High-Fidelity Deep-Sea Perception Using Simulation in the Loop [★]

Tobias Doernbach [*] Arturo Gomez Chavez [*]
Christian A. Mueller [*] Andreas Birk [*]

[*] *Jacobs University Bremen, Germany*

Abstract: Deep-sea robotic operations require a high level of safety, efficiency and reliability. In the development stage of such systems, measures have to be taken into account to validate performance in order to assess the achievement of these requirements. In the context of continuous system integration, we proposed a simulation-in-the-loop framework focusing on the mitigation of discrepancies between simulation and real-world conditions.
While in our previous work we mainly targeted a high-fidelity simulation that embeds spatial conditions from recorded real-world data, this work emphasizes environmental conditions. We propose an optimization cycle which allows to enhance the fidelity of simulated underwater camera images in a *backward optimization* step and to enhance real-world images with knowledge available in simulation in a *forward optimization* step. Experimental results show that the proposed methodology optimized both simulation and real imagery, and subsequently ensures high fidelity.

Keywords: Marine systems, deep-sea robotics, simulation in the loop, sensor data simulation, robot perception, parameter optimization, self-optimizing systems.

1. INTRODUCTION

In recent years, deep-sea research and commercial activities have been profiting from the rapid developments on Unmanned Underwater Vehicle (UUV) capabilities which can be used for inspection, mapping. manipulation and recovery tasks. Especially in inaccessible areas which may be hazardous to humans UUV are deployed in user-guided, semi-autonomous and fully autonomous missions in order to avoid dangerous incidents for human divers.

However, the development and continuous evaluation of such underwater robotic systems typically requires a large amount of manpower and equipment to deploy, operate, and retrieve the robot offshore. This rapidly increases the effort and cost of each development cycle. Specialized testing and deployment strategies are required because effort and costs in case of failure are higher by several orders of magnitude than in ground robotics, for example, when a UUV malfunctions in deep sea and cannot be retrieved anymore.

Consequently, efficient strategies have to be incorporated to validate effectiveness, robustness and reliability of the developed capabilities. Therefore, we recently presented a *simulation in the loop* (SIL) methodology (see Fig. 1) that uses a simulator for underwater robotic activities and integrates parts of the development stack with real-world data recorded from field trials, see Mueller et al. (2018). However, in order to achieve high-accuracy results in terms of realisticness when using the simulation, its parameters have to be optimized prior to deployment with respect to fidelity of real-world sensor data and behavior.

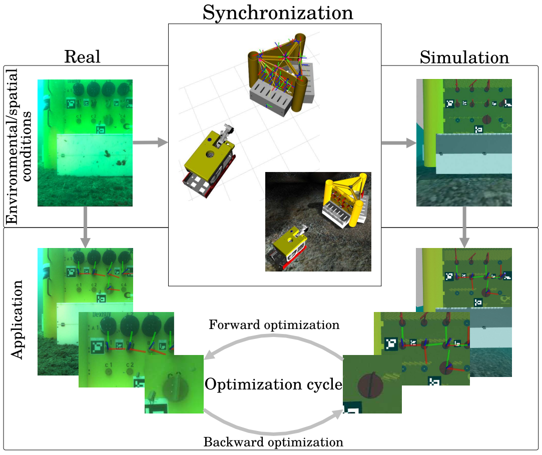

Fig. 1. *Simulation in the loop* in a perception task application (valve and lever pose estimation, Mueller et al. (2018)) leading to an optimization cycle as presented in the current work.

Hence, in this work we propose an approach for high-fidelity simulation of underwater environmental conditions such as degree of visibility captured through an ROV-onboard camera. These conditions can be adapted to the current mission on the fly and allow for integration testing and performance evaluation of perception-related algorithms and methods. Consequently, we provide an example use case of deep-sea camera image enhancement

[★] TD, formerly known as Tobias Fromm 0000-0001-6488-8211, AGC 0000-0002-7132-1026 and CAM 0000-0003-1895-987X contributed equally to this work and share first authorship. The research leading to the presented results has received funding from the European Union's Horizon 2020 Framework Programme (H2020) within the project (ref.: 635491) "Effective dexterous ROV operations in presence of communication latencies (DexROV)".

which profits from this high-fidelity simulation and shows the applicability of our method.

2. SIMULATION IN THE LOOP

In the context of the EU-funded research project *DexROV* (Effective Dexterous ROV Operations in Presence of Communication Latencies Gancet et al. (2016)), in our previous work Fromm et al. (2017) we proposed a versatile integration and validation architecture that allows for pre-deployment testing using simulated and real system components besides each other in a seamless way. That work focused on the *continuous system integration and deployment* of a fully integrated system that may contain simulated components due to their developmental stage.

In our recent work in Mueller et al. (2018), however, we presented a *simulation in the loop* (SIL) methodology (see Fig. 1) which allows for extensive *system benchmarking*. Hence, our particular focus is set on closing the *discrepancy* between simulated and real-world data by augmenting simulation with feedback extracted from collected real-world field trial data. Such feedback may consist of specific *environmental* or *spatial* conditions like degradation of visibility due to haze and low illumination, as well as object pose estimation and robot localization events.

As a result, the proposed method does not only provide the benefits of continuous system integration described in Fromm et al. (2017) like *distributed deployment*, *interface/pipeline testing*, *regression/degradation testing*, *parallelized testing* or *fault recovery/safety testing*. It also provides an augmented virtual environment reflecting conditions from real-world field trials that allows to thoroughly and concurrently investigate, compare, and benchmark behaviors of system components under both real, simulated, and hybrid conditions.

Initially, in Mueller et al. (2018), our focus was on spatial conditions such as object locations, for instance of the ROV and an artificial testing panel used for evaluating autonomous deep-sea manipulation (see Figure 1). In this paper, however, we particularly emphasize environmental conditions, specifically the enhancement of visual fidelity between real-world and simulated data by adapting visual conditions in simulation to those found in real data (see Section 3).

Given this framework which provides spatial as well as environmental fidelity in both the simulation and the real-world domain, we go a further step ahead by closing the loop with enhancing real camera images with knowledge available in simulation (see Section 4).

In the literature, the term *simulation in the loop* has previously been used in different, inconsistent contexts, for instance, with simulation as a sampling and verification step within motion planning approaches (Heckman et al., 2015). Another example is its use as a synonym for *continuous system integration* as introduced in Fromm et al. (2017), e.g. by Iivari and Ronkainen (2015). Other authors even use this term to merely indicate the substitution of real hardware with a simulated representation thereof, like in Cichon et al. (2016).

parameter	**description**
a_H	attenuation factor (hue)
a_S	attenuation factor (saturation)
a_V	attenuation factor (value)
b_H	background intensity (hue)
b_S	background intensity (saturation)
b_V	background intensity (value)
ρ_F	fog density

Table 1. Camera simulation parameters adjusted in the process of our method

For practitioners, our closed-loop SIL methodology is useful two-fold: it firstly allows for *backward optimization* of the simulation environment using real-world data to create a high-fidelity representation of the conditions founds in the field. Automatic adaptation to the respective condition of a mission can be performed during the field trials with the method explained in Section 3, the resulting representation can be used for efficient continuous system integration. Secondly, it provides the ability to perform *forward optimization* of real-world perception and autonomous behavior algorithms using the additional information provided by the simulation with synchronized spatial and environmental conditions.

3. UNDERWATER CAMERA SIMULATION FIDELITY ENHANCEMENT

3.1 Realistic Underwater Camera Simulation

In order to achieve environment conditions in simulated underwater camera images which are close to reality, we primarily adapt the light behavior to replicate color attenuation. The simulated stereo camera applies an exponential attenuation on the pixel intensity as described in Marcusso et al. (2016):

$$i_c^* = i_c e^{-za_c} + (1 - e^{-za_c})b_c \quad \forall c \in \{R, G, B\} \qquad (1)$$

where i_c and b_c correspond to the pixel and background intensity value for color channel c, a_c is a color-dependent attenuation factor, and i_c^* is the attenuated color value. The attenuation depends on the distance z to the object projected on the camera pixel, which is extracted directly from the simulator depth-camera plugin.

Additionally, because deep-sea images are usually prone to light scattering, we integrated the fog functionality present in the used simulator, Gazebo (Koenig and Howard, 2004), with its density as another degree of freedom to simulate this behavior. All this functionality has been implemented using the Gazebo camera plugin to be usable in the given setup.

3.2 Parameters to optimize

In order to provide high-fidelity camera images in simulation whose environmental conditions correspond with real-world data, we use a flexible camera model which exposes a set of parameters adjustable for this purpose. The selected camera simulation parameters to optimize in our method are shown in Table 1. In this work, we present an approach to optimize this 7-dimensional parameter set in an unsupervised manner using specific optimization criteria described in the next subsection.

3.3 Image Quality Optimization Criteria

As a criterion for image similarity, many different measures exist. Our SIL landmark is an artificial object with defined features (see *DexROV* panel in Fig. 2) which facilitates a highly accurate pose estimation of the object even under challenging conditions as described in Mueller et al. (2018). This object is exploited as reference for optimization purposes and therefore we assume to be visible in the real-world images, otherwise no visual reasoning would be feasible.

Images captured from this object provide visual features which are suitable to be reflected in the grayscale-based Feature Similarity Index FSIM (Zhang et al., 2011) and its color extension FSIMc. Therefore, we take the latter as the first image quality optimization criterion θ_1 to determine the similarity between the real-world ground truth image I_g and the simulated image I_i for the respective agent's parameter set a_i so that

$$\theta_1(a_i) = \mathrm{FSIMc}(I_g, I_i). \tag{2}$$

Additionally, color information plays an important role in underwater imaging as the parameter set in Table 1 indicates. Hence, we create three single-channel histograms on the HSV (Hue, Saturation, Value) color space of the real-world ground truth I_g and simulated image I_i and calculate mean of their Bhattacharyya distances d_B (Bhattacharyya, 1943). However, spatial information must not be neglected completely in this process since, due to the nature of simulation, many uniformly-colored areas are present in the image. Whole-image color histograms may lead to overfitting on the predominant color, so instead we divide the image 10×10 regions of interest and calculate the mean of their single-channel histogram distances d_B. The median of all regions, which is more resistant to outlier regions, is then taken as the second optimization measure:

$$\theta_2(a_i) = \operatorname*{median}_{r\in R}\left(\operatorname*{mean}_{c\in C}\left(d_B(h_c(I_{g,r}), h_c(I_{i,r}))\right)\right) \tag{3}$$

where R is the set of regions of interest in an image, $C \subseteq \{H, S, V\}$ are the color channels to regard and $h_c(\cdot)$ is the histogram of color channel c. HSV is more invariant than RGB to color dependencies, so the user can decide whether during optimization they want to set focus on color or luminance for their application by using the respective channels only. In our baseline application, we used all three channels as in our Experimental Results section.

3.4 Particle Swarm Optimization

In our previous work (Mueller et al., 2018), we used handcrafted values for the mentioned parameters to achieve satisfactory results. Manual parameter tuning may be difficult, however, depending on on their number and range. Moreover, parameters can form strong relationships between each other.

Since the parameter optimization problem presented here is discontinuous, all optimization methods relying on a continuous function describing the problem will fail. As for pattern search-based methods, the brute-force Grid Search is one of the most trivial, but least efficient ones.

```
input: parameter space $\mathcal{A}$, maximum number of iterations $t_{max}$, real image $I_g$, color channels $C \subseteq \{H,S,V\}$
initialize all agents $a_i \in \mathcal{A}$ randomly
initialize globally optimal agent $a_{\mathrm{opt}} \leftarrow a_0$
$t \leftarrow 0$
while $t < t_{max}$ do
    for all $a_i \in \mathcal{A}$ do
        generate simulated camera image $I_i(a_i)$
        $\theta_1(a_i) \leftarrow \mathrm{FSIMc}(I_g, I_i)$
        $\theta_2(a_i) \leftarrow \operatorname*{median}_{r\in R}(\operatorname*{mean}_{c\in C}(d_B(h_c(I_{g,r}), h_c(I_{i,r}))))$
        $\theta(a_i) \leftarrow (\theta_1(a_i) + \theta_2(a_i))/2$
        if $\theta(a_{\mathrm{i}}) > \theta(a_{\mathrm{opt}})$ then
            update optimal agent $a_{\mathrm{opt}} \leftarrow a_i$
        end if
    end for
    update all agents' parameter sets to move
    the swarm towards the current optimum
    $t \leftarrow t + 1$
end while
output: globally optimal agent $a_{\mathrm{opt}}$
```

Algorithm 1. Particle Swarm Optimization

Instead, we utilize *Particle Swarm Optimization (PSO)*[1] (Kennedy and Eberhart, 1995).

PSO can be used without assumptions about the data to optimize. As an input, it is able to deal with a number of numerical parameters in arbitrary ranges (for our case, see Table 1) and needs an evaluation function towards which to optimize ($\theta(a_i)$). In our case, we provide $\theta_1(a_i)$ and $\theta_2(a_i)$ of which we take the mean to result in the overall evaluation function

$$\theta(a_i) = \frac{\theta_1(a_i) + \theta_2(a_i)}{2}. \tag{4}$$

Initially, PSO will create a random configuration within the 7-dimensional space formed by the camera simulation parameters and randomly distribute a swarm of agents around it.

As shown in Algorithm 1, in every iteration t a simulated camera image I_i is generated depending on the parameters of the respective agent a_i and the evaluation function $\theta(a_i)$ is resolved. Afterwards, the center of the agent distribution shifts in the direction of the current global optimum. After a specified maximum number of iterations, the position of the globally optimal agent determines the optimized set of camera simulation parameters to be used from then on.

PSO itself has some parameters to tune which resemble the number of agents $|\mathcal{A}|$, the agent velocity ω, per-agent ϕ_P and swarm inertia ϕ_G. In our approach, we heuristically tuned these to minimize the risk of undesired local minima with $|\mathcal{A}| = 20$, $\omega = 0.4$, $\phi_P = 0.3$ and $\phi_G = 0.3$. These values should be reasonable to use in any similar setting so that our method enables the users to closely adapt to the concrete on-site underwater environment conditions. Meta-optimizing the Particle Swarm Optimizer has been subject to research (e.g. Carlisle and Dozier (2001)), but we regard this topic as out-of-scope for the work proposed in this paper.

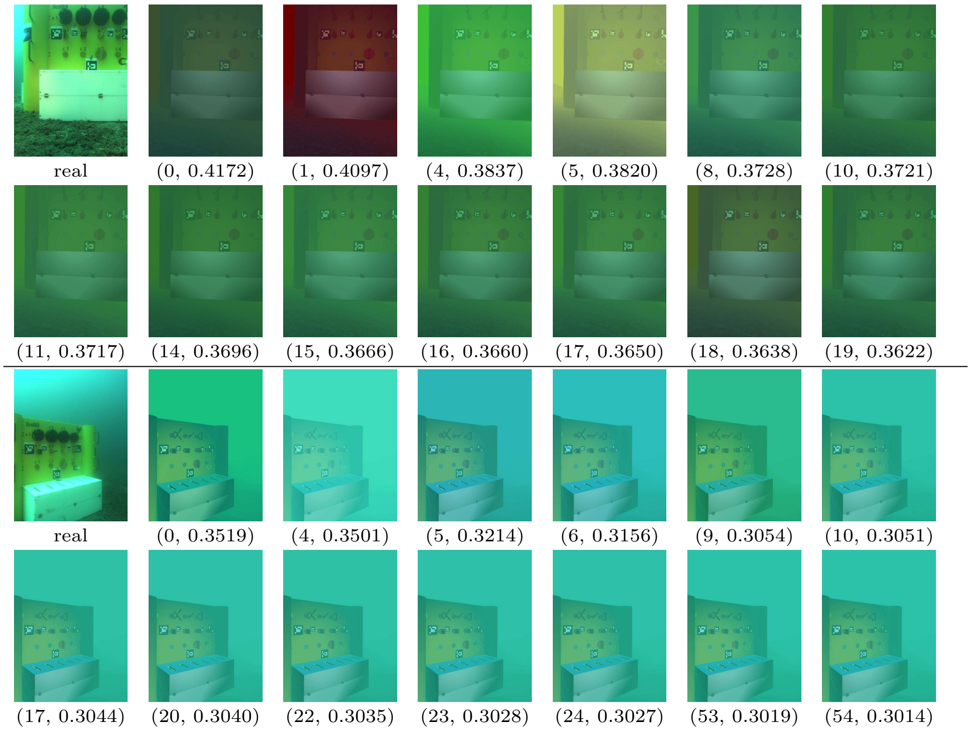

Fig. 2. PSO results: (t, θ) for real-world (top left) to optimized image (bottom right) in a scene focused on artificial objects (top) and on ocean environment (bottom)

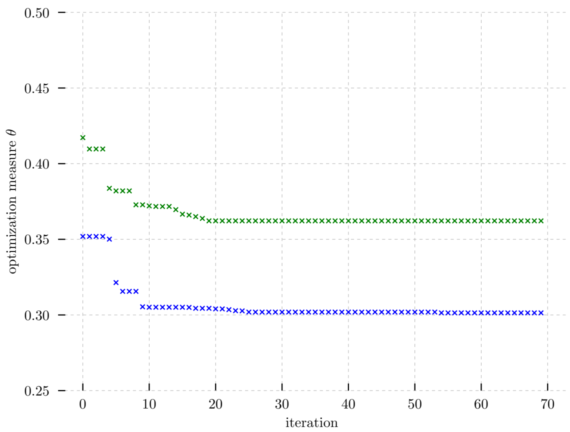

Fig. 3. PSO results for a scene focused on artificial objects (green) and on ocean environment (blue)

3.5 Camera Simulation Fidelity Enhancement Results

The image sequence in Figure 2 shows the optimization steps for two scenes with different characteristics with the θ optimization measure improving over time as in Figure 3. It can be observed that, from the first iteration already, the resulting images are in a reasonable parameter region, however, their fidelity improves over the course of the optimization. We set the maximum number of PSO iterations $t_{max} = 100$, but the parameters converged earlier already, for 18 and 53 iterations, respectively. Since the optimization measures θ_1 and θ_2 do not explicitely contain blurriness or light scattering, PSO overemphasizes the fog density ρ_F by some extent. This could be overcome by integrating a blurriness measure, however, such fine-grain optimization may not be possible with the current architecture.

Nevertheless, for *fully-automatic* backward optimization depending only on collected real-world data, the results are convincing because the optimization procedure can be run in the beginning of every dive before the activation of (semi-)autonomous robot capabilities without the need for manual tuning. An additional feature is the fact that the backward optimization depends on the scene contents, as can be seen in the different results for artificial object-based and clear ocean-based images (Figure 2). Hence, depending on which methods or algorithms are going to be tested on what kind of scene, running our backward optimization method guarantees simulated camera parameter settings that correspond with the respective scene.

4. DEEP-SEA CAMERA IMAGE ENHANCEMENT

In this section, we discuss the *forward optimization* of the real-world data based on the synchronized information from the simulation. First, a brief summary of image enhancement methods for underwater is presented, followed by analysis of results from a DexROV field trial.

4.1 Image Enhancement Using Dark-Channel Prior

Backscattering is the main source of degradation in underwater images as pointed out in Schechner and Karpel (2005); Treibitz and Schechner (2009), which causes a haze effect. The *dehazing* process has been identified as an ill-posed inversed problem, since atmospheric light and depth of observed objects varies continuously. In order to

[1] used implementation: `https://github.com/tisimst/pyswarm`

mitigate this effect, the dark-channel prior (DCP) method has been proposed (Zeng and Dai, 2016) and used actively in recent years; this method works on the principle that image patches in non-object regions (sky, water) have very low intensity for at least one color channel. In this way, bright patches in object regions indicate the presence of backscatter and its effects on the image can be reverted.

Basically, DCP and derived methods used the estimated dark channel to compute the image atmospheric light and transmission map, which are denoted as e^{-za_c} and b_c in Equation 1, respectively. A comparison of dark channel-based dehazing methods can be found in Lee et al. (2016).

4.2 Image Enhancement Using SIL Forward Optimization

Taking advantage of the inferred spatial conditions between the simulated and real environment, the ground-truth *transmission map*

$$t(z) = e^{-za_c} \tag{5}$$

can be retrieved using a known model of the target object (DexROV panel) and the simulator's depth camera. Thus, the dehazed image can be constructed by solving for i_c in 1. As for the atmospheric light b_c, the simulated depth camera can accurately segment free-space from objects; subsequently, we choose b_c as the average intensity value of the free (water) region.

Our method works image frame-wise and does not optimize any global parameters, but for the given problem of image dehazing, no global method can be constructed because the amount of haze and hence the target color intensity is depth-dependent. As an additional asset, our method is able to obtain enhanced deep-sea images online, opposite to current dark channel methods which consume significant computation time during the transmission map refinement, 10 s up to 25 s for images with resolution of 600 × 400 pixels according to Lee et al. (2016). For our application, 640 × 480 pixels images are used.

4.3 Deep-Sea Camera Image Enhancement Results

The images in Fig. 4 show a comparison between the DCP and SIL enhancement methods.

Qualitatively, it can be observed that the SIL-produced image has higher contrast than both the original image and the DCP-enhanced one. This is particularly visible on the handles which are integral components of deep-sea operations such as detection, tracking and manipulation of valves, levers etc. Likewise, through the use of the synchronized simulation, the water region can be clearly segmented and its original can be preserved by the largest extent. Some imperfect transmission maps from DCP cause color mismatches as visible at the panel bottom (Fig. 4 center). Refinement techniques exist, such as soft-matting, however they demand for extra computation time. On the other hand, we can see that the sea floor shows better contrast for DCP as, in the considered scenario, there is no previously known model for it. Hence, with our method, the colors cannot be optimized for this region since no depth information is available and consequently, the sea floor shifts towards the ambient color.

descriptor	**real**	**DCP**	**SIL**
ORB	74	61	**82**
BRIEF	110	108	**124**
SIFT	**123**	102	118
SURF	167	131	**187**
KAZE	128	148	**156**
AKAZE	86	80	**94**

Table 2. Camera image enhancement results: number of features per image descriptor for real, DCP-enhanced and SIL-enhanced image

In order to evaluate the impact on object recognition or manipulation tasks, we compute several feature descriptors in different images showing instances of the panel. For this the original and DCP and SIL enhanced images are used (see Fig. 5). As a measure, we count the number of feature matches, found through k-nearest-neighbor, whose Euclidean distance is less than a threshold of 0.25 in the feature space.

In Table 2, it can be seen that overall more features are matched in the SIL-enhanced image, which should improve object detection and tracking algorithms. For example, Fig. 5 shows that the region marked in red with the furthest handle in the panel exhibits feature matching only after the SIL forward optimization. As for the DCP enhancement, it causes feature degradation despite the fact that it recovers part of the original colors of the image. Summarized, for feature-reliant detection and tracking techniques, the presented method allows for an improvement of features to be detected and hence for the information available for the respective algorithms.

5. CONCLUSION

In the context of *continuous system integration and deployment* (Fromm et al. (2017)), in our previous work, we proposed a *simulation in the loop* (SIL) architecture (Mueller et al. (2018)). Deep-sea robotics projects like DexROV (Gancet et al. (2016)) can profit from such an architecture to provide an assessment of the system component performance with respect to robot self-localization, scene modeling, object perception and other (semi-)autonomous functionality.

However, meaningful assessment can only be achieved if the fidelity between simulation and real-world conditions exceeds a certain threshold. In our previous work we presented methods for *backward optimization* concerning spatial conditions. In this publication, we used the same setup to improve the environmental conditions as to lift the simulation utility above said threshold. Exploiting this setup, we employed *forward optimization* on an image enhancement task as to make the results more useful in an underwater context.

Summarized, backward/forward optimization in a closed-loop simulation allows for a significant utility increase in terms of algorithm and method development, especially in harsh and remote areas with difficult visual conditions like found in deep-sea robotics.

REFERENCES

Bhattacharyya, A. (1943). On a measure of divergence between two statistical populations defined by their

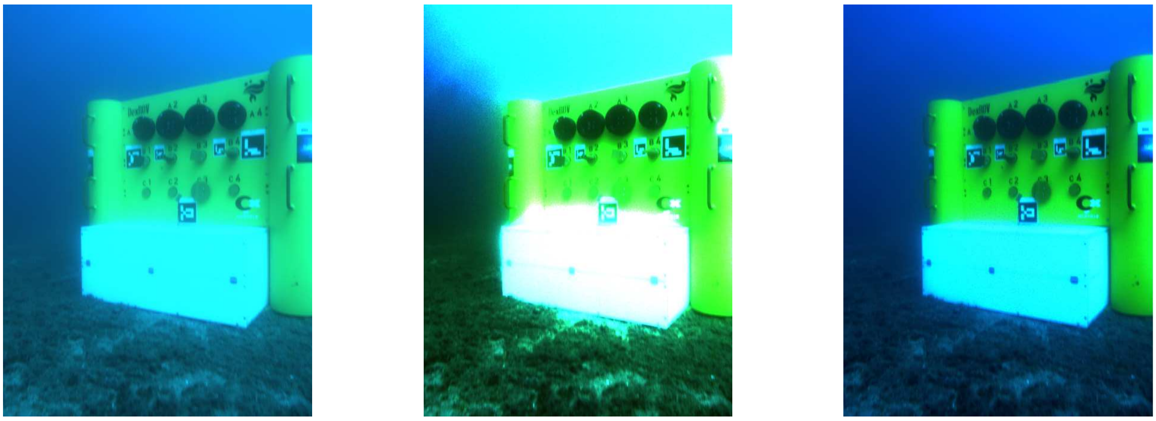

Fig. 4. Image enhancement comparison of real-world camera image (left) with DCP (Zeng and Dai (2016), center) and our SIL forward optimization method (right)

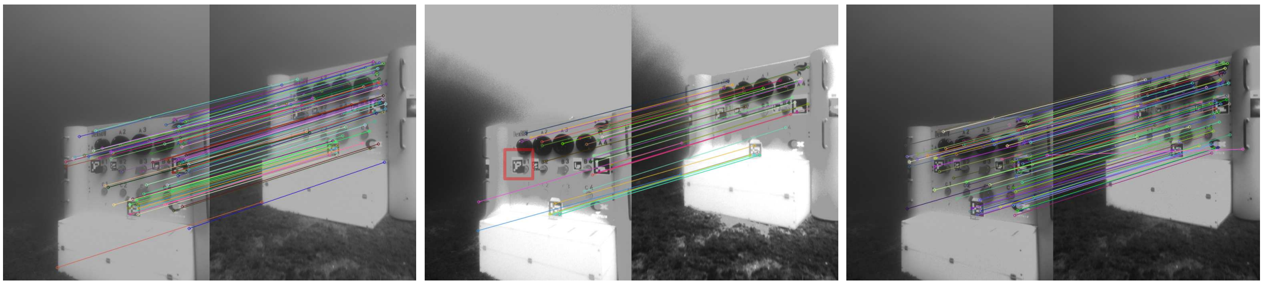

Fig. 5. SURF feature correspondence matching between different artifical object instances: real-world camera image (left), DCP-enhanced image (center) and SIL-enhanced image (right)

probability distributions. *Bulletin of the Calcutta Mathematical Society*, 35, 99–109.

Carlisle, A. and Dozier, G. (2001). An Off-The-Shelf PSO. In *Workshop on Particle Swarm Optimization*.

Cichon, T., Loconsole, C., Buongiorno, D., Solazzi, M., Schlette, C., and Frisoli, A. (2016). Combining an exoskeleton with 3D simulation in-the-loop. In *International Workshop on Human-Friendly Robotics*.

Fromm, T., Mueller, C.A., Pfingsthorn, M., Birk, A., and Di Lillo, P. (2017). Efficient Continuous System Integration and Validation for Deep-Sea Robotics Applications. In *OCEANS (Aberdeen)*.

Gancet, J., Weiss, P., Antonelli, G., Pfingsthorn, M., Calinon, S., Turetta, A., Walen, C., Urbina, D., Govindaraj, S., Letier, P., Martinez, X., Salini, J., Chemisky, B., Indiveri, G., Casalino, G., di Lillo, P., Simetti, E., de Palma, D., Birk, A., Fromm, T., Mueller, C., Tanwani, A., Havoutis, I., Caffaz, A., and Guilpain, L. (2016). Dexterous Undersea Interventions with Far Distance Onshore Supervision: the DexROV Project. In *IFAC Conference on Control Applications in Marine Systems*.

Heckman, C., Keivan, N., and Sibley, G. (2015). Simulation-in-the-loop for Planning and Model-Predictive Control. In *Robotics: Science and Systems, Realistic, Rapid, and Repeatable Robot Simulation Workshop*.

Iivari, A. and Ronkainen, J. (2015). Building a Simulation-in-the-loop Sensor Data Testbed for Cloud-enabled Pervasive Applications. *Procedia Computer Science*, 56, 357–362.

Kennedy, J. and Eberhart, R. (1995). Particle Swarm Optimization. In *International Conference on Neural Networks*.

Koenig, N. and Howard, A. (2004). Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator. In *International Conference on Intelligent Robots and Systems*.

Lee, S., Yun, S., Nam, J., Won, C., and Jung, S. (2016). A review on dark channel prior based image dehazing algorithms. *EURASIP Journal on Image and Video Processing*, 2016(1), 4.

Marcusso, M., Scherer, S., Voss, M., Douat, L., and Rauschenbach, T. (2016). UUV Simulator: A Gazebo-based package for underwater intervention and multi-robot simulation. In *OCEANS (Monterey)*.

Mueller, C.A., Fromm, T., Gomez Chavez, A., Koehntopp, D., and Birk, A. (2018). Robust Continuous System Integration for Critical Deep-Sea Robot Operations Using Knowledge-Enabled Simulation in the Loop. *ArXiv*. `https://arxiv.org/abs/1803.02127`.

Schechner, Y. and Karpel, N. (2005). Recovery of Underwater Visibility and Structure by Polarization Analysis. *Journal of Oceanic Engineering*, 30(3), 570–587.

Treibitz, T. and Schechner, Y. (2009). Active Polarization Descattering. *Transactions on Pattern Analysis and Machine Intelligence*, 31(3), 385–399.

Zeng, L. and Dai, Y. (2016). Single Image Dehazing Based on Combining Dark Channel Prior and Scene Radiance Constraint. *Chinese Journal of Electronics*, 25(6), 1114–1120.

Zhang, L., Zhang, L., Mou, X., and Zhang, D. (2011). FSIM: A Feature Similarity Index for Image Quality Assessment. *Transactions on Image Processing*, 20(8), 2378–2386.