Universitat
de les Illes Balears

MASTER'S THESIS

UNTITLED THESIS

Frank William Hammond Espinosa

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

Centre for Postgraduate Studies

Academic year 2024/25

UNTITLED THESIS

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.

Untitled thesis

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*—This is a very abstract abstractionary abstract.**

I. Introduction

Blabla cites SOTA etc avorrit.

II. Mathematical model

A. Theoretical grounding and notation

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 formulas of 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 scalar 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

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

where the left term follows a standard 0-indexed array notation, and the right is standard mathematical function notation.

[1]The term *continuous* here refers to the domain of the representation. In fact, the functions involved will not be continuous in general.

Then, the functions I_c^b 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 relationship between the two representations respects pointwise operations (and, therefore, 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 employ 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 is possible to show that $\mathcal{L}^p(\Omega)$ is a normed vector space when equipped with $\| \cdot \|_p$ as a norm.

Finally, define $L^p(\Omega)$ 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 $p = 2$.

[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 Riemmann integral counterpart, whenever the latter is defined.

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 us to work with operators defined on function spaces in a more abstract fashion than with multidimensionals arrays. By doing so, the involved operators and formulas we obtain often have much cleaner closed forms.

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 interpeted 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 $\lim_n x_n \in X$,*

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

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 $x \neq y$ and $\lambda \in (0, 1)$, F is said to be **strictly convex**.*

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 $\lim_n J(x_n) = +\infty$.*
*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.*

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.

4) Prima: Explicar:

1) Molt breument espais L^p, modelització d'imatges com a funcions a L^p.
2) Resum molt resumit d'anàlisi convexa: conjugada convexa; diferenciabilitat de Gateaux i principi de Pascal.
3) Resum molt resumit d'operador proximal i algorisme de Chambolle-Pock.

B. Physical model and formalization of the problem

1) Underwater Image Formation Model: PENDENT: Explicar un mínim d'on surt aquella equació (UIFM) i posar cites.
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 funcion and B the background light. According to the UIFM, for each color channel $c \in \{r, g, b\}$ and pixel x,

$$I^c(x) = B^c\left(1 - t(x)\right) + \left(g^c * \left(J^c t\right)\right)(x) + \xi^c(x), \quad (2)$$

where $\xi(x)$ is random noise on the pixel x, which will be modelled as white noise (i.i.d. centered gaussians).
Ideally, the desired image J is obtained by inverting the previous formula from the captured image I. However, due to the ill-posedness and general analytical untractability of the problem, it is convenient to split it into two subproblems: define, for each color channel $c \in \{r, g, b\}$, the auxiliary maps given by

$$I_1^c(x) = J^c(x)t(x) + B^c(1 - t(x)) \quad (3)$$

and

$$I_2^c(x) = \left(\left(J^c t\right) * g^c\right)(x) + \xi^c(x), \quad (4)$$

so that $I = I_1 + I_2$. In fact, in order to avoid inverting the convolution present in eq. (4), traditional techniques simply assume $g \equiv 0$ and get rid of I_2 altogether (CITA) by incorporating the noise into I_1. This, of course, simplifies the problem but results in a biased model which does not capture the full array of physical phenomena involved in the image formation process.
Aquí pentura citar s'article de referència.

C. Formalization of the problem

Our approach consists of firstly estimating B, t and a preliminary version of J with traditional methods. This allows the use of eq. (3) to obtain an estimation of I_1 and take $I_2 = I - I_1$. Finally, we utilize an unfolded variational approach to invert eq. (4). 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 $c \in \{r, g, b\}$:

$$\underset{(J,g)\in D}{\operatorname{argmin}} E(J, g) = \frac{1}{2}\|J * g - I_2^c\|_2^2 + \mathcal{R}_1(J) + \mathcal{R}_2(g), \quad (5)$$

where $D = L^2(\Omega) \times L^2(\Omega)$, and $\mathcal{R}_1$ and $\mathcal{R}_2$ are some regularization functions in $\mathcal{L}(L^2(\Omega), \mathbb{R})$. The reason behind the use of a generic regularization will become clear when discussing the algorithmic approach to the solution via unfolding.

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:

$$\begin{cases} g_{n+1}^c = \underset{g\in L^2(\Omega)}{\operatorname{argmin}} E(J_n^c, g) \\ J_{n+1}^c = \underset{J\in L^2(\Omega)}{\operatorname{argmin}} E(J, g_{n+1}^c) \end{cases} \quad (6)$$

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

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

Both problems in eq. (6) can be trivially reduced to one of this form by appropriately setting s to I_2, and y and R to either J_n and $\mathcal{R}_2$ or g_{n+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 unique 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 (REFERENCIAR PART DE BACKGROUND MATEMATIC) and satisfies the hypotheses of the Chambolle-Pock algorithm. It yields the following iterates:

$$\begin{cases} x_{n+1} = \operatorname{prox}_{\tau R}(x_n - \tau K^* z_n) \\ \tilde{x}_{n+1} = 2x_{n+1} - x_n \\ z_{n+1} = \operatorname{prox}_{\sigma N^*}(z_n + \sigma K \tilde{x}_{n+1}) \end{cases} \quad (7)$$

Closed-form expressions for K^* and $\operatorname{prox}_{\sigma N^*}$ can be derived analytically (cf. (REF APENDIX)). They are given by

$$\operatorname{prox}_{\sigma N^*}(x) = \frac{x - \sigma s}{\sigma + 1} \quad \text{and} \quad K^* z = z * \overline{y}, \quad (8)$$

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

III. Algorithmic approach

A. First half of the problem (I_1).

Falta: explicar RCP; potser DCP

The way of inverting the formula in eq. (3) is the same as in (CITAR ARTICLE REFERÈNCIA), although the implementation is fully self-made (the original implementation is written in MATLAB and, to the best of the author's knowledge, only publicly available in binary form).

First, the background light is chosen by iteratively splitting the image into four regions. 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 our 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.

Falta explicar com se calcula es transmission map i es guided filter. He posat s'algoritme pes transmission map a 2.

Once the background light B has been computed, it is possible to estimate the trasmission map t. We proceed in a similar fashion to (CITAR ART. REF): first, a coarse version of the transmission map t_0 is obtained via

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

where the *saturation map* S is defined as 0 for black $(0, 0, 0)$ pixels and $S = 1 - \frac{\min(I^r, I^g, I^b)}{\max(I^r, I^g, I^b)}$ otherwise.

Then, a fine version of the transmission map, t, is obtained by passing t_0 through a guided filter with I^r as the guide. Cal explicar què fa es guided filter? O basta citar?

Finally, we estimate a first version of J as follows, for each color channel $c \in \{r, g, b\}$:

$$J^c(x) = \frac{I^c(x) - B^c}{\max(t(x), 0.1)} + (1 - B^c) \cdot B^c \quad (9)$$

B. Second half of the problem (I_2).

In order to solve the minimization subproblems in eq. (6), the iterative schema in eq. (7) with the formulae in eq. (8) is used, with the aforementioned symbolic substitutions. The missing operator $\operatorname{prox}_{\tau R}$ 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.

The final algorithm used for obtaining an approximate solution to eq. (5) is summarized in Algorithm 3.

Algorithm 1: Estimate background light.

Data: I
Result: J
img $\leftarrow I$;
while *Size(img) > min_size* **do**
  quadrants $\leftarrow$ Split(img);
  max_score $\leftarrow 0$;
  for q *in quadrants* **do**
    $\mu \leftarrow$ Mean(q);
    $\sigma \leftarrow$ Std(q);
    score $\leftarrow \mu - \sigma$;
    if *score > max_score* **then**
      max_score $\leftarrow$ score;
      img $\leftarrow q$;
    end
  end
end

Algorithm 2: Estimate scene radiance and transmission map.

Data: I, B, patch radius r
Result: J_0, t
; /* Compute coarse transmission map */
$S \leftarrow$ Saturation(I);
Initialize t with the width and height of I;
for *pixel* x **do**
  ; /* Ω_x: patch around x, radius r */
  r_min $\leftarrow \min_{y \in \Omega_x}(1 - I^r[y])/(1 - B^r)$;
  g_min $\leftarrow \min_{y \in \Omega_x}(I^g[y])/B^g$;
  b_min $\leftarrow \min_{y \in \Omega_x}(I^b[y])/B^b$;
  s_min $\leftarrow \min_{y \in \Omega_x}(S[y]) \cdot \lambda$;
  $t[x] \leftarrow 1 -$ min(r_min, g_min, b_min, s_min)
end
; /* Refine transmission map */
$t \leftarrow GuidedFilter(I^r, t)$;
; /* Estimate J_0 */
for *channel* c **do**
  $J_0^c \leftarrow (I^c - B^c)/(max(t, 0.1)) + (1 - B^c) \cdot B^c$
end
return J_0, t

Algorithm 3: Solve second half of the problem.

Data: I_2, t, J_0
Data: σ, τ
Result: J
$J \leftarrow J_0$;
Initialize g;
Initialize dual variables $\tilde{g}$, $\tilde{J}$;
for $n = 1$ **to** G **do**
  ; /* Fix J, estimate g. */
  $\overline{J} \leftarrow DoubleFlip(J)$;
  for $s = 1$ **to** S **do**
    $tmp \leftarrow NeuralNet_{ns}^{g}\left(g - \tau \cdot \tilde{g} * \overline{J}\right)$;
    $g \leftarrow 2 \cdot tmp - g$;
    $\tilde{g} \leftarrow (\tilde{g} + \sigma \cdot (g * J) - \sigma \cdot I_2)/(\sigma + 1)$;
  end
  ; /* Fix g, estimate J. */
  $\overline{g} \leftarrow DoubleFlip(g)$;
  for $s = 1$ **to** S **do**
    $tmp \leftarrow NeuralNet_{ns}^{J}\left(J - \tau \cdot \left(\tilde{J} * \overline{g}\right)\right)$;
    $J \leftarrow 2 \cdot tmp - J$;
    $\tilde{J} \leftarrow (\tilde{J} + \sigma \cdot (J * g) - \sigma \cdot I_2)/(\sigma + 1)$;
  end
end
return J / t

C. Implementation

Xerrar de U2Fold, la implementació en pytorch dels algoritmes d'abans i de la UNET (he de mirar bé si lo que faig és realment una UNET pq això de sumar en comptes de concatenar és raro), etc.

IV. Experimentation

A. Algorithmic selection of hyperparameters

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.

Of course, the target metric plays a central role in determining the output choice. 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 [TODO: ACABAR]

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. APPENDIX

Only if deemed necessary.