizar una red neuronal para seleccionar hiperparámetros adecuados a cada imagen: exponentes para transmission map, coeficientes de saturación o regularización, etc.
3) Probar con otros tipos de redes neuronales (ViT, ResNet...)
4) Probar con maneras más sofisticadas de estimar la g (p.ej. no suponer que es gaussiano, sino de una familia más general).
5) Probar a hacer unfolding no solo con uno de los proximales sino con los dos.
6) Probar de compartir los pesos de la red (aunque esto probablemente vaya peor)

B. *Use of LLMs in this work*

Large Language Models (LLMs) were used mostly as a consult material helpful to build early intuition. Concretely, the Gemini 2.5 Pro model available for free at https://aistudio.google.com (at the moment of writing this) was used. Personally, I find the base model useful for scientific tasks and the lack of memory across chats by default to be an advantage instead of otherwise, since it forced me to write specific prompts for the concrete task I was interested in early on, instead of having a long, ongoing conversation. That often led me to realizing the solution to the problem myself while articulating the question.

Almost every line of code under the "src/u2fold" directory of the provided repository was written by me and not automatically generated. I generally strongly dislike AI-generated code since it tends to contain very verbose comments, many unnecessary checks or early returns and generic names that do not reflect the intention behind the code. I subscribe to the philosophy that inline comments (not docstrings) tend to get outdated extremely quickly on relatively large codebases such as this thesis', and that the programmer should name their abstractions and structure their code in such a way that, generally speaking, it is understandable to someone external but fluent in the language just by reading it, with no need for intercalated explanations.

That being said, I did use AI tools for the following coding-related scenarios:

- Write parts of code snippets under the "scripts" directory since those were intended to be ran few times and not be modified afterwards, and were small enough for manual inspection and polishing. The open-source tool Gemini CLI proved especially useful for this.
- "Code reviews" of certain modules. Although very good for catching obvious mistakes early, this proved to be of limited usefulness for larger or more important modules since current LLMs often show sycophantic behavior and will agree on retrospectively dubious architectural choices. This led to a false sense of security that was sometimes counterproductive.

For the mathematical part of the thesis, I found the utilization of LLMs incredibly useful not because of the reasonings they present, which are often irrecoverably flawed in very subtle and hard to spot ways (hence, a waste of time to pay attention to), but because they do identify useful sources of information or relatively unknown but certainly related theorems. I would often ask for a proof of some conjecture I had come up with and then completely ignore the answer except for the references it provided. I would then go through the references and find something that was indeed useful. An example of this was the Titchmarsh-Lions theorem, which I was unaware of but was crucial to prove the uniqueness of solution for the greedy subproblems.

Finally, they were used for styling and spelling review.

C. *Difficulties found while writing the thesis*

Comentar lo poco concreto que era el artículo de referencia y lo complicado que fue de seguir. Concretamente, la manera en la que estiman el transmission map es 0 explícita y especialmente no se entiende qué hacen para estimar la g.

One particular limitation to be difficulty I encountered was the pitiful and ascientific, yet ubiquitous practice of not publishing source code along with research papers, but only a compiled binary instead. This is done possibly in the hopes of being cited when compared to other approaches, but discouraging researches from reproducing or improving their methods. This hinders one of the pillars of science, reproducibility, and should not be allowed in a journal that considers itself serious and scientific. This was the case for [21] and, most importantly, the main reference [30], which made fine-tuning algorithmic parameters or filling in details of the (often poor) exposition much harder than needed.

También tal vez la mala idea que fue la aplicación de CLI (funciona mucho mejor configurarlo con un JSON)

D. *Original contributions*

Paper [30] served as an initial reference for this work. However, several modifications were made, many of which were fully original.

The mathematical model used is, of course, a classical one in the field and the same as in [30]. Its formalization and treatment, however, are original. Concretely, to the best of the author's knowledge, the "unification" of forward-scattering and direct components into a single one has not been done previously in the literature. All of the mathematical treatment and justifications (existence and uniqueness of solutions to the problems...) of the derived variational approach are also novel.

The additive UNet architecture is new too.

The implementations of all algorithms and neural networks described in this paper were made from scratch on *pytorch*. In addition to those, so were the guided filter's and all defined metrics', some of which required colorspace conversions. Free and open source library kornia's implementation of colorspace conversions was consulted, specifically for the coefficients of the conversion matrices.

VII. Appendix

A. Auxliary proofs

Proposition 1. *The sets $\mathcal{K}$, $\mathcal{S}_r$ and $\mathcal{D}$ are all closed.*

Proof. Let S be one of the sets above, and let $\{f_n\}_n \subseteq S$ denote a sequence converging in $L^2(\Omega)$ to a given limit $f \in L^2(\Omega)$. Showing that the set S is closed is equivalent to showing that $f \in S$.

A well-known result about L^p spaces states that L^p convergence implies pointwise convergence up to a subsequence [4]. This result will be used in all three cases. For notational simplicity, simply take f_n as the converging subsequence.

First, consider $S = \mathcal{K}$. The result above immediately implies that f is nonnegative by taking limits on the inequality $f_n(x) \geq 0$. Another well-known result about $L^p(\Omega)$ spaces states that, whenever $\mu(\Omega) < +\infty$, the following inequality holds for $1 \leq p < q < +\infty$ and any measurable function g:

$$\|g\|_p \leq \mu(\Omega)^{1/p-1/q}\|g\|_q.$$

Thus, for $p = 1$, $q = 2$,

$$\|f_n - f\|_1 \leq \mu(\Omega)^{1/2}\|f_n - f\|_2.$$

Taking limits, this implies that f_n converges to f in L^1. As a consequence, $\|f\|_1 = \lim_n \|f_n\|_1 = \lim_n 1 = 1$.

Second, for $S = \mathcal{S}_r$, the result is immediate by taking limits on the converging subsequence outside of $[-r, r]^2$.

Finally, for $S = \mathcal{D}$, consider, for each f_n, the function ϕ_n such that $f_n = \phi_n \circ \|\cdot\|_2$. Take some $x \in L^2(\Omega)$ such that $\|x\|_2 = 1$. Then, for each $t \in [0, +\infty)$,

$$\phi_n(t) = f_n(tx).$$

This defines a function ϕ by $\phi(t) = \lim_n \phi_n(t) = \lim_n f_n(tx) = f(tx)$. Since x was arbitrary, it also shows that $f = \phi \circ \|\cdot\|_2$ as desired. □

Proposition 2 (Injectivity of convolution)**.** *Suppose that g is the continuous representation of a non-null digital image. Then, the function $K\colon L^2(\Omega) \to L^2(\Omega)$ given by $Kx = x * g$ is injective.*

Proof. Since the mapping has been shown to be well-defined and linear, it suffices to show that $ker(K) = \{0\}$; that is, that $Kx = 0$ a.e. implies $x = 0$ a.e.

This proof will make use of the Titchmarsh-Lions convolution theorem, which states that for any two compactly supported distributions T_1, T_2, the convex hull of $supp(T_1 {*} T_2)$ is the sum of the convex hulls of $supp(T_1)$ and $supp(T_2)$ (chapter 45 of [8]). Properly defining distributions, convolutions between them or their supports is out of the scope of this work[7], but it suffices to know that any $L^p(\Omega)$ function for $p \geq 1$ and a bounded measurable set Ω defines a distribution, and that the definitions given for distributions correspond exacty to those given for $L^p(\Omega)$ functions, the latter being much simpler.

The support of a function $f \in L^p(\Omega)$ in this context is defined as the smallest closed subset of $\mathbb{R}^2$ such that f is 0 almost everywhere outside of it. Formulaically:

$$supp(f) = \bigcap_{A \text{ closed}, f=0 \text{ a.e. in } A^c} A,$$

where f is extended to be 0 outside of Ω.

It is immediate to see that, for a bounded and measurable set $\Omega \subseteq \mathbb{R}^2$, an $L^p(\Omega)$ function has a compact support, and that said is empty if, and only if, the function is null almost everywhere. It is also immediate to see that a subset of $\mathbb{R}^n$ is empty if, and only if, its convex hull is empty.

Thus, if x is such that $x * g = 0$ a.e., its support is empty, and so is its convex hull. This implies that either the convex hull of $supp(g)$ or the convex hull of $supp(x)$ are empty. By hypothesis, g is not null, and hence it must be that x is zero almost everywhere. □

In the following two propositions, the usual identification between a Hilbert space and its dual via $x \mapsto (y \mapsto \langle x, y\rangle)$ is implicitly assumed. This is done, in particular, for $L^2(\Omega)$.

Proposition 3. *Let $N\colon L^2(\Omega) \to \mathbb{R}$ be defined by*

$$N(x) = \|x - s\|_2^2$$

for some $s \in L^2(\Omega)$. Then,

$$prox_{\sigma N^*}(x) = \frac{x - \sigma s}{\sigma + 1}.$$

Proof. By the Moreau decomposition theorem,

$$\text{prox}_{\sigma N^*}(x) = x - \sigma \text{prox}_{N/\sigma}\left(\frac{x}{\sigma}\right). \tag{12}$$

Thus, it is sufficient to compute $\text{prox}_{\lambda N}$ for an arbitrary $\lambda > 0$ and then replace $\lambda = \frac{1}{\sigma}$.

For a fixed $z \in L^2(\Omega)$, consider the functional $J_{z,\lambda}(y) = \frac{\|y-z\|_2^2}{2\lambda} + \frac{1}{2}\|y - s\|_2^2$, so that

$$\text{prox}_{\lambda N}(z) = \text{argmin}_y J_{z,\lambda}(y).$$

Clearly, $J_{z,\lambda}$ is Gâteaux differentiable and

$$J_{z,\lambda}(y; h) = \langle h, \frac{1}{\lambda}(y - z)\rangle + \langle h, y - s\rangle.$$

Whence $\forall h\colon J_{z,\lambda}(y; h) = 0$ if, and only if, $y = \frac{z+\lambda s}{\lambda+1}$. Thus, by the Fermat principle, and taking $\lambda = \frac{1}{\sigma}$,

$$\text{prox}_{N/\sigma}(z) = \text{argmin}_y J_{z,1/\sigma}(y) = \frac{\sigma z + s}{\sigma + 1}.$$

Combining this last equation with eq. (12), we have that

$$\text{prox}_{\sigma N^*}(x) = x - \frac{\sigma}{\sigma + 1}(x + s) = \frac{x - \sigma s}{\sigma + 1}. \quad \square$$

Proposition 4. *Let $K\colon L^2(\Omega) \to L^2(\Omega)$ be defined by*

$$Kx = x * y$$

for some $y \in L^2(\Omega)$. Then, the dual operator K^* is given by

$$K^* x = x * \overline{y}.$$

Proof. By definition, K^* is the only operator satisfying the equality

$$\langle Ka, b\rangle = \langle a, K^* b\rangle$$

[7] See [8] for a detailed exposition.

for all $a, b \in L^2(\Omega)$, where $\langle \cdot, \cdot \rangle$ is the usual scalar product in $L^2(\Omega)$: $\langle f, g \rangle = \int_\Omega fg \, dm$, where m is the Lebesgue measure restricted to Ω.

Note that, in order to properly define convolution between functions with domain $\Omega \subseteq \mathbb{R}^n$, it is first necessary to extend to $\mathbb{R}^n$ by setting them to 0 in Ω^c.

It is well-known that for any given $\varphi, \phi, \psi \colon \mathbb{R}^n \to \mathbb{R}$ such that their pairwise convolutions are defined and integrable, the following equality holds:

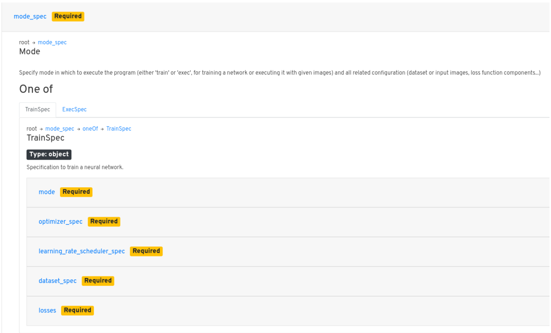
$$\langle \varphi * \phi, \psi \rangle = \langle \phi, \psi * \overline{\varphi} \rangle \tag{13}$$

where $\overline{\varphi}(x) = \varphi(-x)$. Therefore,

$$\langle a * y, b \rangle = \langle a, b * \overline{y} \rangle.$$

Thus, K^* is simply given by $K^* b = b * \overline{y}$. □

B. Screeshots



mode_spec Required

root → mode_spec

Mode

Specify mode in which to execute the program (either 'train' or 'exec', for training a network or executing it with given images) and all related configuration (dataset or input images, loss function components...)

One of

TrainSpec ExecSpec

root → mode_spec → oneOf → TrainSpec

TrainSpec

Type: object

Specification to train a neural network.

mode Required

optimizer_spec Required

learning_rate_scheduler_spec Required

dataset_spec Required

losses Required

Figure 1. Screenshot of a fragment of the generated documentation

References

[1] J. Adler and O. Öktem. Learned primal-dual reconstruction. *IEEE Transactions on Medical Imaging*, 37(6):1322–1332, 2018.

[2] C. Ancuti, C. O. Ancuti, T. Haber, and P. Bekaert. Enhancing underwater images and videos by fusion. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 81–88, 2012.

[3] C. O. Ancuti, C. Ancuti, C. De Vleeschouwer, and P. Bekaert. Color balance and fusion for underwater image enhancement. *IEEE Transactions on image processing*, 27(1):379–393, 2017.

[4] R. Ash. *Real Analysis and Probability*. Bibliographie-P. Academic Press, 1972.

[5] B. Cai, X. Xu, K. Jia, C. Qing, and D. Tao. Dehazenet: An end-to-end system for single image haze removal. *IEEE Transactions on Image Processing*, 25(11):5187–5198, 2016.

[6] J. Y. Chiang and Y.-C. Chen. Underwater image enhancement by wavelength compensation and dehazing. *IEEE Transactions on Image Processing*, 21(4):1756–1769, 2012.

[7] C. Clason and T. Valkonen. Introduction to nonsmooth analysis and optimization, 2024.

[8] W. F. Donoghue. *Distributions and Fourier transforms*, volume 32. Academic Press, 1969.

[9] P. Drews, Jr., E. do Nascimento, F. Moraes, S. Botelho, and M. Campos. Transmission estimation in underwater single images. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, June 2013.

[10] G. Foresti. Visual inspection of sea bottom structures by an autonomous underwater vehicle. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 31(5):691–705, 2001.

[11] A. Galdran, D. Pardo, A. Picón, and A. Alvarez-Gila. Automatic red-channel underwater image restoration. *Journal of Visual Communication and Image Representation*, 26:132–145, 2015.

[12] K. Gregor and Y. LeCun. Learning fast approximations of sparse coding. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 399–406, Madison, WI, USA, 2010. Omnipress.

[13] J. Gui, X. Cong, Y. Cao, W. Ren, J. Zhang, J. Zhang, J. Cao, and D. Tao. A comprehensive survey and taxonomy on single image dehazing based on deep learning. *ACM Comput. Surv.*, 55(13s), July 2023.

[14] T. Guo, X. Li, V. Cherukuri, and V. Monga. Dense scene information estimation network for dehazing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 0–0, 2019.

[15] K. He, J. Sun, and X. Tang. Single image haze removal using dark channel prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(12):2341–2353, 2011.

[16] K. He, J. Sun, and X. Tang. Guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(6):1397–1409, 2013.

[17] J. Jaffe. Computer modeling and the design of optimal underwater imaging systems. *IEEE Journal of Oceanic Engineering*, 15(2):101–111, 1990.

[18] J.-H. Kim, W.-D. Jang, J.-Y. Sim, and C.-S. Kim. Optimized contrast enhancement for real-time image and video dehazing. *Journal of Visual Communication and Image Representation*, 24(3):410–425, 2013.

[19] B. Li, X. Peng, Z. Wang, J. Xu, and D. Feng. Aod-net: All-in-one dehazing network. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[20] C.-Y. Li, J.-C. Guo, R.-M. Cong, Y.-W. Pang, and B. Wang. Underwater image enhancement by dehazing with minimum information loss and histogram distribution prior. *IEEE Transactions on Image Processing*, 25(12):5664–5677, 2016.

[21] C.-Y. Li, J.-C. Guo, R.-M. Cong, Y.-W. Pang, and B. Wang. Underwater image enhancement by dehazing with minimum information loss and histogram distribution prior. *IEEE Transactions on Image Processing*, 25(12):5664–5677, 2016.

[22] P. Li, J. Tian, Y. Tang, G. Wang, and C. Wu. Deep retinex network for single image dehazing. *IEEE Transactions on Image Processing*, 30:1100–1115, 2021.

[23] J.-L. Lisani, A.-B. Petro, C. Sbert, A. Alvarez-Ellacuria, I. A. Catalan, and M. Palmer. Analysis of underwater image processing methods for annotation in deep learning based fish detection. *IEEE Access*, 10:130359–130372, 2022.

[24] R. Liu, X. Fan, M. Hou, Z. Jiang, Z. Luo, and L. Zhang. Learning aggregated transmission propagation networks for haze removal and beyond. *IEEE Transactions on Neural Networks and Learning Systems*, 30(10):2973–2986, 2019.

[25] C. H. Mazel. In situ measurement of reflectance and fluorescence spectra to support hyperspectral remote sensing and marine biology research. In *OCEANS 2006*, pages 1–4, 2006.

[26] B. L. McGlamery. A Computer Model For Underwater Camera Systems. In S. Q. Duntley, editor, *Ocean Optics VI*, volume 0208, pages 221 – 231. International Society for Optics and Photonics, SPIE, 1980.

[27] V. Monga, Y. Li, and Y. C. Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Processing Magazine*, 38(2):18–44, 2021.

[28] E. Trucco and A. Olmos-Antillon. Self-tuning underwater image restoration. *IEEE Journal of Oceanic Engineering*, 31(2):511–519, 2006.

[29] C. Wang, Y. Zou, and Z. Chen. Abc-net: Avoiding blocking effect color shift network for single image dehazing via restraining transmission bias. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 1053–1057, 2020.

[30] J. Xie, G. Hou, G. Wang, and Z. Pan. A variational framework for underwater image dehazing and deblurring. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(6):3514–3526, 2021.