Evaluation and Optimization of Underwater Image Restoration Algorithms

Alan Le Boudec, Artur Mkrtchyan, Barbara Dzaja, Vincent Rodin, Hai Nam Tran

alan.leboudec@etudiant.univ-brest.fr

9/9/2021

Plan

- Introduction
  - Context
  - Problem statement
  - Objectives
- Background
- Approach
- Evaluation
- Future work

Introduction

Context

- Cooperation with **Sea-ue**
- Underwater **drone** with cameras provide by the University of Split
- Poor underwater image quality
- Image processing required for recognition or object detection

Problem statement

- Onboard processing: limited energy and hardware resources
  - Limited speed of underwater data
  - Standalone program
- Real time constraint vs. quality results
  - Best quality means more processing time
  - Immediate processing and therefore fast execution

Introduction

Objectives

- Apply and test state-of-the-art algorithms to improve the quality of underwater images.

- Evaluation of the various possibilities for optimising these algorithms for real-time application.

Background

Algorithms

- Five state-of-the-art underwater image processing algorithms
  - Underwater Hazelines (*Berman et al., 2017*)
  - Local color mapping and color transfert (*Protasiuk et al., 2019*)
  - Fusion enhancing (*Ancuti et al., 2012*)
  - Backscatter removing (*Zhang et al., 2016*)
  - Automatic red-channel underwater image restoration (*Galdran et al., 2012*)

Background

Evaluation criteria

Full reference

Compare two images using mathematical calculations.

- Peak Signal to Noise Ratio(*PSNR*)
- Visual Information Fidelity (*VIF*)
- Information Fidelity Criterion (*IFC*)
- Structural Similarity (*SSIM*)
- Mean Square Error (*MSE*)
- Norm or Euclidean distance

No reference

Compute the criteria for the input image

- Blind/Referenceless Image Spatial Quality Evaluator (*BRISQUE*)
- Naturalness Image Quality Evaluator (*NIQE*)
- Underwater Image Quality Measures (*UIQM*)

Approach

Approach divided in 2 steps:

- **Evaluation algorithms by criteria**
- **Optimization of the best suitable algorithms**

The main lines of our approach:

- Choose 15 good quality images
- Degrade the images
- Apply the algorithms
- Quality measurement using evaluation criteria
- Algorithms runtime execution measurement
- Choose the best algorithm on average
- Optimise the chosen algorithm

Approach

1. Benchmark

Set of 15 images:

- Good quality images
- Anticipate the results gained with the algorithm

Original

Degrade by ourself:

- Blue filter
- Matlab "speckle" noise

Degrade

Approach

2. Criteria evaluation

9 criteria are selected to compare each algorithm,
separated in 2 types:

- No reference
- Full reference

Compute each metric for all algorithms.

Point system to give a score, if the algorithm is the best for 1 criterion it wins 1 point on this metric for this image.

Highest score at the best image quality.

Approach

3. Optimization

Several steps of **Optimization**:

1. Source code improvement
    a. Compact code
    b. Code adaptation (GUI)
2. Improve the speed of image processing
    a. Targeted the incriminating functions
    b. Call to C function or Matlab library
3. Adaptation to video processing :
    a. Creation of the function that will apply algorithm in a loop on each frame of the video stream
    b. The parallelism of processing

Evaluation

2 experiments:

- Measure the quality and make comparison between the algorithms
- Measure the execution time for optimization
  - Calculate the execution time after optimization of the best image quality algorithm.

Working environment:
OS Ubuntu 18.04,
Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz,
16 GB of RAM

Experiment 1
Objective: Evaluate image quality

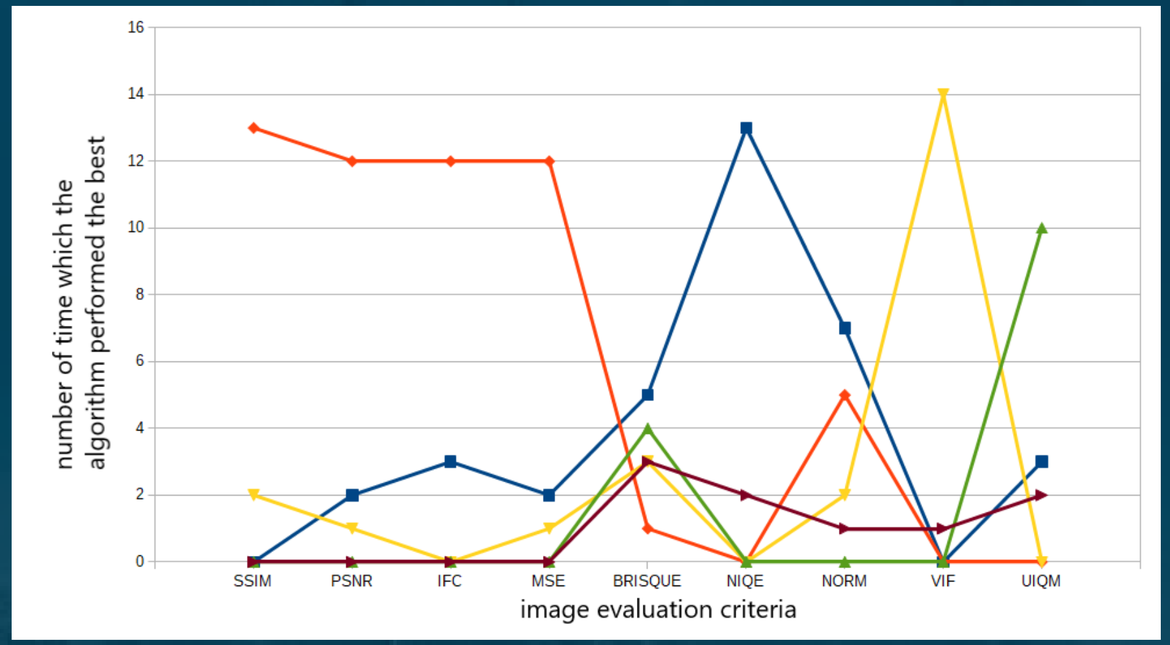
Experiment 2.1
Objective: Evaluate execution time (without optimization)

Experiment 2.2
Objective: Evaluate execution time (with optimization)

Evaluation

The measurments of ***Experiment 1*** show that:

- ***Fusion*** *algorithm is the best 4 times*
  - *PSNR (Full reference)*
  - *SSIM (Full reference)*
  - *IFC (Full reference)*
  - *MSE (Full reference)*
- ***Automatic Red-channel Underwater Image Restoration*** *algorithm is the best 3 times*
  - *BRISQUE (No reference)*
  - *NIQE (No reference)*
  - *NORM (Full reference)*



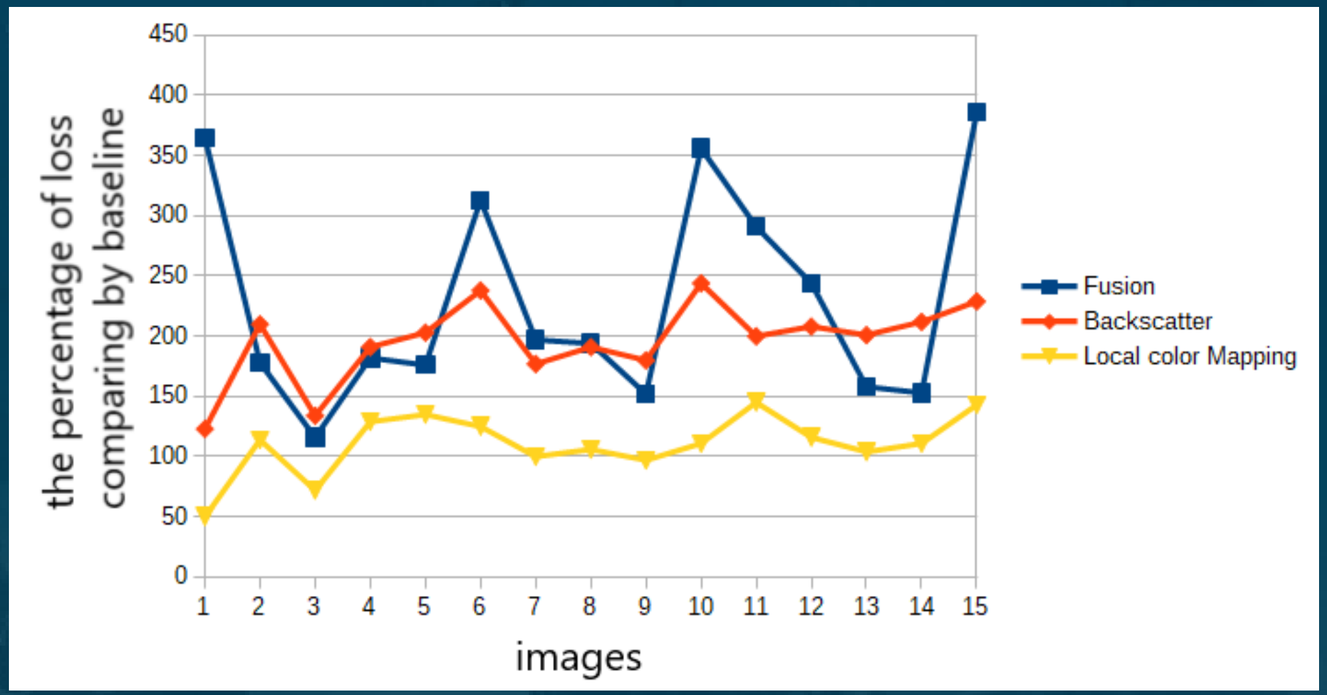
Evaluation

The measurments of ***Experiment 2.1*** show that:

- ***Automatic red-channel underwater image restauration*** *algorithm is the fastest and was chosen as a baseline*

- ***Hazelines*** *algorithm has been removed which is almost 20 times longer than others*

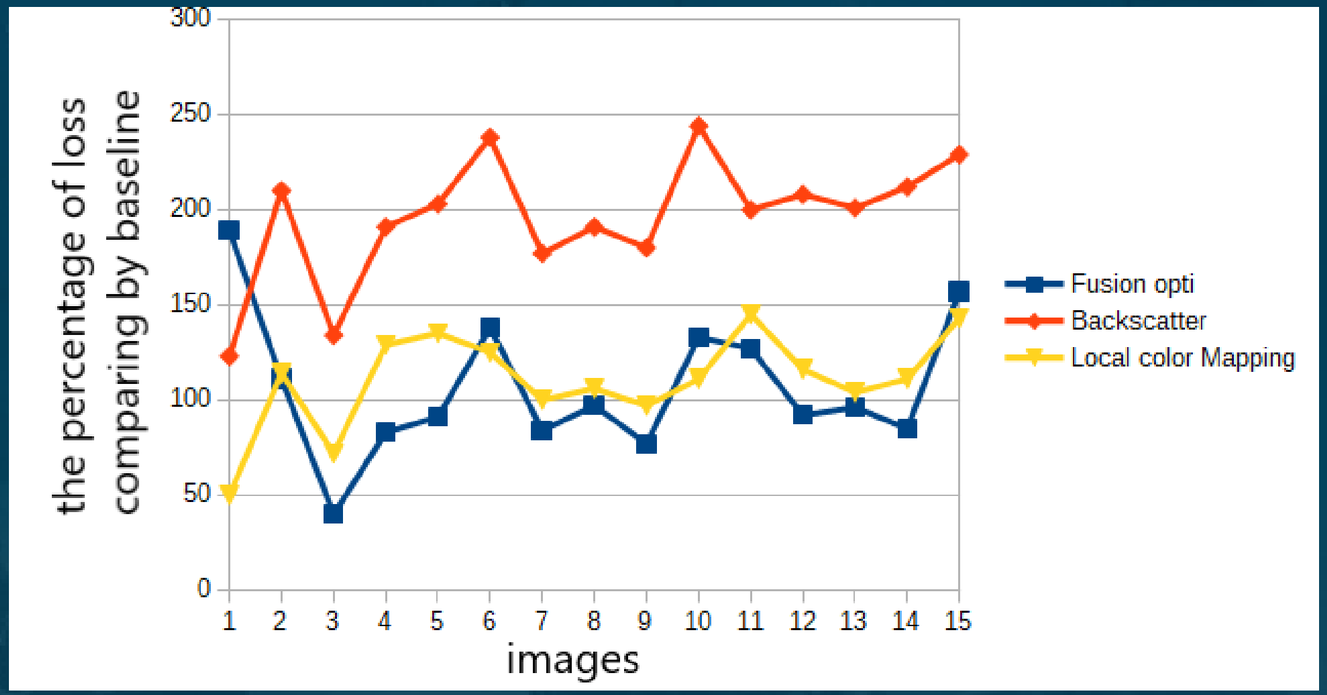
- ***Local color Mapping*** *is the best one compare to the baseline*



Evaluation

The measurements of ***Experiment 2.2*** reveals that :

- ***Fusion*** *show the best results on runtime measurments.*
- *Before optimization* ***Fusion*** *execution time : 3 s by frame.*
- *After optimization* ***Fusion*** *execution time : 2 s by frame.*



Evaluation

The evaluation of algorithms with these differents criteria reveals the following:

- ***Fusion*** and ***Automatic Red channel Underwater Image Restoration*** algorithms have the best image quality results.

- To apply a real-time *video processing* we need to process 30 images/frames per second.

- ***Fusion*** algorithm after improvement, and despite a gain of 50%, takes 2 seconds in average per image/frame, this would mean 1.5 minutes for 1 second of video.

Future work

- Test the algorithms directly on the underwater rover.
- With the results, see another way to optimize as rewrite in C code.
- Comparing other algorithms
- Finding alternative criteria for evaluation