EUROGRAPHICS Workshop on Graphics and Cultural Heritage (2018)
R. Sablatnig and M. Wimmer (Editors)

Improving Marker-Based Tracking for Augmented Reality in Underwater Environments

Jan Čejka[1], Marek Žuži[1], Panagiotis Agrafiotis[2], Dimitrios Skarlatos[2], Fabio Bruno[3], and Fotis Liarokapis[1]

[1] Human Computer Interaction Laboratory, Faculty of Informatics, Masaryk University, Czech Republic
[2] Photogrammetric Vision Laboratory, Department of Civil Engineering and Geomatics, Cyprus University of Technology, Cyprus
[3] 3D Research s.r.l. – University of Calabria, Italy

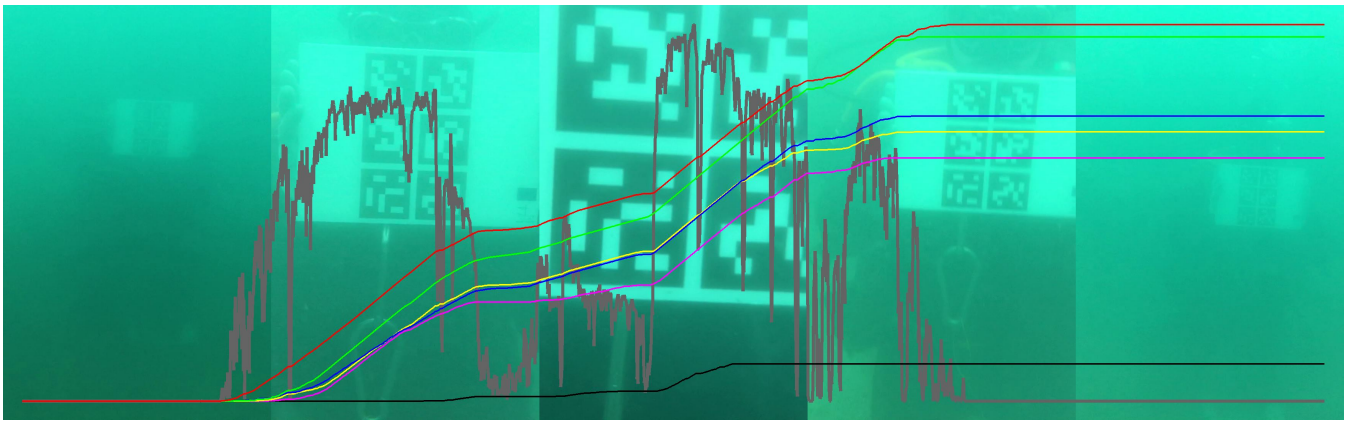


Figure 1: *Number of markers detected by ARUco in video without using any image enhancement algorithm (black), and in videos enhanced by CLAHE (magenta), debluring (yellow), white balancing (blue), and our MBUWWB algorithm (green), at different marker visibility levels (gray). Combination of fast ARUco and our fast algorithm provides results comparable with more robust and much slower AprilTag2 (red).*

Abstract

Underwater sites are a challenging environment for augmented reality. Images taken under water are degraded in several ways, most importantly they are affected by unbalanced colors due to uneven absorption of light in each color channel, blurring and desaturation caused by turbidity, or noise due to the presence of larger organisms, plants, or bubbles. In this paper, we introduce a new method based on white balancing that enhances underwater images to improve the results of detection of markers. We compare this method with several image enhancement methods, and evaluate their performance when applied to the problem of detecting markers under water. Our results show that our method improves the detection in underwater environments while keeping the computation time low.

CCS Concepts

•***Computing methodologies*** → *Mixed / augmented reality; Computer vision;*

1. Introduction

Underwater cultural heritage assets are widely spread into the Mediterranean. Unlike land archaeological sites, however, submerged settlements, ancient ports and other coastal industrial installations, especially shipwrecks, are not accessible to the general public nor all experts, due to their environment and depth. On the other hand, underwater archaeological parks already exist in Mediterranean allowing divers to explore ancient cities. Augmented reality (AR) gives us an opportunity to enhance people's vision of the real world and is the perfect medium for exploiting cultural heritage sites. One of the main technical requirements of achieving effective AR applications is accurate tracking. Although there has been a lot of research in the past few years in vision tracking, it is still almost unexplored in underwater environments and in particular in the open sea.

DOI: 10.2312/gch.20181337

The first time that underwater AR was proposed was in 1999 by Gallagher et al. [Gal99] where the development of a HMD for Navy divers was presented. In 2009, Morales et al. [MKMK09] presented an experimental AR system that provided visual aids to increase commercial divers' capability to detect, perceive, and understand elements in underwater controlled environments. In the same year, researchers developed the AREEF system which allowed people to discover underwater word of corals, fish or fairy-tale wrecks in a swimming pool in a comfortable and safe way [BBM09]. In 2013, the AREEF system was improved in order to be used by more than one person [OBLS13] and also by children [OBS16]. Also in 2013, Brown et al. [BW13] presented a system for using augmented reality under water to help divers with navigation and identification of fish species.

Conditions in sea are different, however. The vision is degraded by several factors, most importantly by turbidity, different attenuation of light in different wave lengths, and a presence of large particles in water. This has a huge impact on a detection and recognition of objects, which is an inherent part of each AR application. To our knowledge, there was only one attempt to adopt AR technology directly in the submerged environment by using underwater tablet. This device was equipped with an underwater positioning and orientation system that guided the diver tourists during the diving session while providing information about the archaeological artifacts spotted during the visit [BLB*16,BLM*16]. However, this solution was based on acoustic sensors and not vision.

The focus of this research is on improving videos taken in sea environments before they are processed by marker detecting algorithms to increase the performance of these algorithms at such places. Although similar tests were already done in laboratory conditions [dSCGF*15], as a part of an evaluation of a single image improving algorithm [AC14, GLW16], or in underwater photogrammetry [SPOP16, ADGS17], to our best knowledge there is no evaluation of real-time image enhancing algorithms focusing on a detection of markers located in underwater marine environments that can be used for AR.

This paper is based on work of Žuži et al. [ŽČB*18], which evaluates off-line algorithms for dehazing. In contrast to it, this paper focuses on real-time image enhancing algorithms. Our main contribution is a new method for real-time improvement of underwater images containing markers for AR, and comparison of its performance with three state-of-the-art methods in four different underwater open sea environments. This comparison is done based on the number of markers detected in improved videos. We demonstrate that applying online image enhancing algorithms improves the performance of marker-based tracking. Our results exhibit a clear improvement of marker-based tracking, which indicates that it can be used in the future for underwater vision tracking.

The rest of the paper is structured as follows. Section 2 presents related work done in improving images and marker-based tracking. Section 3 illustrates the proposed methodology, Section 4 describes the image enhancement algorithms that were tested, and Section 5 shows our method to estimate marker visibility. Section 6 demonstrates our results obtained from tested videos, Section 7 results of a cultural heritage use case, and Section 8 discussions. Finally, Section 9 presents conclusions and future work.

2. Related work

This section presents related work and consists of three parts: improvement of images taken in standard conditions, improvement of images taken under water, and marker-based tracking.

2.1. Improvement of images taken in standard conditions

Improvement of images is one of the most common problems solved when preprocessing images. The most important problems that are solved are removing effects of noise and sharpening of the image, while preserving colors, edges, and overall perception of objects in the image. The most common techniques used for general image improvements are median and bilateral filters [Wei06, Tsi16], and unsharp filters [KCFK17]. Lee and Woo [LW09] present white-balancing technique that improves colors of an image using data from a detected marker, but unlike us, they focus on improving image appearance and not on improving marker detection. Images taken outside of water are often degraded similarly as images taken under water. There are several techniques focusing on improving images damaged by fog or haze, some of these techniques acquire additional data about the scene to ease dehazing. Treibitz et al. [TS09] discusses advantages and disadvantages of using polarization filters to obtain several images of the same scene with different polarization of the light to suppress the effect of fog in final image. Kopf et al. [KNC*08] uses depth information at each pixel to relight the scene or remove the effect of haze.

Moreover, the problem of removing haze from single images without any other additional information available is also becoming more important topic of research. He et al. [HST11] discovers that most regions of haze-free images have at least one color channel (red, green, or blue) that contains very low values. They call this channel a *dark channel*, and proposes a *dark channel prior*, which is a method for dehazing images based on this information. Zhu et al. [ZMS15] describes a similar method that is based on a difference between saturation and value channels of pixels represented in HSV color space. In 2008, Fattal [Fat08] assumes the illumination of objects in the scene and depth of these objects are two statistically uncorrelated functions, uses methods for independent component analysis to separate these functions, and recovers the image without haze. In 2014, Fattal [Fat14] describes another technique that is based on *color lines*, a method for representing colors described by Omer et al. [OW04]. This technique processes small patches in input images with colors that belong to the same color line, and uses properties of this color line to estimate the amount of haze in this patch.

In addition to this, Zhang et al. [ZH14] describes a method that tests several possible solutions, and chooses the best solution for each pixel of input image individually. He also provides an implementation for GPUs that runs real-time. Li et al. [LTT*15] builds on methods that compute the depth of image from a video sequence, and develops a method that simultaneously reconstructs the scene and removes fog in the scene, using the depth cues from scene reconstruction for removing the fog and vice versa. Ancuti et al. [AC14] focuses especially at a detection of SIFT features for matching two images, and proposes a method that enhances contrast in images to recover and match as many features as possible.

In recent years, there has been a lot of attention in using neural networks. Cai et al. [CXJ*16] was inspired by techniques based on dark channel, enhancing contrast, attenuation of colors, and hue disparity, and designed a convolutional neural network (CNN) that consists of several convolving and max-pooling layers to obtain a transmission map for further removal of fog. Ren et al. [RLZ*16] designs a CNN to obtain firstly an assumption of coarse transmission map, and than use this information as an input to another CNN to obtain a precise transmission map. Both these CNN do not depend on any assumptions or prior knowledge of input images, however, they need a substantial amount of training images with known ground truth to train them. This problem is tackled by Li et al. [LGG18], who proposes a neural network for improving underwater images without needing training images with known ground truth.

2.2. Improvement of underwater images

Restoration of underwater images represents a greater challenge than restoration of images taken outside of water. This is due to the fact that underwater images are degraded by much more phenomena than other images, most importantly by turbidity, floating particles, and an uneven absorption of different color channels of light. Ancuti et al. [AAHB12] proposes a system based on a fusion of multiple images that are derived from the input image using common image improving techniques like white-balancing and contrast enhancement. Li et al. [LLZ*14] creates a pipeline of several filters to improve images degraded by uneven illumination, noise, turbidity, and uneven absorption of colors. Chiang et al. [CC12] uses the dark channel prior proposed by He et al. [HST11] to estimate the depth of objects in the input image. This depth is then used to reduce uneven illumination caused by the scattering of artificial light, and to reduce the effect of turbidity in the image. Although the dark channel prior provides very good results in images taken outside of water, it cannot be used for images taken in large depths under water, due to the missing red and green channels.

Some algorithms were designed especially to overcome this issue. Carlevaris-Bianco et al. [CBME10] uses a difference between the maximum in red, green and blue channels to get a coarse initial estimate of the depth in the image. This initial estimate is refined by using a natural image matting, similarly as in a work by He et al. [HST11]. Drews et al. [DNBC16] suggests ignoring the red channel at all, and use only green and blue channels to estimate the depth in the image. Gao et al. [GLW16] presents a bright channel prior, which operates with the original red channel and inverted green and blue channels. These methods try to estimate the depth from the input image. However, the depth can be obtained using other methods. Drews et al. [DNCE15] focuses on processing a video instead of using individual images, and reconstructs the scene from it using methods for scene reconstruction. Similarly to other methods, this depth is used to reduce the effects of turbidity in images. Babaee et al. [BN15] proposes using a sonar in environments with high turbidity to obtain 3D positions of objects in the image. Cho et al. [CSK16] uses a depth precisely measured in a few points in the input image, and uses incremental Gaussian process to estimate the depth in the rest of the image.

2.3. Marker-based tracking

There are different algorithms for marker-based tracking. They use different types of markers to achieve faster detection and recognition of markers, better robustness in uneven lighting conditions, or better differentiation between individual markers when multiple markers are used. Square markers [KB99, GJnSMCMJ14, WO16, Fia05] are well detectable, and their corners are sufficient to compute marker's position. The inner part of markers is used to carry an information to distinguish between different markers. ARToolKit [KB99] uses arbitrary images, making them very versatile and user-friendly. Binary codes are also used, often in a form of a two-dimensional matrix. This form is more robust, especially when it is augmented with Hamming error-correcting codes [KB99, WO16, Fia05] or specifically created dictionaries [GJnSMCMJ14].

Markers in shapes of disks and ellipses are also used [KPS10, NF02]. Instead of using only four points to compute the position of the marker, as it is done in case of square markers, the whole boundary is used to compute the position, making it more precise and more robust to occlusions. On the other hand, spherical shape imposes difficulties when designing robust patters inside the marker. Irregular shapes of markers are also used [BKJ05, BAC*16], allowing to design very precise markers, but usually at the price of increased detection time. Performance of marker-based detectors is usually evaluated in clean view, focusing only on the size and orientation of the marker, occlusions, optics of the camera, and illumination of the marker [TATM13, GJnSMCMJ14, NF02]. Cesar et al. [dSCGF*15] compares marker detectors in a simulated underwater environment in different levels of turbidity, lighting conditions, viewing distances, and viewing angles.

3. Methodology

In this paper, two marker detection libraries were used to detect markers in our videos, ARUco [GJnSMCMJ14] and AprilTag2 [WO16]. These libraries were chosen for several reasons: their source code is available as open source and is frequently updated, they detect markers in real-time, and they are robust to different lighting conditions. We refer to corresponding papers for detailed information about these algorithms. Detectors detects markers in gray-scale images so all images are converted to YUV color space, and then markers are detected in Y channel. This color space was chosen, because it is supported by most mobile devices. An implementation of ARUco that is a part of OpenCV 3.2.0 was employed. For AprilTag2, we used source code available at the site of the project, version 0.9.8. Both libraries are optimized to use multiple threads; additionally, OpenCV is optimized to use SIMD instructions.

Performance of these two marker detection libraries was evaluated in laboratory conditions in a work of Cesar et al. [dSCGF*15]. Their results showed that AprilTag (the predecessor of AprilTag2) provided better results than ARUco in highly turbid water environments, and only a slightly better results in other environments. However, the detection time of AprilTag was approximately three times higher than that of ARUco, which can make the it unusable in real-time AR applications. We used 6 markers from ARUco

DICT_6X6_50 dictionary. To distinguish between markers, this dictionary uses labels with 36 bits of information in a form of a binary matrix of 6 rows and 6 columns, and can correct up to 6 incorrectly detected bits. The markers were printed on a A4 paper, formed in a grid of two rows and three columns. Each marker measured 8 centimeters, and there was 1 centimeter space between each row and column. The detection of markers was evaluated on four set of videos from [SAB*16], see Figure 2. These videos were taken in Mediterranean sea; the locations were chosen by experts in underwater archeology to represent the most typical environments of archaeological sites. These videos were processed off-line on a standard PC using different image enhancing algorithms. Also, the detection of markers was performed off-line on each frame of these processed videos.

4. Tested image enhancing algorithms

Four real-time image enhancing algorithms were tested including: contrast limited adaptive histogram equalization (CLAHE), deblurring (also known as unsharp mask), white balancing, and our newly designed method based on white balancing adapted to detection of markers that we call *marker-based underwater white balancing* – MBUWWB.

Contrast limited adaptive histogram equalization

Contrast limited adaptive histogram equalization (CLAHE) [PAA*87] is a variation of ordinary histogram equalization. Unlike ordinary histogram equalization, it works with a histogram of a small neighbor window around each pixel, and reduces the contrast of output image by clipping the highest values of input image histogram. CLAHE has a single parameter *clip limit*, which influences the amount of values clipped in the histogram. CLAHE was applied to Y channel, leaving U and V channels unchanged. For *clip limit*, we experimented with six values from 1 to 6.

Debluring

Deblur filter (or unsharp mask [KCFK17]) emphasizes high frequencies in input image by subtracting its low frequencies from itself. This filter is defined with the following equation:

$$I_{out} = (1+w)\cdot I_{in} - w\cdot \text{Gaussian}(I_{in},\sigma) \quad (1)$$

where w represents the weight of subtracted low frequencies, and $\text{Gaussian}(I_{in},\sigma)$ is a gaussian filter with standard deviation σ applied to the input image I_{in}. Deblur filter was also applied to Y channel, leaving U and V channels unchanged. We experimented with 16 combinations of values for σ and w: four values 1.0, 2.0, 3.0, and 4.0 for σ, and the same four values for w.

White balancing

White balancing (WB) transforms the colors in input image, so that white objects appeared as white under different illuminations (sun, clouded sky, or others). Although there are many sophisticated methods for white balancing, we used a method presented by Limare et al. [LLM*11]. This method was chosen because of its simplicity, processing speed, and universality.

foreach *color channel* **do**
  compute histogram of this channel;
  $channel_{min} \leftarrow$ *black*-th percentile of values in histogram;
  $channel_{max} \leftarrow$ *white*-th percentile of values in histogram;
  linearly transform all pixel intensities so that
  $channel_{min} = 0$ and $channel_{max} = 255$;
end

Algorithm 1: Pseudocode of white balancing algorithm

The algorithm is described by a pseudocode in Algorithm 1.

With values 0 for *black* and 100 for *white*, the algorithm transforms colors of the input image in a way that the colors of each color channel use the full range of possible values. This is similar to ordinary histogram equalization, but unlike histogram equalization, this transformation is done linearly without equalization. When choosing values higher than 0 for *black* and lower than 100 for *white*, the algorithm ignores the darkest and brightest pixels in the input image, which makes it more robust to noise. It should be noted that colors of restored images may not represent colors of objects properly, due to the simplicity of the algorithm. This is not a problem, however, since the restored image is not presented to the user; it is only processed by marker detectors. The white balancing algorithm is applied to all channels of RGB image, and then we convert the result into YUV space and use the Y channel for marker detection. We experimented with 16 combinations of values for percentiles *black* and *white*: four values 0, 1, 2, and 3 for percentile *black*, and four values 97, 98, 99, and 100 for percentile *white*.

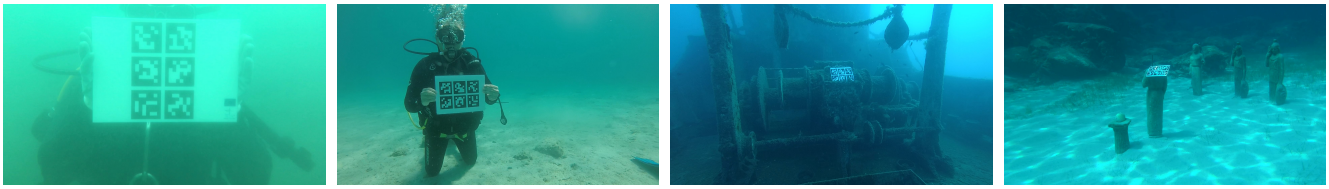
Marker-based underwater white balancing

Marker-based underwater white balancing (MBUWWB) algorithm is our variation of the white balancing algorithm described in the previous section adapted to the problem of marker-based tracking in underwater environments. An intrinsic part of white balancing algorithms is to find colors that are subsequently mapped to the white and the black in the filtered image. White balancing algorithm described in the previous section chooses these colors as percentiles of values in input image histogram.

This behavior was adapted to a marker-based tracking. We assume that the marker is black and white, and instead of computing the histogram of the whole image, only the histogram of the part of the image which contains markers is computed. More precisely, if we detected any markers in the previous frame, we use this part of this previous frame with markers for computation of the histogram. If the previous frame did not contain any marker (or we process the first frame of the video), we compute the histogram of the whole image. This algorithm was applied to all channels of RGB image, and then the result is converted into YUV space, similarly as with ordinary white balancing algorithm. We also experimented with the same 16 combinations of values for percentiles *black* and *white*.

5. Visibility conditions

In this paper the main focus is on the detection of markers in different underwater visibility conditions. These conditions are influ-



Environment 1	**Environment 2**	**Environment 3**	**Environment 4**
High level of turbidity	Moderate level of turbidity	Moderate level of turbidity	Low level of turbidity
Depth of 5 to 6 meters	Depth of 7 to 9 meters	Depth of 20 to 22 meters	Depth of 7 to 9 meters
iPad Pro 9.7-inch tablet	GoPro camera	GARMIN VIRB XE camera	NVIDIA SHIELD tablet
1920 × 1080, MPEG-2	1920 × 1080, MPEG-4	1920 × 1440, MPEG-4	1920 × 1080, MPEG-4
30 fps, 85 seconds	29.97 fps, 31 seconds	24 fps, 160 seconds	30 fps, 81 seconds

Figure 2: *Four different environments used in evaluation.*

enced mainly by turbidity of water, lighting conditions, presence of caustics, presence of small objects floating in the water, and also the size of the markers. All of these effects change the result of marker detection in differently with different impact. To cover all these effects, visibility in image is defined as follows. The computation of visibility is based on the number of detected markers. We compute *marker visibility* of each frame of our videos as a weighted average of the number of markers detected by all marker detectors and all image enhancing techniques. This average is used to obtain a value that is not biased towards any marker detector, any image processor, and any factor influencing conditions under water. Value of *marker visibility* can be expressed as:

$$marker\,visibility = \mathrm{avg}_{det,proc}(w_{proc} * markers(det, proc(frame))) \quad (2)$$

where $\mathrm{avg}_{det,proc}$ is an average took over all marker detectors *det* and image processors *proc*, w_{proc} is a weight of image processor *proc*, and $markers(det, proc(frame))$ is a number of markers detected by detector *det* in frame *frame* processed by image processor *proc*. Weight w_{proc} represents the weight of image processing algorithm. This weight was used, because we experimented with different number of parameter combinations for each image processing algorithm, and we computed the average of the number of detected markers of all parameter combinations. This weight equalizes all image processing algorithms, and is defined as

$$w_{proc} = 1/\#parameter\,combinations(proc) \quad (3)$$

where $\#parameter\,combinations(proc)$ is a number of combinations of parameters of image processor *proc*. This weight is therefore 1 for the results of detecting markers in the original unprocessed video, $1/6$ for CLAHE, and $1/16$ debluring, white balancing, and MBUWWB. Six markers are detected, so the value of *marker visibility* for given frame ranges from 0 (no marker is detected by any detector in the frame no matter what image processor is used) to 6 (all markers are detected by all detectors in the frame no matter what image processor is used). It is worth-mentioning that *marker visibility* was not used directly. 30 visibility levels were defined that represent 30 bins of values and have a range from 0 to 6 with an interval of 0.2. Each frame was assigned with one visibility level based on its *marker visibility*, and declare all frames with the same visibility level as frames with comparable visibility. The number of frames in each visibility level is shown in Figure 3.

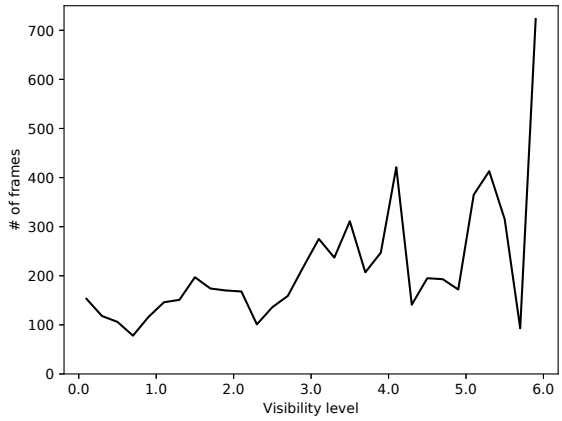


Figure 3: *Number of video frames with given visibility level. Frames without any markers (frames with zero marker visibility) are not considered in this graph.*

6. Results

Results are based on the number of correctly detected and identified markers in video frames enhanced by algorithms described in Section 4. Since no marker was recognized incorrectly if detected during tests, our comparisons do not examine false positives. All image enhancing algorithms and marker detection algorithms were compared to find the best parameters for each image enhancing algorithm, and then compared the results of all algorithms using these parameters. In addition to this, the processing and detection times of each algorithm was also measured.

Parameters of image enhancing algorithms

The average number of markers detected in frames of the same visibility level for each of 30 visibility levels was computed. This average was calculated separately for each marker detector and each combination of parameters of image enhancing algorithm. We present the results in supplementary materials. Instead of using the results for each visibility level, we compute an average of all these

averages. This gives us overall number of detected markers that is not affected by a number of frames with given level of visibility. The results of this overall average for ARUco detector are in Table 1, the results for AprilTag2 detector are in Table 2.

Comparison of algorithms

Our results show that image enhancement improves marker detection performed by ARUco detector. Using the overall average results, ARUco detector found 2.460 markers in frames without any processing, and 2.679, 3.199, 3.133, and 3.429 markers in frames processed by CLAHE, Debluring, WB, and MBUWWB, respectively, when using the best combination of parameters. The results of AprilTag2 detector do not show any significant improvement. Without any image processing, AprilTag2 was able to find 3.278 markers. When the frames were processed by CLAHE, Debluring, WB, and MBUWWB, AprilTag2 was able to find 3.267, 3.295, 3.246, and 3.263 markers. The numbers of AprilTag2 are higher than most of the results of ARUco detector. This indicates that AprilTag2 is a superior marker detector when compared to ARUco (as was similarly observed by Cesar et al. [dSCGF*15]). Despite this, the highest number of detected markers was obtained by using ARUco detector and enhancing images with our MBUWWB algorithm. Based on these findings, we further compared the following combination of algorithms and parameters:

ARUco + Original ARUco detector without using any image enhancing method;
ARUco + CLAHE ARUco detector and CLAHE with clip limit 2;
ARUco + Debluring ARUco detector and debluring with $\sigma = 4.0$ and $w = 4.0$;
ARUco + WB ARUco detector and white balancing with $black = 2$ and $white = 99$;
ARUco + MBUWWB ARUco detector and MBUWWB with $black = 2$ and $white = 99$;
AprilTag2 + Debluring AprilTag2 detector and debluring with $\sigma = 4.0$ and $w = 1.0$ (we use only one combination of image enhancing algorithm and AprilTag2 detector, since other combinations reported similar results);

Figure 4 provides a comparison of selected algorithms using the average number of detected markers per visibility level. It is clearly shown that the last four combinations (*ARUco + Debluring*, *ARUco + WB*, *ARUco + MBUWWB*, and *AprilTag2 + Debluring*) provide the highest average number of detected markers, with *ARUco + MBUWWB* providing the highest numbers at most visibility levels. In this graph, we can also observe an unexpected drop in the number of detected markers of combinations *ARUco + Original* and *ARUco + CLAHE* between visibility levels 4.0 and 5.0. We assume that this behavior may be caused by lower number of evaluated frames in these visibility levels (see Figure 3) or by nature of evaluated marker detecting algorithm that stops finding markers well at this level of visibility.

Processing time

The time spent for detection of markers and for enhancing images was also computed. For the evaluation, a desktop PC with processor Intel Core i5 760, 8 GB of operating memory, and operating system

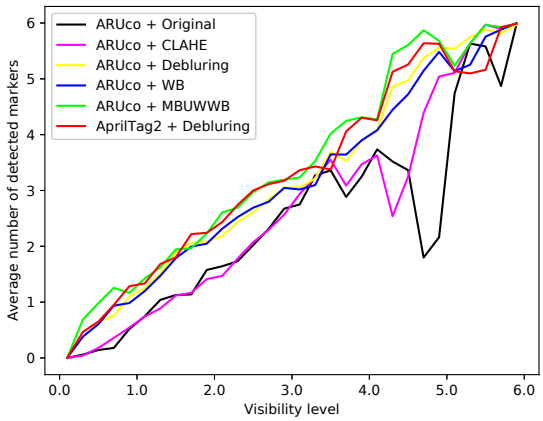


Figure 4: *Comparison of average number of detected markers per visibility level of the best combinations of marker detectors and image enhancing algorithms.*

Windows 10 was used. The measurement was performed using one video from **Environment 2**. To compare the performance of marker detecting algorithms, the time necessary to detect markers in the video without any image enhancement was measured. To compare the performance of image enhancing algorithms, the time required to enhance separate video frames, without performing any detection of markers was computed. As already mentioned, optimized implementations of ARUco and AprilTag2 algorithms were used. ARUco spent approximately 24.3 ms per frame, while AprilTag2 spent 246.9 ms per frame. It must be noted that the implementation of AprilTag2 is not optimized to use SIMD instructions, although we expect the computation time of an optimized version to be still much higher than the computation time of ARUco.

An implementation of CLAHE and debluring using functions available in OpenCV 3.2.0 was used. Since this library is optimized for using SIMD instructions and multiple threads, we implemented a similarly optimized version of white balancing and MBUWWB algorithms. The processing times were as follows: CLAHE 6.8 ms, debluring 28.2 ms, WB 7.0 ms, and MBUWWB 7.0 ms. The times were approximately the same for all parameters of these algorithms. In case of ARUco, the computation time of the whole combination with image pre-processing algorithms was also calculated. The parameters mentioned in the previous section was used again, and used the sum of the image preprocessing time and marker detection time. These are the following: CLAHE 41.8 ms, debluring 77.2 ms, WB 40.6 ms, and MBUWWB 32.1 ms.

7. Augmented reality reconstruction at Baiae

To assess the effectiveness of our results, a pilot underwater testing was performed in the sunken city called Baiae, which is located in Italy. Baiae contains buildings from ancient Rome, which submerged during the last 2000 years due to volcanic activity in this area. The focus of the testing was limited to one building, Villa a Protiro, with a characteristic mosaic in one of the rooms. Using the

Video without enhancing	*avg*	**2.460**							
CLAHE	*clip limit*	1	2	3	4	5	6		
	avg	2.658	**2.679**	2.645	2.583	2.501	2.418		
Debluring	σ	1.0				2.0			
	w	1.0	2.0	3.0	4.0	1.0	2.0	3.0	4.0
	avg	2.528	2.539	2.532	2.504	2.630	2.747	2.821	2.877
	σ	3.0				4.0			
	w	1.0	2.0	3.0	4.0	1.0	2.0	3.0	4.0
	avg	2.681	2.847	2.946	3.053	2.730	2.933	3.082	**3.199**
WB	*black*	0				1			
	white	97	98	99	100	97	98	99	100
	avg	2.996	2.998	3.006	2.975	3.100	3.097	3.105	3.052
	black	2				3			
	white	97	98	99	100	97	98	99	100
	avg	3.128	3.120	**3.133**	3.086	3.106	3.109	3.113	3.080
MBUWWB	*black*	0				1			
	white	97	98	99	100	97	98	99	100
	avg	3.380	3.391	3.401	3.376	3.404	3.412	3.417	3.395
	black	2				3			
	white	97	98	99	100	97	98	99	100
	avg	3.405	3.412	**3.429**	3.388	3.409	3.420	3.420	3.391

Table 1: *Overall average number of markers* avg *detected by ARUco for each parameter of each image enhancing algorithm. This average is taken over averages per visibility level, so they are not affected by the number of frames with given level. The best result for each algorithm is emphasized.*

Video without enhancing	*avg*	**3.278**							
CLAHE	*clip limit*	1	2	3	4	5	6		
	avg	**3.267**	3.141	2.961	2.801	2.667	2.553		
Debluring	σ	1.0				2.0			
	w	1.0	2.0	3.0	4.0	1.0	2.0	3.0	4.0
	avg	3.141	2.980	2.806	2.640	3.200	3.067	2.928	2.768
	σ	3.0				4.0			
	w	1.0	2.0	3.0	4.0	1.0	2.0	3.0	4.0
	avg	3.264	3.205	3.131	3.064	**3.295**	3.276	3.225	3.170
WB	*black*	0				1			
	white	97	98	99	100	97	98	99	100
	avg	3.203	3.208	3.198	**3.246**	3.184	3.182	3.182	3.229
	black	2				3			
	white	97	98	99	100	97	98	99	100
	avg	3.145	3.148	3.148	3.200	3.070	3.079	3.079	3.143
MBUWWB	*black*	0				1			
	white	97	98	99	100	97	98	99	100
	avg	3.228	3.241	3.244	**3.263**	3.217	3.239	3.231	3.248
	black	2				3			
	white	97	98	99	100	97	98	99	100
	avg	3.209	3.217	3.216	3.223	3.194	3.190	3.187	3.198

Table 2: *Overall average number of markers* avg *detected by AprilTag2 for each parameter of each image enhancing algorithm. This average is taken over averages per visibility level, so they are not affected by the number of frames with given level. The best result for each algorithm is emphasized.*

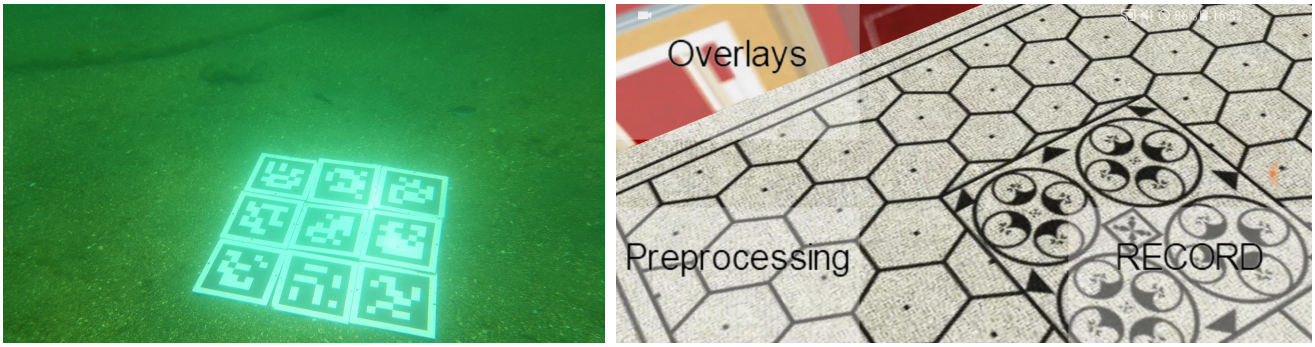


Figure 5: *Experiment performed at Baiae. In the left figure, we see markers placed at the location of the room with mosaic of Villa a Protiro. In the right figure, we see the reconstructed room in AR. With prefiltered image, we were able to detect more markers in more frames.*

Combination	# of frames with detected markers	# total number of detected markers
ARUco + Original	2888	15259
ARUco + CLAHE	3132	18865
ARUco + Debluring	3130	18864
ARUco + WB	3126	19041
ARUco + MBUWWB	3136	19407
AprilTag2 + Debluring	3141	19812

Table 3: *Results of the experiment conducted in Baiae, showing number of frames with at least one marker detected, and number of detected markers.*

improved approach, divers were able to perceive a 3D reconstruction of Villa a Protiro in AR. We used 9 markers from the same ARUco *DICT_6X6_50* dictionary, forming a grid of 3×3 markers. Size of each marker was 19 cm, and the space between markers was approximately 5 cm. A single video was recorded using Samsung S8 with resolution of 1920×1080 and length 141 seconds at 30 fps. Uncompressed images in NV21 format were stored as obtained from the camera. The setup and the application is illustrated in Figure 5.

The results of this experiment are shown in Table 3 and illustrate that AprilTag2 and ARUco with prefiltered input provides more images where at least one marker is detected, and therefore more images where the position of the camera can be calculated. Another interesting point is that ARUco detects the most markers when combined with MBUWWB, which provides us with more data for further processing, e.g., to calculate position of the camera more precisely. AprilTag2 provided slightly better results than ARUco combined with MBUWWB; however, the computation time is much larger, as mentioned in Section 6.

8. Discussion

Results indicate that proper choice of marker detection algorithm, which consists of steps like image preprocessing, thresholding, contour detection, and other, is very important for detecting markers in underwater environments. These environments heavily affect visibility, and thus they also affect the result of marker detection. It is not clear to us how exactly these conditions influence each detection step, but it was shown that the whole detection can be improved by adding another image preprocessing step. The combination of two fast algorithms, ARUco detector and our MBUWWB image enhancing algorithm, provided better results in less time when compared to more robust and much slower AprilTag2 in terms of the number of detected markers. This shows that proper image enhancing algorithm can replace complex preprocessing and thresholding methods in AprilTag2, and still obtain comparable results, as can also be seen in Figure 1.

Processing times showed us that different image enhancing algorithms influence the detection in different ways. Although debluring provided results comparable with our MBUWWB, images sharpened with debluring contain much more contours that must be rejected, which increases the detection time. Our combination of algorithms provided the result in a much smaller computation time than more robust solution. Also, when more markers creating a single multi marker are detected, we can compute a position of this multi marker with higher precision. Our experiment did not include any test to compare this precision, however, because in our experience, it is too difficult to obtain precise location of the marker in underwater environments to have a ground truth to which results could be compared.

Table 1 indicates that in case of the combination of ARUco and debluring algorithm, higher values of σ and w may lead to a higher number of detected markers. We performed additional tests with higher values of σ and w, and found that the highest overall average number of markers 3.550 was reached at $\sigma = 8.0$ and $w = 9.0$, which is comparable with our MBUWWB. However, the computation time also increased, due to larger gaussian kernel size that increases with σ and higher sharpness of features that increases with w. Processing time increased to 52.5 ms (image processing only) and 140.7 ms (image processing and marker detection), which makes it less usable for real-time applications when compared to MBUWWB.

Using more sophisticated algorithms for image enhancement is disputable, due to the time necessary to enhance images before detecting markers. Although some of these algorithms (especially those based on neural networks) provide results in very small time

on PC, the processing power of mobile devices designed for augmented reality is lower, which reduces the number of usable algorithms. The results of white balancing methods indicate an important characteristic of computer vision performed in underwater environments. Luminance channel I, which is used in computer vision algorithms, is computed as a weighted sum of individual color channels R, G, and B:

$$I = w_r R + w_g G + w_b B$$

where w_r, w_g, and w_b are weights of these colors. However, since white balancing methods apply an affine transformation to each color channel individually, the white-balanced image can be obtain by using different weights to the original color channels. This indicates that a simple modification of the weights can improve the performance of computer vision algorithms in underwater environments, which can be done for free as a part of ordinal white balancing performed in digital cameras.

9. Conclusion

In this paper, we focused on the real-time detection of markers in underwater open sea environments that can be used for AR. We developed a method for enhancing images that is specialized to underwater environments and detection of markers. We compared this method with three methods for enhancing images on four sets of videos, each taken in a different environment. The results showed that our method combined with a fast marker detector gives better results than more sophisticated marker detector that runs much slower. In the future, a hybrid approach will be employed based on data generated by the visual tracking techniques illustrated in this paper and data from an acoustic modem. The solution will be integrated with a customized underwater tablet, which will estimate the position of the receiver by computing the distance from at least three fixed beacons placed on the seabed. The incoming positioning data from the various sensors will be finally processed through data fusion and error estimation algorithms.

Acknowledgments

This research was part of the i-MareCulture project (Advanced VR, iMmersive Serious Games and Augmented REality as Tools to Raise Awareness and Access to European Underwater CULTURal heritagE, Digital Heritage) that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 727153.

References

[AAHB12] ANCUTI C., ANCUTI C. O., HABER T., BEKAERT P.: Enhancing underwater images and videos by fusion. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Washington, DC, USA, 2012), IEEE Computer Society, pp. 81–88. 3

[AC14] ANCUTI C., CODRUTA A.: Effective contrast-based dehazing for robust image matching. 1871–1875. 2

[ADGS17] AGRAFIOTIS P., DRAKONAKIS G. I., GEORGOPOULOS A., SKARLATOS D.: The effect of underwater imagery radiometry on 3D reconstruction and orthoimagery. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2/W3* (2017), 25–31. doi:10.5194/isprs-archives-XLII-2-W3-25-2017. 2

[BAC*16] BERGAMASCO F., ALBARELLI A., COSMO L., RODOLÁ E., TORSELLO A.: An accurate and robust artificial marker based on cyclic codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence 38* (2016), 2359–2373. doi:10.1109/TPAMI.2016.2519024. 3

[BBM09] BLUM L., BROLL W., MÜLLER S.: Augmented reality under water. In *SIGGRAPH '09: Posters* (New York, NY, USA, 2009), ACM. doi:10.1145/1599301.1599398. 2

[BKJ05] BENCINA R., KALTENBRUNNER M., JORDA S.: Improved topological fiducial tracking in the reacTIVision system. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops* (2005). doi:10.1109/CVPR.2005.475. 3

[BLB*16] BRUNO F., LAGUDI A., BARBIERI L., MUZZUPAPPA M., RITACCO G., COZZA A., COZZA M., PELUSO R., LUPIA M., CARIO G.: Virtual and augmented reality tools to improve the exploitation of underwater archaeological sites by diver and non-diver tourists. In *Digital Heritage. Progress in Cultural Heritage: Documentation, Preservation, and Protection: 6th International Conference, EuroMed 2016* (Cham, 2016), Springer International Publishing, pp. 269–280. doi:10.1007/978-3-319-48496-9_22. 2

[BLM*16] BRUNO F., LAGUDI A., MUZZUPAPPA M., LUPIA M., CARIO G., BARBIERI L., PASSARO S., SAGGIOMO R.: Project VISAS: Virtual and augmented exploitation of submerged archaeological site – overview and first results. In *Marine Technology Society Journal* (2016), vol. 50, pp. 119–129. 2

[BN15] BABAEE M., NEGAHDARIPOUR S.: Improved range estimation and underwater image enhancement under turbidity by opti-acoustic stereo imaging. In *OCEANS 2015* (2015), pp. 1–7. doi:10.1109/OCEANS-Genova.2015.7271611. 3

[BW13] BROWN H. C., WANG H.: Underwater augmented reality: Navigation and identification. In *2013 OCEANS - San Diego* (2013), pp. 1–5. 2

[CBME10] CARLEVARIS-BIANCO N., MOHAN A., EUSTICE R. M.: Initial results in underwater single image dehazing. In *OCEANS 2010* (2010), pp. 1–8. doi:10.1109/OCEANS.2010.5664428. 3

[CC12] CHIANG J. Y., CHEN Y.-C.: Underwater image enhancement by wavelength compensation and dehazing. *IEEE Transactions on Image Processing 21* (2012), 1756–1769. doi:10.1109/TIP.2011.2179666. 3

[CSK16] CHO Y., SHIN Y.-S., KIM A.: Online depth estimation and application to underwater image dehazing. In *OCEANS 2016 MTS/IEEE Monterey* (2016), pp. 1–7. doi:10.1109/OCEANS.2016.7761109. 3

[CXJ*16] CAI B., XU X., JIA K., QING C., TAO D.: Dehazenet: An end-to-end system for single image haze removal. *IEEE Transactions on Image Processing 25* (2016), 5187–5198. doi:10.1109/TIP.2016.2598681. 3

[DNBC16] DREWS P. L., NASCIMENTO E. R., BOTELHO S. S., CAMPOS M. F. M.: Underwater depth estimation and image restoration based on single images. *IEEE Computer Graphics and Applications 36* (2016), 24–35. doi:10.1109/MCG.2016.26. 3

[DNCE15] DREWS P., NASCIMENTO E. R., CAMPOS M. F. M., ELFES A.: Automatic restoration of underwater monocular sequences of images. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2015), pp. 1058–1064. doi:10.1109/IROS.2015.7353501. 3

[dSCGF*15] DOS SANTOS CESAR D. B., GAUDIG C., FRITSCHE M., DOS REIS M. A., KIRCHNER F.: An evaluation of artificial fiducial markers in underwater environments. In *OCEANS 2015* (2015), pp. 1–6. doi:10.1109/OCEANS-Genova.2015.7271491. 2, 3, 6

[Fat08] FATTAL R.: Single image dehazing. *ACM Trans. Graph. 27*, 3 (2008). doi:10.1145/1360612.1360671. 2

[Fat14] FATTAL R.: Dehazing using color-lines. *ACM Trans. Graph. 34* (2014). doi:10.1145/2651362. 2

[Fia05] FIALA M.: Artag, a fiducial marker system using digital techniques. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (Washington, DC, USA, 2005), IEEE Computer Society, pp. 590–596. doi:10.1109/CVPR.2005.74. 3

[Gal99] GALLAGHER D. G.: Development of miniature, head-mounted, virtual image displays for navy divers. In *OCEANS '99* (1999), vol. 3, pp. 1098–1104. doi:10.1109/OCEANS.1999.800143. 2

[GJnSMCMJ14] GARRIDO-JURADO S., NOZ SALINAS R. M., MADRID-CUEVAS F. J., MARÍN-JIMÉNEZ M. J.: Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition 47* (2014), 2280–2292. doi:http://dx.doi.org/10.1016/j.patcog.2014.01.005. 3

[GLW16] GAO Y., LI H., WEN S.: Restoration and enhancement of underwater images based on bright channel prior. 1–15. 2, 3

[HST11] HE K., SUN J., TANG X.: Single image haze removal using dark channel prior. *IEEE Trans. Pattern Anal. Mach. Intell. 33*, 12 (2011), 2341–2353. doi:10.1109/TPAMI.2010.168. 2, 3

[KB99] KATO H., BILLINGHURST M.: Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *Augmented Reality, 1999. (IWAR '99) Proceedings. 2nd IEEE and ACM International Workshop on* (1999), pp. 85–94. doi:10.1109/IWAR.1999.803809. 3

[KCFK17] KRASULA L., CALLET P. L., FLIEGEL K., KLÍMA M.: Quality assessment of sharpened images: Challenges, methodology, and objective metrics. *IEEE Transactions on Image Processing 26* (2017), 1496–1508. doi:10.1109/TIP.2017.2651374. 2, 4

[KNC*08] KOPF J., NEUBERT B., CHEN B., COHEN M., COHEN-OR D., DEUSSEN O., UYTTENDAELE M., LISCHINSKI D.: Deep photo: Model-based photograph enhancement and viewing. *ACM Trans. Graph. 27*, 5 (2008). doi:10.1145/1409060.1409069. 2

[KPS10] KÖHLER J., PAGANI A., STRICKER D.: Robust detection and identification of partially occluded circular markers. In *VISAPP 2010 - Proceedings of the Fifth International Conference on Computer Vision Theory and Applications* (2010), pp. 387–392. 3

[LGG18] LI C., GUO J., GUO C.: Emerging from water: Underwater image color correction based on weakly supervised color transfer. *IEEE Signal Processing Letters 25* (March 2018), 323–327. doi:10.1109/LSP.2018.2792050. 3

[LLM*11] LIMARE N., LISANI J. L., MOREL J., PETRO A. B., SBERT C.: Simplest color balance. *IPOL Journal 1* (2011). doi:10.5201/ipol.2011.llmps-scb. 4

[LLZ*14] LI Y., LU H., ZHANG L., LI J., SERIKAWA S.: Real-time visualization system for deep-sea surveying. 1–10. 3

[LTT*15] LI Z., TAN P., TAN R. T., ZOU D., ZHOU S. Z., CHEONG L.-F.: Simultaneous video defogging and stereo reconstruction. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 4988–4997. doi:10.1109/CVPR.2015.7299133. 2

[LW09] LEE J., WOO W.: Real-time color correction for marker-based augmented reality applications. In *International Workshop on Ubiquitous Virtual Reality 2009* (2009), pp. 32–35. 2

[MKMK09] MORALES R., KEITLER P., MAIER P., KLINKER G.: An underwater augmented reality system for commercial diving operations. In *Underwater Intervention Conference 2011* (2009), vol. 1, pp. 1–8. 2

[NF02] NAIMARK L., FOXLIN E.: Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In *Proceedings of International Symposium on Mixed and Augmented Reality* (2002), pp. 27–36. doi:10.1109/ISMAR.2002.1115065. 3

[OBLS13] OPPERMANN L., BLUM L., LEE J.-Y., SEO J.-H.: Areef multi-player underwater augmented reality experience. In *2013 IEEE International Games Innovation Conference (IGIC)* (2013), pp. 199–202. doi:10.1109/IGIC.2013.6659137. 2

[OBS16] OPPERMANN L., BLUM L., SHEKOW M.: Playing on areef: Evaluation of an underwater augmented reality game for kids. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services* (New York, NY, USA, 2016), ACM, pp. 330–340. doi:10.1145/2935334.2935368. 2

[OW04] OMER I., WERMAN M.: Color lines: Image specific color representation. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)* (2004), vol. 2, pp. 946–953. doi:10.1109/CVPR.2004.1315267. 2

[PAA*87] PIZER S. M., AMBURN E. P., AUSTIN J. D., CROMARTIE R., GESELOWITZ A., GREER T., ROMENY B. T. H., ZIMMERMAN J. B.: Adaptive histogram equalization and its variations. *Computer Vision Graph. Image Processing 39* (1987), 355–368. doi:10.1016/S0734-189X(87)80186-X. 4

[RLZ*16] REN W., LIU S., ZHANG H., PAN J., CAO X., YANG M.-H.: Single image dehazing via multi-scale convolutional neural networks. In *ECCV 2016* (2016), Springer International Publishing, pp. 154–169. doi:10.1007/978-3-319-46475-6_10. 3

[SAB*16] SKARLATOS D., AGRAFIOTIS P., BALOGH T., BRUNO F., CASTRO F., PETRIAGGI B. D., DEMESTICHA S., DOULAMIS A., DRAP P., GEORGOPOULOS A., KIKILLOS F., KYRIAKIDIS P., LIAROKAPIS F., POULLIS C., RIZVIC S.: Project imareculture: Advanced vr, immersive serious games and augmented reality as tools to raise awareness and access to european underwater cultural heritage. In *Digital Heritage* (2016), Springer International Publishing, pp. 805–813. 4

[SPOP16] SARAKINOU I., PAPADIMITRIOU K., OLGA G., PATIAS P.: Underwater 3D modeling: Image enhancement and point cloud filtering. 441–447. 2

[TATM13] TOYOURA M., ARUGA H., TURK M., MAO X.: Detecting markers in blurred and defocused images. In *2013 International Conference on Cyberworlds* (2013), pp. 183–190. doi:10.1109/CW.2013.58. 3

[TS09] TREIBITZ T., SCHECHNER Y. Y.: Polarization: Beneficial for visibility enhancement? In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2009), pp. 525–532. doi:10.1109/CVPRW.2009.5206551. 2

[Tsi16] TSIRIKOLIAS K. D.: Low level image processing and analysis using radius filters. *Digital Signal Processing 50* (2016), 72–83. doi:https://doi.org/10.1016/j.dsp.2015.12.001. 2

[Wei06] WEISS B.: Fast median and bilateral filtering. *ACM Trans. Graph. 25*, 3 (2006), 519–526. doi:10.1145/1141911.1141918. 2

[WO16] WANG J., OLSON E.: AprilTag 2: Efficient and robust fiducial detection. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (October 2016). 3

[ŽČB*18] ŽUŽI M., ČEJKA J., BRUNO F., SKARLATOS D., LIAROKAPIS F.: Impact of dehazing on underwater marker detection for augmented reality. *Frontiers in Robotics and AI 5* (2018), 1–13. doi:10.3389/frobt.2018.00092. 2

[ZH14] ZHANG J., HU S.: A gpu-accelerated real-time single image dehazing method using pixel-level optimal de-hazing criterion. *Journal of Real-Time Image Processing 9* (2014), 661–672. doi:10.1007/s11554-012-0244-y. 2

[ZMS15] ZHU Q., MAI J., SHAO L.: A fast single image haze removal algorithm using color attenuation prior. *IEEE Transactions on Image Processing 24* (2015), 3522–3533. doi:10.1109/TIP.2015.2446191. 2