Visual Cryptography for Detecting Hidden Targets by Small-Scale Robots

Danilo Avola[1], Luigi Cinque[2], Gian Luca Foresti[1], and Daniele Pannone[2(✉)]

[1] Department of Mathematics, Computer Science and Physics, University of Udine, Via delle Scienze 206, 33100 Udine, Italy
{danilo.avola,gianluca.foresti}@uniud.it
[2] Department of Computer Science, Sapienza University, Via Salaria 113, 00198 Rome, Italy
{cinque,pannone}@di.uniroma1.it

Abstract. The last few years have seen a growing use of robots to replace humans in dangerous activities, such as inspections, border control, and military operations. In some application areas, as the latter, there is the need to hide strategic information, such as acquired data or relevant positions. This paper presents a vision based system to find encrypted targets in unknown environments by using small-scale robots and visual cryptography. The robots acquire a scene by a standard RGB camera and use a visual cryptography based technique to encrypt the data. The latter is subsequently sent to a server whose purpose is to decrypt and analyse it for searching target objects or tactic positions. To show the effectiveness of the proposed system, the experiments were performed by using two robots, i.e., a small-scale rover in indoor environments and a small-scale Unmanned Aerial Vehicle (UAV) in outdoor environments. Since the current literature does not contain other approaches comparable with that we propose, the obtained remarkable results and the proposed method can be considered as baseline in the area of encrypted target search by small-scale robots.

Keywords: Visual cryptography · Encrypted target
Shares generation · Target recognition · Rover · UAV · RGB camera
SLAM

1 Introduction

Over the last decade, many efforts have been made to use robots in place of humans in a wide range of complex and dangerous activities, such as search and rescue operations [1,2], land monitoring [3–5], and military missions [6,7]. In particular, the use of small-scale robots has been progressively promoted due to the following aspects: cost reduction, risk reduction for humans, and failure reduction caused by human factor (e.g., carelessness, inaccuracy, tiredness). Some significant examples are the defusing of Improvised Explosive Devices (IEDs) placed

M. De Marsico et al. (Eds.): ICPRAM 2018, LNCS 11351, pp. 186–201, 2019.
https://doi.org/10.1007/978-3-030-05499-1_10

on the ground and the constant monitoring of wide areas at low-altitudes. In these contexts, the use of rovers and UAVs, respectively, equipped with a vision based system can be efficiently adopted also optimizing the missions in terms of performance, speed, and security. Other examples are tasks as object recognition in outdoor environments and change detection in indoor environments, where small-scale robots are used to support moving video-surveillance systems to automatically detect novelties in the acquired video streams.

Often, especially in military field, the protection from intruders of data acquired during the exploration of areas of interest is a crucial task. This is due to the fact that the acquired data may contain sensible information, such as faces of persons, images of restricted areas, or strategic targets. To prevent the leak or the steal of information from images, in this paper a client-server based system to exploit the visual cryptography technique for encrypting acquired visual data is presented. In detail, the first step consists in using visual cryptography to generate two shares from a target image. Following a public key cryptography approach, only one share, i.e., the private key, is stored in the server. Later, the small-scale robot, used to explore the area of interest, captures the scene by an RGB camera and hides the data contained in it by using the same visual cryptography algorithm. Finally, only a share, i.e., the public key, is sent to the server and made available for subsequent decryption processes.

This paper improves and expands the work presented in [8] by introducing the following extensions:

- In addition to the small-scale rover, also a small-scale UAV is used to enrich the tests, thus confirming the goodness of the previous results;
- In addition to the single indoor environment, other challenging indoor environments (for the rover) and some challenging outdoor environments (for the UAV) are used to stress the ability of the proposed method;
- In addition to the previously adopted target objects, also areas of interest are used to test the proposed visual cryptography algorithm. In particular, parts of acquired images are used as invisible markers to identify the areas.

Like for the system presented in [8], the small-scale rover exhaustively explores the area of interest by a Simultaneous Localization and Mapping (SLAM) algorithm, while the small-scale UAV is manually piloted. This last choice is due to the fact that the implementation of a SLAM algorithm for a UAV introduces additional complex issues that, currently, are not the focus of the present paper. Notice that, as for the work presented in [8], the SLAM algorithm for the small-scale rover is inherited by the method reported in [9].

The rest of the paper is structured as follows. In Sect. 2, a brief overview about the visual cryptography is discussed. In Sect. 3, the system architecture and the visual cryptography algorithm are presented. In Sect. 4, the experimental results are shown. Finally, Sect. 5 concludes the paper.

2 Visual Cryptography Overview

As well known, the term visual cryptography is referred to a family of techniques used to encrypt an image by splitting it into n images called shares. The latter do not allow to distinguish any information about the original image unless they are combined together. This means that if only one share is available, the source data is inaccessible. In our knowledge, the only work in literature that combines small-scale robots, visual cryptography, and a SLAM algorithm (at least for the small-scale rover) is that presented in [8]. In particular, in that work, the authors proposed a client-server rover based system to search encrypted objects in unknown environments.

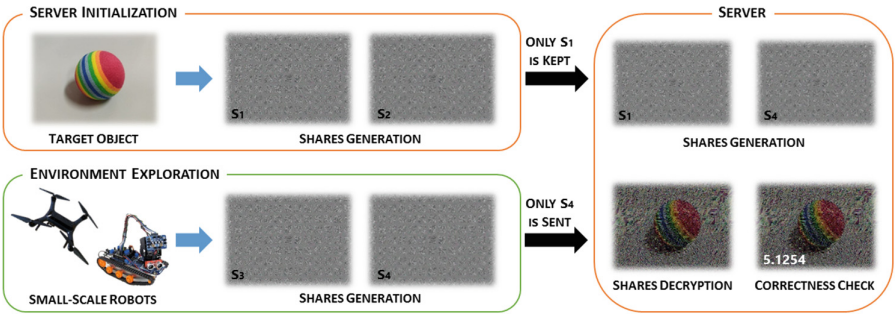


Fig. 1. System architecture. The server initialization stage shows the storing of the private key. The environment exploration stage shows the generation of the public key. Finally, the server stage shows the decryption of the data.

The visual cryptography technique was introduced by Naor and Shamir [10]. The method they proposed has never been heavily modified, but some improvements and variants can be found in the current literature [11]. Authors in [12], for example, proposed an improvement for the perfect black visual cryptography scheme, thus allowing to achieve real-time performance for the decryption step. Another example is reported in [13], where the authors proposed a verifiable multi-toned visual cryptography scheme to securely transmit confidential images (e.g., medical, forensic) on the web. Since, in visual cryptography, the image resulting from the algorithm is decrypted by the human sight, some techniques to enhance the quality of the decrypted images have been also proposed. In particular, the most popular technique is the dithering (or halftoning) [14,15], which allows to create halftone images. Some works that present a visual cryptography technique for halftone images are reported in [16–18]. In the state-of-the-art, works that combine several approaches to improve the ciphering performance [19,20] or to integrate the visual cryptography with traditional protection schemes to enhance them [21–23] can be also found.

Concerning the environment exploration, the present literature is based on the SLAM approaches [24–26]. The aim of these approaches is to maximize the

area coverage during the environment exploration and, at the same time, to make the robot conscious of its absolute position within it. SLAM approaches can be used with several sensors, such as depth/time of flight cameras [27,28], thermal cameras [29], or a fusion of them [30,31]. In addition, these approaches can be used for different tasks, such as mosaicking generation [32], pipe rehabilitation [33], environment mapping [34], and others.

3 Architecture and Visual Cryptography Algorithm

In this section, the system architecture and the visual cryptography algorithm are described. Initially, the client-server approach is explained, then the steps required to encrypt and decrypt the images are reported.

3.1 System Architecture

In Fig. 1, the architecture of the proposed client-server system is shown. The small-scale rover and the small-scale UAV are considered as the client side, instead a standard workstation is considered the server side. Before starting the environment exploration, the server must be initialized. This is done by storing in it an encrypted target image T. This image can represent a specific object or a location of interest that requires to be hidden. By applying the visual cryptography algorithm on T, two shares, i.e., S_1 and S_2, are generated. The adopted public key cryptography approach is designed to store only a share, i.e., S_1, on the server. The latter is defined as the private key of the target image, while the share S_2 is discarded. The environment can be automatically explored with a small-scale rover driven by a SLAM algorithm, or manually explored by piloting a small-scale UAV. During the exploration, the used robot acquires the scene with a standard RGB camera. On the acquired images, the visual cryptography algorithm is applied to generate, once again, two shares, i.e., S_3 and S_4. While S_3 is discarded, S_4 is sent to the server and defined as public key. The latter is used in conjunction with S_1 to decrypt the target image T.

The advantage of using shares instead of clear images is that even if an intruder makes a physical attack (e.g., clients or servers are stolen) or a digital attack (e.g., a video stream is sniffed), the original information cannot be recovered. Moreover, the encryption of objects or locations of interest can be used to generate invisible markers. This is due to the fact that a target image can be decrypted only by using the correct shares. This means that when an image is decrypted, the robot is in a specific position that represents the target spot within the environment.

3.2 Visual Cryptography Algorithm

In this section, the visual cryptography algorithm is explained. For the encrypting and decrypting stages, the approach reported in [18] is used. It consists of

several steps to create the shares. The first is the application of a dithering algorithm to the original image I. The dither is a form of noise intentionally applied to reduce the quantization error. As a result, I is converted into an approximate binary image so that the encryption and decryption processes are easier and the decrypted image has a good quality. In the current literature a wide range of dithering algorithms is available:

- **Average Dithering** [35]**:** is one of the simplest techniques. It consists in calculating the middle tone of each area and in assigning this value to that portion of image;
- **Floyd-Steinberg** [36]**:** is still the most used. It consists in diffusing the quantization error to the near pixels of each pixel of the image;
- **Average Ordered Dithering** [37]**:** is similar to average dithering, but it generates cross-hatch patterns;
- **Halftone Dithering** [36]**:** looks similar to newspaper halftone and produces clusters of pixel areas;
- **Jarvis Dithering** [38]**:** is similar to Floyd-Steinberg, but it distributes the quantization error farther than it, increasing computational cost and time.

Due to its easiness of implementation and its good quality results, in the present approach, as dithering algorithm, the Floyd-Steinberg was chosen. This algorithm diffuses the quantization error to the neighbour pixels as follows:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & p & \frac{7}{16} \\ \frac{3}{16} & \frac{5}{16} & \frac{1}{16} \end{bmatrix} \tag{1}$$

where, the pixel p is the current pixel examined during the execution of the algorithm. Considering that I is scanned from left to right and from top to bottom, the pixels are quantized only once. Since the proposed system uses colour images, the chosen dithering algorithm is applied to each channel of I, thus obtaining three dithered images. In Fig. 2, the result of this step is shown.

(a) **(b)** **(c)**

Fig. 2. Dithered images for channels: (a) cyano, (b) magenta, and (c) yellow. (Color figure online)

Fig. 3. Sharing and stacking combination in grayscale images. In both pictures, (a) and (b), the first column is the original pixel (i.e., black and white, respectively), the second and third columns are the *share 1* and *share 2*, respectively. Finally, the last column is the stacked shares.

Algorithm 1. Algorithm generating shares from a grayscale image.

```
1: procedure SharesfromGray(I)
2:     Transform I to halftone image H
3:     for each pixel in H do
4:         Choose randomly a share among those in
5:             Figure 3
6:     end for
7: end procedure
```

Algorithm 2. Algorithm generating shares from a color image.

```
1: procedure SharesfromColor(colorImage)
2:     Transform colorImage in three halftone images C, M and Y
3:     for each pixel p_{i,j} in C,M and Y do
4:         According to Algorithm 1, create
5:             C1_{i,j}, C2_{i,j}, M1_{i,j}, M2_{i,j}, Y1_{i,j}, Y2_{i,j}
6:         Combine C1_{i,j}, M1_{i,j} and Y1_{i,j} for the corresponding block of Share 1
7:         Combine C2_{i,j}, M2_{i,j} and Y2_{i,j} for the corresponding block of Share 2
8:     end for
9:     After stacking the two Shares, the original image can be decrypted.
10: end procedure
```

After the generation of the dithered images, also the shares can be created. Since the latter are generated starting from grayscale images, the shares generation algorithm can be defined as follows:

1. I is tranformed into a black and white halftone image H;
2. For each pixel in the halftone image, a random combination is chosen among those depicted in Fig. 3;
3. Repeat step 2 until every pixel in H is decomposed.

The pseudo-code is reported in Algorithm 1. To generate the shares for colour images, the third method presented in [18] was used. The choice fell on this method since it requires only two shares to encrypt/decrypt a colour image, in addition, it does not sacrifice too much image contrast in the resulting image. The method works as follows. First, a dithered image for each channel of I is created. Assuming that we are using the YCMK profile, we obtain a dithered

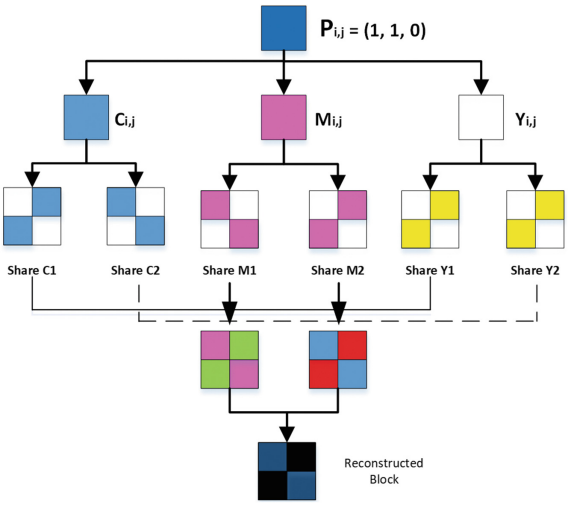


Fig. 4. Decomposition and reconstruction of colour pixel [8].

image for *Cyan* (C), *Magenta* (M), and *Yellow* (Y) channels. Subsequently, for each halftone image the Algorithm 1 is used to generate six 2×2 sharing images, called *C1, C2, M1, M2, Y1*, and *Y2*. Each of these shares is composed by two white pixels and two colour pixels. To generate the coloured *share 1*, *C1, M1*, and *Y1* are combined together, while for generating the coloured *share 2*, *C2, M2*, and *Y2* are combined. The colour intensity of the share blocks is a value between 0 and 1, where 0 means the absence of that colour and 1 means full intensity. So, for a pixel $p_{i,j}$ the colour intensity for each channel is defined as (I_C, I_M, I_Y). For each block generated with this method, we have that the colour intensity is $(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$, while after stacking *shares 1* and *shares 2*, the range of colour intensity is between $(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$ and $(1, 1, 1)$. As for the grayscale algorithm, the decryption step simply consists in overlapping the two shares, thus obtaining the decrypted image I_{dec}. In Algorithm 2 the pseudo-code of the method is shown, while in Fig. 4 a representation of the algorithm is depicted.

Since the images acquired by the robot may not be acquired at the same distance, position, and angulation of I, and since the pixels composing the two shares must be almost perfectly aligned to perform the decryption, a morphing procedure is applied on S_4. This allows to optimize, in some cases, the alignment of the two shares [39,40]. Considering that with the shares a feature-based (e.g., by using keypoints and homography) alignment cannot be performed due to their random pixel arrangement, we have defined eight standard transformations to apply. In Fig. 5, these transformations are shown.

After the generation of I_{dec}, a check to verify the validity of the decrypted image is performed. In particular, the difference between the standard deviation of I_{dec} and its smoothed copy $Smooth_{I_{dec}}$ is computed. The smoothing operation is performed by using a median filter with kernel size of 3×3. Formally:

$$Correctness = I_{dec} - Smooth_{I_{dec}} \tag{2}$$

If I_{dec} is a valid decrypted image, we have that the standard deviation between it and its smoothed copy is low (e.g., less than 10), otherwise higher values are obtained.

3.3 SLAM Algorithm

This section briefly reports the SLAM algorithm, previously presented in [9], used by the rover to exhaustively analyse an unknown environment. Notice that, the used rover is composed by two main components. The first is the base, which is composed in turn by the tracks, while the second is the upper part, that contains the RGB sensor and the ultrasonic sensor. Between the base and the upper part a servomotor is mounted, in a way such that the two parts can be moved independently from each other.

The algorithm considers the environment to be explored as a two-dimensional Cartesian plane $C(X, Y)$, composed by points of the form $c(x, y)$ reachable by the rover. At each c, the rover examines the environment by rotating the base of $0°, 90°, 180°$, and $270°$ with respect to the x axis. Once the rover has assumed a position, it starts moving the upper part at $60°, 90°$, and $120°$ with respect to its local coordinates, thus acquiring an image for each of these angles. If T is not found in these images, the rover checks with the proximity sensor the next point c in which it can move. Once obtained the distance d with the proximity sensor, the robot checks if it is less than the distance $d_{threshold}$, which represents the maximum distance within which the sensor detects an obstacle. The threshold depends on the size of the robot, the size of the object to be searched, and the resolution of the images acquired by the RGB sensor. If $d < d_{threshold}$, it means that there is an obstacle and the rover cannot move to that point. Otherwise, the rover moves to the new point c' and repeat the steps.

With respect to the work presented in [8], an improvement that has been made to the SLAM algorithm is the addition of the tilt movement of the RGB sensor. This allows enhancing the acquisition of flat objects, such as credit cards, books, and so on. The tilt movement is performed by moving the camera of $30°, 60°$, and $75°$ with respect to the camera starting position.

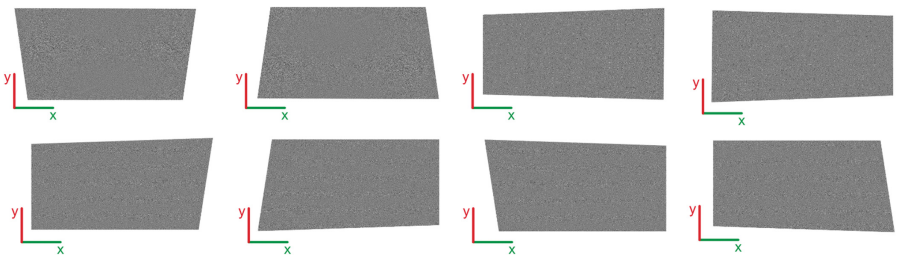


Fig. 5. Transformations applied to S_4 for optimizing the shares overlap.

(a) (b)

Fig. 6. The used robots: (a) the small-scale rover, (b) the small-scale UAV.

4 Experimental Results

In this section, the experimental tests are presented. In detail, we first report the experiments performed with the small-scale rover (Fig. 6a), then the experiments performed with the small-scale UAV (Fig. 6b). Considering that, currently, there are no datasets for this kind of systems/methods, in all the experiments our acquisitions are used. The experiments were performed in controlled conditions, with unnoticeable changes of illumination, and without moving objects. The experiments with the small-scale rover were performed in three indoor environments, while the experiments with the small-scale UAV were performed in a wide outdoor environment. The communication between robots and server was made by a direct Wi-Fi connection, to reduce the delay introduced by sending the network packets.

4.1 Experiments with the Small-Scale Rover

In Fig. 6a, the small-scale rover used in three indoor environments is shown. It is composed by an Arduino UNO micro-controller, which handles both the servomotors and the ultrasonic sensor used by the SLAM algorithm, and by a Raspberry Pi 2 model B, which handles the camera/video stream and the communications with the server. Despite the used rover is able to perform all the required tasks (i.e., SLAM algorithm and sending the share to the server), its low computational power affects the time needed to explore the environment. The first set of experiments was performed by running the entire system on-board. After capturing the frame, the Raspberry Pi 2 proceeded with the pipeline described in Sect. 3, and it took about 20 s for each frame for completing the entire pipeline. To overcome this problem, the client-server approach was adopted for improving the performance.

In Fig. 7, the used indoor environments are shown. In detail, we chosen two challenging environments (Fig. 7a and b) and one easy environment (Fig. 7c) to test exhaustively both the SLAM algorithm and the visual cryptography pipeline. The challenging environments are rooms containing desks and seats,

while the easy environment is a hallway. For each environment, the rover starts the recognition always from the same starting point. To stress the system, we used both clear (i.e., just placed on the ground) and covered (i.e., underneath the desks or the seats) objects. A total of 20 objects (reported in Table 1) with high variability of colours and sizes was used.

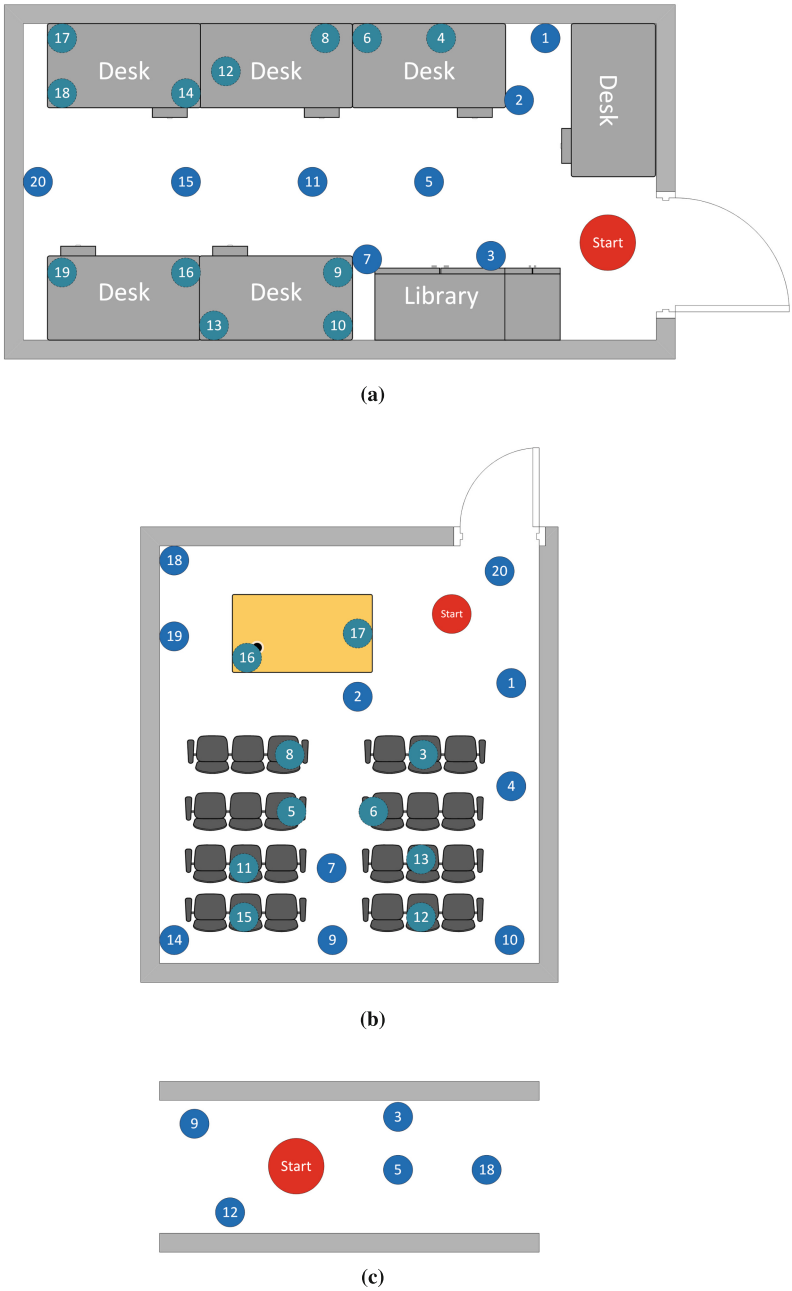
Table 1. List of objects used during the experiments [8].

Object number	Object type	Object type	Object type
Object 1	Ball	Object 11	Cup
Object 2	Toy Gun	Object 12	Coffe Machine
Object 3	USB Keyboard	Object 13	Wallet
Object 4	Pen	Object 14	Plastic Bottle
Object 5	Calculator	Object 15	Monitor
Object 6	Pencil	Object 16	Toy Robot
Object 7	Paperweight	Object 17	DVD Case
Object 8	Credit card	Object 18	Small Box
Object 9	Sponge	Object 19	Big Box
Object 10	USB mouse	Object 20	Book

Concerning the first challenging environment (Fig. 7a), the confusion matrix is reported in Table 2. In this environment, we have noticed that a good correctness value is in the range of $[4, 6]$, while a wrong decrypted image has a correctness value between $[19, 25]$. As it is possible to see, the proposed system works generally well, but due to their characteristics the decryption fails for the credit card and the cup. In detail, the decryption fails because even if we apply the transformation shown in Fig. 5, it may be not sufficient to correctly align the shares. The sponge, differently from the experiments presented in [8], was correctly decrypted thanks to the tilt movement of the RGB camera implemented in this new version of the SLAM algorithm.

Regarding the second environment, the correctness values are reported in Table 3. These values slightly differ from the ones obtained in the first indoor environment due to the different illumination conditions. In this case, the values obtained for good decrypted images are in the range of $[7, 9]$, while the values corresponding to bad decrypted images are in the range of $[27, 31]$. As for the first environment, the credit card is hard to decrypt, due to its flat shape that makes difficult to acquire correctly the object and then to generate a good share.

Finally, in Table 4 the correctness values of the third indoor environment are shown. As for the previous environments, we have low values for a correct decrypted image and high values otherwise. In detail, correct values are in the range of $[3, 5]$, while high values are in the range of $[15, 18]$. Since this is an easy environment, we used only a subset of the objects, and to make the experiments



(a)

(b)

(c)

Fig. 7. Environments used for indoor experiments: (a) and (b) challenging environments, (c) an easy (c) environment. Each environment was chosen to test the system in different conditions. The dark blue circles are the objects placed in clear, while the dashed green circles represent the covered objects. (Color figure online)

Table 2. Confusion matrix of the first indoor environment [8].

	Object 1	Object 2	Object 3	Object 4	Object 5	Object 6	Object 7	Object 8	Object 9	Object 10	Object 11	Object 12	Object 13	Object 14	Object 15	Object 16	Object 17	Object 18	Object 19	Object 20
Object 1	4.9792	24.2623	21.1189	21.6967	24.7812	19.2538	24.8378	20.1352	23.0027	22.5186	23.0507	21.1661	22.7217	23.8669	19.1155	19.5032	24.8488	22.9081	20.3874	21.4209
Object 2	19.7321	4.5369	20.5471	20.9900	19.9134	21.0880	19.7300	24.3049	19.5657	24.5802	21.3941	19.2844	21.0542	23.4158	23.7681	22.2694	23.1173	24.3618	19.3288	20.8220
Object 3	19.2771	20.1729	5.4403	23.3305	24.2668	22.4946	19.4241	24.5365	23.8022	20.7157	22.2620	24.9087	23.2941	24.0338	21.5996	21.8237	22.3643	20.6145	23.4941	22.0233
Object 4	22.8809	20.8465	19.8323	4.9511	21.1748	23.7287	23.6818	23.0111	19.8010	19.1293	22.3590	20.8049	24.6365	24.8854	20.7197	23.8049	24.3767	22.5852	24.3041	24.6624
Object 5	22.2949	23.3703	22.4605	19.1551	4.8931	22.8778	22.1272	21.2339	24.6228	23.9772	24.0945	21.2352	22.5591	24.2353	24.6010	23.0108	20.2407	22.9231	19.4323	21.4404
Object 6	23.0016	24.6024	23.8657	21.9073	23.5405	4.8341	24.8307	24.9278	24.1849	21.3333	21.7285	20.4801	23.7065	24.2970	24.4823	22.3497	22.5932	19.8933	24.3983	21.7024
Object 7	20.2340	24.3979	23.5755	24.2949	20.7097	23.0394	5.3286	19.7369	21.4439	20.6517	23.3000	20.7003	24.3772	23.9595	21.3402	21.9874	23.1688	24.0062	22.6578	22.4484
Object 8	20.9563	21.7385	23.2828	24.3064	23.3251	19.1117	23.0487	11.2480	21.6269	19.7022	23.8881	20.9491	20.4774	21.0563	21.2542	22.2793	22.3715	21.3749	21.3888	22.0922
Object 9	22.9452	24.7055	23.3341	21.4005	23.9912	19.8060	19.3628	19.5055	5.1504	20.9453	20.8104	19.0701	22.2394	19.5722	19.8791	22.7868	24.1559	24.8453	22.4250	24.9811
Object 10	22.3212	22.0928	20.9841	21.5800	21.9508	19.4262	24.3264	19.3878	21.6171	5.6533	21.3672	22.6808	23.9118	24.3174	24.5867	20.1447	20.5515	24.3872	22.5602	22.0230
Object 11	22.6769	23.9165	22.1913	20.2125	21.7234	21.5675	24.7963	22.7203	23.1723	23.3210	14.3278	22.1019	22.3402	19.9390	22.3723	23.1688	21.5587	24.0176	23.3883	21.1602
Object 12	21.7253	21.3183	23.6533	23.4056	21.5817	23.1625	24.6713	23.7054	23.2334	19.6560	21.3396	5.1818	21.7563	19.3020	20.3721	24.0051	19.0939	24.1823	19.4684	23.0143
Object 13	22.0013	20.3080	22.4297	19.7331	23.0270	22.5975	19.3359	19.3381	19.9150	19.1177	21.6111	23.9933	5.2348	22.1208	24.1832	19.5862	24.4483	19.6481	22.1020	19.8589
Object 14	22.3562	19.0275	23.6001	24.0923	24.5009	24.9218	22.0308	20.6285	19.6045	22.0471	22.5137	23.5773	19.4978	5.3232	22.1019	20.0263	24.6313	22.5429	21.6438	24.6515
Object 15	22.9355	21.7117	24.0382	22.1957	22.3233	23.0804	21.2031	20.4357	22.4735	24.2013	21.4407	19.6757	21.6631	20.8011	4.8028	24.0002	21.4218	21.3411	21.1627	19.8415
Object 16	20.5608	19.5209	21.5764	20.5437	20.7853	21.5492	19.7152	21.9704	23.2384	20.4614	23.7104	19.4445	21.3633	19.0204	20.3241	4.0026	20.1351	19.8549	20.6085	20.0494
Object 17	19.8319	22.5933	24.4063	24.6363	20.3271	21.8960	21.2561	22.1427	20.5892	19.4101	21.6180	20.0431	19.1566	24.7281	21.5836	24.7694	5.5248	19.0441	23.0802	23.2357
Object 18	22.8708	22.3139	20.3087	23.6342	20.3682	21.2252	24.3456	24.1383	21.4146	20.9081	22.6518	24.4612	24.4546	22.5496	20.9954	24.1184	21.6544	3.8087	19.1991	22.1946
Object 19	23.2990	20.0758	21.0192	20.1263	20.9316	21.4231	22.2914	19.2924	22.3164	20.6489	20.4490	20.4589	19.9250	24.7385	24.6140	23.9123	23.3696	20.0549	4.7207	20.1327
Object 20	19.0072	20.8985	23.1977	22.7515	22.2584	21.6342	20.7246	22.0100	23.5693	23.5744	22.4563	23.4860	22.8732	19.7393	22.0264	21.0836	19.5529	19.8871	20.1890	19.8832

Table 3. Confusion matrix of the second indoor environment.

	Object 1	Object 2	Object 3	Object 4	Object 5	Object 6	Object 7	Object 8	Object 9	Object 10	Object 11	Object 12	Object 13	Object 14	Object 15	Object 16	Object 17	Object 18	Object 19	Object 20
Object 1	3.6294	30.6232	27.5079	30.6535	29.5294	27.3902	28.1140	29.1875	30.8300	30.9596	27.6305	30.8824	30.8287	28.9415	30.2011	27.5675	28.6870	30.6629	30.1688	30.8380
Object 2	29.6230	7.0714	30.3965	30.7360	29.7149	30.0310	29.9725	28.5689	29.6219	27.6847	29.8242	27.1273	28.1077	27.1847	27.3885	30.2938	29.7793	28.2684	30.8009	27.1378
Object 3	28.7550	28.5262	8.5310	30.1808	27.7475	28.9591	28.7823	29.5853	29.8375	30.0187	28.1041	29.7188	29.6204	27.6504	27.4760	28.9935	30.8390	28.3615	29.3411	27.8952
Object 4	30.0051	28.0204	29.0238	8.3982	30.5636	30.8372	29.1889	27.5545	27.5972	28.0300	30.3629	28.0171	30.2571	27.9741	30.7171	28.3999	27.7864	28.0043	29.4642	28.8932
Object 5	28.4066	30.3233	29.3411	29.1989	8.8344	28.1434	30.0288	30.0149	28.5218	29.2713	27.3034	27.2158	29.1232	30.1167	30.7360	27.5196	29.2753	28.8776	27.0476	28.3485
Object 6	27.6487	30.1771	28.2449	29.1141	27.6626	8.2040	28.0519	29.6163	29.7569	29.9926	28.8022	27.3353	27.9159	30.6533	27.6095	30.3033	29.1534	30.9845	27.3127	28.7707
Object 7	27.4266	30.8476	27.0185	30.0996	30.2692	30.4748	7.1689	28.5991	28.0395	30.2003	28.7257	30.6426	27.7274	28.0552	27.5822	27.5443	30.4772	29.3188	29.1994	27.5798
Object 8	30.4121	29.4882	28.4038	29.0530	28.6072	27.3039	27.9597	27.2466	27.7356	27.9598	28.6691	27.1986	30.6109	30.7791	28.9635	28.9570	28.3509	30.6002	28.4770	27.4448
Object 9	30.1210	28.5590	27.9668	28.6156	27.3858	27.5279	30.7682	30.8245	8.1504	27.2391	27.9391	28.4126	30.2848	27.0616	27.1721	27.6760	29.5965	29.9269	29.5910	28.8037
Object 10	29.1880	28.1853	29.9788	27.7558	29.7471	27.7340	28.4739	29.5025	30.1209	7.1623	30.7175	30.1029	28.9472	28.7434	28.7871	28.2254	29.0340	29.0431	30.2705	30.1793
Object 11	29.5773	28.5144	30.2463	29.1313	28.4029	30.7560	30.5038	29.2006	29.4899	29.3482	4.4155	28.2050	28.8837	27.9220	30.3772	27.7791	27.9037	27.6828	27.9107	28.7428
Object 12	28.2444	30.6935	28.7208	27.7393	30.6195	30.9190	28.7555	27.4445	28.0323	28.6349	29.3796	7.5244	29.4114	29.8449	27.8870	27.4697	28.1867	28.2751	28.6967	29.0314
Object 13	27.3421	28.0499	30.2041	27.1169	30.7154	29.9213	28.9544	29.3141	27.9491	28.8354	30.8524	29.1872	8.0423	27.9264	28.9556	29.4962	29.7165	28.5821	28.4697	30.9519
Object 14	27.1510	30.5407	30.6531	30.1847	27.3948	28.0475	28.3414	29.7189	27.5462	29.8849	27.4270	29.6150	28.9767	8.5581	29.8601	30.6149	30.5637	28.3367	29.7950	27.7912
Object 15	27.1222	29.9763	29.0001	28.9197	30.6189	29.4395	29.4707	30.4378	30.2220	29.3069	27.7317	27.9597	30.5460	27.1147	7.9798	27.6717	30.9147	29.8508	29.0019	28.8844
Object 16	27.2385	29.7279	27.1697	27.2858	29.0866	27.3869	30.2726	30.2702	29.8898	27.5995	29.6384	29.0744	30.8919	29.5960	30.2013	7.9076	28.7296	30.3013	27.3339	27.5327
Object 17	27.6936	28.5638	30.3255	30.2135	27.2419	28.5970	29.1075	28.6672	29.6274	29.5119	28.1679	28.7266	27.0619	30.9363	27.6687	27.4249	7.7448	27.7925	28.9588	28.3580
Object 18	30.8065	30.6813	27.2107	29.9514	28.0765	28.6913	29.1915	30.7709	28.6710	30.9322	28.2058	29.8044	29.6654	29.1565	29.7924	29.6661	27.7125	7.2560	30.9963	27.6845
Object 19	27.1304	29.2448	30.5275	29.6767	27.7617	28.4757	28.8429	30.9266	27.6256	30.4221	29.5791	28.5051	27.7637	28.7130	28.9281	27.4824	29.3580	27.9048	7.7692	29.3319
Object 20	28.0072	28.1618	29.4684	28.0611	30.2975	30.9307	29.9210	28.3755	29.3363	27.4311	30.6252	30.5186	30.2710	28.0429	29.3774	27.0901	28.7010	28.2509	27.6459	7.3575

Table 4. Confusion matrix of the third indoor environment.

	Object 3	Object 5	Object 9	Object 12	Object 18
Object 3	3.5521	17.0391	16.9653	15.4878	15.3570
Object 5	16.4951	4.9195	16.0212	16.7558	15.6714
Object 9	17.2538	15.7653	4.0119	17.0972	17.6727
Object 12	17.8779	16.6416	15.4159	3.2986	15.7725
Object 18	17.5222	15.7628	17.4429	15.7306	4.8585

more challenging, we tried to occlude the targets. Despite the uncovered parts were correctly decrypted, we obtained high values for the correctness measures due to the occluded ones.

Regarding the execution time, to explore the first and second environment the rover took about 20 to 50 min for each one. This is due to the random approach of the used SLAM algorithm, and also due to the fact that for this kind of rover the exploration under the desks is difficult. Comparing with the timing reported in [8], the lower bound has been improved by 10 min. This is due to the tilt movement implemented in the new version of the SLAM algorithm, which allows to find targets quickly. Concerning the third environment, the rover took about 10 to 25 min to explore it. This is mainly due to the fact that no obstacles, such as seats and desks, were present.

4.2 Experiments with the Small-Scale UAV

In Fig. 6b, the small-scale UAV used in a wide outdoor environment is shown. The rotors are handled by a Pixhawk 2, while the camera, the visual cryptography algorithm, and the Wi-Fi connection are managed, as for the rover, by the Raspberry Pi 2. Concerning the correctness values, they are reported in Table 5. In these experiments, we obtained the best results in term of correctness. This is due to the fact that the pose of the object placed on the ground is the same to the pose of the object at the time of the server initialization. By using the UAV, also flat objects, such as credit cards, can be correctly decrypted (as depicted in Fig. 8). A condition that must be respected for achieving a correct decryption is that the drone must perform a stabilized flight. This means that the flight height must be the same both during the target acquired and during the server initialization. In fact, zoom in/out activities (i.e., change in flight height), pitch, roll, and jaw movements can influence the share alignment.

(a) **(b)**

Fig. 8. The cup (a) and the credit card (b) viewed from the UAV.

Table 5. Confusion matrix of the outdoor environment.

	Object 1	Object 2	Object 3	Object 4	Object 5	Object 6	Object 7	Object 8	Object 9	Object 10	Object 11	Object 12	Object 13	Object 14	Object 15	Object 16	Object 17	Object 18	Object 19	Object 20
Object 1	3.3147	20.7174	18.3810	20.7401	19.8971	18.2926	18.8355	19.6406	20.8725	20.8947	18.4728	20.9118	20.8715	19.4561	20.4008	18.4257	19.2653	20.7472	20.3766	20.8785
Object 2	19.9672	2.5357	20.5474	20.8020	20.0362	20.2732	20.2294	19.1767	19.9664	18.5136	20.1181	18.0955	18.8308	18.1385	18.2914	20.4704	20.0845	18.9513	20.8507	18.1033
Object 3	19.3162	19.1447	3.2655	20.3856	18.5606	19.4693	19.3368	19.9389	20.1281	20.2641	18.8281	20.0391	19.9653	18.4878	18.3570	19.4951	20.8792	19.0212	19.7558	18.6714
Object 4	20.2538	18.7653	19.5179	3.1991	20.6727	20.8779	19.6416	18.4159	18.4479	18.7725	20.5222	18.7628	20.4429	18.7306	20.7878	19.0500	18.5898	18.7533	19.8481	19.4199
Object 5	19.0550	20.4925	19.7558	19.6492	3.4172	18.8575	20.2716	20.2612	19.1413	19.7035	18.2276	18.1619	19.5924	20.3375	20.8020	18.3897	19.7065	19.4082	18.0357	19.0114
Object 6	18.4865	20.3829	18.9336	19.5856	18.4969	3.1021	18.7889	19.9622	20.0676	20.2445	19.3516	18.2515	18.6869	20.7400	18.4571	20.4775	19.6150	20.9884	18.2345	19.3280
Object 7	18.3200	20.8857	18.0139	20.3247	20.4519	20.6061	2.5844	19.1993	18.7796	20.4002	19.2942	20.7319	18.5455	18.7914	18.4366	18.4082	20.6079	19.7391	19.6496	18.4349
Object 8	20.5591	19.8662	19.0529	19.5397	19.2054	18.2279	18.7197	2.6233	18.5517	18.7199	19.2518	18.1490	20.7081	20.8344	19.4726	19.4678	19.0132	20.7002	19.1077	18.3336
Object 9	20.3408	19.1692	18.7251	19.2117	18.2894	18.3959	20.8262	20.8684	3.0752	18.1793	18.7043	19.0595	20.4636	18.0462	18.1291	18.5070	19.9473	20.1952	19.9432	19.3528
Object 10	19.6410	18.8890	20.2341	18.5669	20.0603	18.5505	19.1055	19.8769	20.3407	2.5811	20.7882	20.3271	19.4604	19.3076	19.3404	18.9190	19.5255	19.5323	20.4529	20.3845
Object 11	19.9330	19.1358	20.4347	19.5985	19.0522	20.8170	20.6278	19.6505	19.8674	19.7611	2.7077	18.9037	19.4128	18.6915	20.5329	18.5843	18.6778	18.5121	18.6830	19.3071
Object 12	18.9333	20.7701	19.2906	18.5544	20.7146	20.9392	19.3166	18.3334	18.7742	19.2262	19.7847	2.7622	19.8085	20.1336	18.6652	18.3523	18.8900	18.9563	19.2725	19.5236
Object 13	18.2565	18.7874	20.4030	18.0877	20.7866	20.1910	19.4658	19.7356	18.7119	19.3765	20.8893	19.6404	3.0211	18.6948	19.4667	19.8722	20.0374	19.1865	19.1023	20.9639
Object 14	18.1132	20.6555	20.7399	20.3886	18.2961	18.7856	19.0061	20.0392	18.4097	20.1637	18.3203	19.9613	19.4825	3.2791	20.1451	20.7112	20.6728	19.0025	20.0962	18.5934
Object 15	18.0916	20.2322	19.5001	19.4398	20.7142	19.8296	19.8530	20.5783	20.4165	19.7302	18.5488	18.7198	20.6595	18.0860	2.9899	18.5038	20.9360	20.1381	19.5014	19.4133
Object 16	18.1789	20.0459	18.1273	18.2143	19.5649	18.2902	20.4544	20.4526	20.1673	18.4496	19.9788	19.5558	20.9189	19.9470	20.4010	2.9538	19.2972	20.4759	18.2504	18.3995
Object 17	18.5202	19.1728	20.4941	20.4101	18.1814	19.1978	19.5806	19.2504	19.9706	19.8839	18.8760	19.2950	18.0465	20.9522	18.5015	18.3186	2.8724	18.5944	19.4691	19.0185
Object 18	20.8549	20.7610	18.1580	20.2136	18.8074	19.2685	19.6436	20.8282	19.2532	20.9492	18.9044	20.1033	19.9990	19.6174	20.0943	19.9996	18.5344	2.6280	20.9972	18.5134
Object 19	18.0978	19.6836	20.6456	20.0075	18.5713	19.1067	19.3822	20.9449	18.4692	20.5666	19.9343	19.1288	18.5728	19.2848	19.4461	18.3618	19.7685	18.6786	2.8846	19.7490
Object 20	18.7554	18.8713	19.8513	18.7958	20.4731	20.9480	20.1907	19.0316	19.7522	18.3233	20.7189	20.6390	20.4533	18.7822	19.7831	18.0675	19.2758	18.9382	18.4845	2.6788

5 Conclusions

In recent years, autonomous (or semi-autonomous) small-scale robots have been increasingly used to face dangerous activities, including civilian and military operations. Usually, these robots send the acquired data to a ground station to perform a wide range of processing. In some cases, there may be the need to protect the sent data from intruders. In this paper, a system to encrypt video streams acquired by small-scale robots engaged in exploration tasks is presented. In particular, the paper shows results for two small-scale robots: a rover and a UAV. While the latter is manually piloted during the mission, the first is equipped with a SLAM algorithm that allows it to explore autonomously the environment and search exhaustively different targets. The experimental tests where performed in indoor and outdoor environments showing the effectiveness of the proposed method. Currently, in literature, there are no other approaches comparable with that we propose. For this reason, it can be considered as baseline in the area of encrypted target search by small-scale robots.

References

1. Cacace, J., Finzi, A., Lippiello, V., Furci, M., Mimmo, N., Marconi, L.: A control architecture for multiple drones operated via multimodal interaction in search rescue mission. In: IEEE International Symposium on Safety, Security, and Rescue Robotics, pp. 233–239 (2016)
2. Kiyani, M.N., Khan, M.U.M.: A prototype of search and rescue robot. In: International Conference on Robotics and Artificial Intelligence, pp. 208–213 (2016)
3. Avola, D., Foresti, G.L., Martinel, N., Micheloni, C., Pannone, D., Piciarelli, C.: Real-time incremental and geo-referenced mosaicking by small-scale UAVs. In: Battiato, S., Gallo, G., Schettini, R., Stanco, F. (eds.) ICIAP 2017. LNCS, vol. 10484, pp. 694–705. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68560-1_62
4. Avola, D., Foresti, G.L., Martinel, N., Micheloni, C., Pannone, D., Piciarelli, C.: Aerial video surveillance system for small-scale UAV environment monitoring. In: IEEE International Conference on Advanced Video and Signal Based Surveillance, pp. 1–6 (2017)
5. Avola, D., Cinque, L., Foresti, G.L., Martinel, N., Pannone, D., Piciarelli, C.: A UAV video dataset for mosaicking and change detection from low-altitude flights. IEEE Trans. Syst. Man Cybern. Syst. **PP**, 1–11 (2018)
6. Kaur, T., Kumar, D.: Wireless multifunctional robot for military applications. In: International Conference on Recent Advances in Engineering Computational Sciences, pp. 1–5 (2015)
7. Kopuletý, M., Palasiewicz, T.: Advanced military robots supporting engineer reconnaissance in military operations. In: Mazal, J. (ed.) MESAS 2017. LNCS, vol. 10756, pp. 285–302. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76072-8_20
8. Avola, D., Cinque, L., Foresti, G.L., Marini, M.R., Pannone, D.: A rover-based system for searching encrypted targets in unknown environments. In: International Conference on Pattern Recognition Applications and Methods, vol. 1, pp. 254–261 (2018)

9. Avola, D., Foresti, G.L., Cinque, L., Massaroni, C., Vitale, G., Lombardi, L.: A multipurpose autonomous robot for target recognition in unknown environments. In: IEEE International Conference on Industrial Informatics, pp. 766–771 (2016)
10. Naor, M., Shamir, A.: Visual cryptography. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 1–12. Springer, Heidelberg (1995). https://doi.org/10.1007/BFb0053419
11. Liu, S., Fujiyoshi, M., Kiya, H.: A cheat preventing method with efficient pixel expansion for Naor-Shamir's visual cryptography. In: IEEE International Conference on Image Processing, pp. 5527–5531 (2014)
12. Li, P., Yang, C.N., Kong, Q.: A novel two-in-one image secret sharing scheme based on perfect black visual cryptography. J. R. Time Image Process. **14**, 41–50 (2018)
13. Shivani, S.: VMVC: verifiable multi-tone visual cryptography. Multimed. Tools Appl. **77**, 5169–5188 (2018)
14. Alex, N.S., Anbarasi, L.J.: Enhanced image secret sharing via error diffusion in halftone visual cryptography. In: International Conference on Electronics Computer Technology, pp. 393–397 (2011)
15. Pahuja, S., Kasana, S.S.: Halftone visual cryptography for color images. In: International Conference on Computer, Communications and Electronics, pp. 281–285 (2017)
16. Lin, C.C., Tsai, W.H.: Visual cryptography for gray-level images by dithering techniques. Pattern Recognit. Lett. **24**, 349–358 (2003)
17. Babu, R., Sridhar, M., Babu, B.R.: Information hiding in gray scale images using pseudo-randomized visual cryptography algorithm for visual information security. In: International Conference on Information Systems and Computer Networks, pp. 195–199 (2013)
18. Hou, Y.C.: Visual cryptography for color images. Pattern Recognit. **36**, 1619–1629 (2003)
19. Stinson, D.R.: An introduction to visual cryptography. In: Public Key Solutions, pp. 28–30 (1997)
20. Shyu, S.J.: Efficient visual secret sharing scheme for color images. Pattern Recognit. **39**, 866–880 (2006)
21. Yang, D., Doh, I., Chae, K.: Enhanced password processing scheme based on visual cryptography and OCR. In: International Conference on Information Networking, pp. 254–258 (2017)
22. Kadhim, A., Mohamed, R.M.: Visual cryptography for image depend on RSA algamal algorithms. In: Al-Sadeq International Conference on Multidisciplinary in IT and Communication Science and Applications, pp. 1–6 (2016)
23. Joseph, S.K., Ramesh, R.: Random grid based visual cryptography using a common share. In: International Conference on Computing and Network Communications, pp. 656–662 (2015)
24. Leonard, J.J., Durrant-Whyte, H.F.: Simultaneous map building and localization for an autonomous mobile robot. In: IEEE/RSJ International Workshop on Intelligent Robots and Systems, Intelligence for Mechanical Systems, pp. 1442–1447 (1991)
25. Sim, R., Roy, N.: Global a-optimal robot exploration in slam. In: IEEE International Conference on Robotics and Automation, pp. 661–666 (2005)
26. Trivun, D., Šalaka, E., Osmanković, D., Velagić, J., Osmić, N.: Active SLAM-based algorithm for autonomous exploration with mobile robot. In: IEEE International Conference on Industrial Technology, pp. 74–79 (2015)

27. Li, C., Wei, H., Lan, T.: Research and implementation of 3D SLAM algorithm based on kinect depth sensor. In: International Congress on Image and Signal Processing, BioMedical Engineering and Informatics, pp. 1070–1074 (2016)
28. Walas, K., Nowicki, M., Ferstl, D., Skrzypczyński, P.: Depth data fusion for simultaneous localization and mapping - RGB-DD SLAM. In: IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, pp. 9–14 (2016)
29. Chen, L., Sun, L., Yang, T., Fan, L., Huang, K., Xuanyuan, Z.: RGB-T SLAM: a flexible slam framework by combining appearance and thermal information. In: IEEE International Conference on Robotics and Automation, pp. 5682–5687 (2017)
30. Mur-Artal, R., Tardós, J.D.: ORB-SLAM2: an open-source slam system for monocular, stereo, and RGB-D cameras. IEEE Trans. Robot. **33**, 1255–1262 (2017)
31. Camurri, M., Bazeille, S., Caldwell, D.G., Semini, C.: Real-time depth and inertial fusion for local SLAM on dynamic legged robots. In: IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, pp. 259–264 (2015)
32. Bu, S., Zhao, Y., Wan, G., Liu, Z.: Map2DFusion: real-time incremental UAV image mosaicing based on monocular slam. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 4564–4571 (2016)
33. Kim, D.Y., Kim, J., Kim, I., Jun, S.: Artificial landmark for vision-based slam of water pipe rehabilitation robot. In: International Conference on Ubiquitous Robots and Ambient Intelligence, pp. 444–446 (2015)
34. Balcilar, M., Yavuz, S., Amasyali, M.F., Uslu, E., Çakmak, F.: R-slam: resilient localization and mapping in challenging environments. Robot. Auton. Syst. **87**, 66–80 (2017)
35. Boiangiu, C.A., Bucur, I., Tigora, A.: The image binarization problem revisited: perspectives and approaches. J. Inf. Syst. Oper. Manag. **6**, 1–10 (2012)
36. Knuth, D.E.: Digital halftones by dot diffusion. ACM Trans. Graph. **6**, 245–273 (1987)
37. Bayer, B.E.: An optimum method for two-level rendition of continuous-tone pictures. In: IEEE International Conference on Communications, vol. 26, pp. 11–15 (1973)
38. Jarvis, J.F., Judice, C.N., Ninke, W.: A survey of techniques for the display of continuous tone pictures on bilevel displays. Comput. Graph. Image Process. **5**, 13–40 (1976)
39. Avola, D., Bernardi, M., Cinque, L., Foresti, G.L., Massaroni, C.: Adaptive bootstrapping management by keypoint clustering for background initialization. Pattern Recognit. Lett. **100**, 110–116 (2017)
40. Avola, D., Cinque, L., Foresti, G.L., Massaroni, C., Pannone, D.: A keypoint-based method for background modeling and foreground detection using a PTZ camera. Pattern Recognit. Lett. **96**, 96–105 (2017)