Journal of Industrial Information Integration 15 (2019) 42–49

Contents lists available at ScienceDirect

Journal of Industrial Information Integration

journal homepage: www.elsevier.com/locate/jii

Mixed reality and remote sensing application of unmanned aerial vehicle in fire and smoke detection

Shabnam Sadeghi Esfahlani

Faculty of Science and Engineering, Anglia Ruskin University, Chelmsford, United Kingdom

ARTICLE INFO

Keywords:
Fire detection
Autonomous flight
Crazyflie 2.0
Monocular camera
Computer vision

ABSTRACT

This paper proposes the development of a system incorporating inertial measurement unit (IMU), a consumer-grade digital camera and a fire detection algorithm simultaneously with a nano Unmanned Aerial Vehicle (UAV) for inspection purposes. The video streams are collected through the monocular camera and navigation relied on the state-of-the-art indoor/outdoor Simultaneous Localisation and Mapping (SLAM) system. It implements the robotic operating system (ROS) and computer vision algorithm to provide a robust, accurate and unique inter-frame motion estimation. The collected onboard data are communicated to the ground station and used the SLAM system to generate a map of the environment. A robust and efficient re-localization was performed to recover from tracking failure, motion blur, and frame lost in the data received. The fire detection algorithm was deployed based on the color, movement attributes, temporal variation of fire intensity and its accumulation around a point. The cumulative time derivative matrix was utilized to analyze the frame-by-frame changes and to detect areas with high-frequency luminance flicker (random characteristic). Color, surface coarseness, boundary roughness, and skewness features were perceived as the quadrotor flew autonomously within the clutter and congested area. Mixed Reality system was adopted to visualize and test the proposed system in a physical environment, and the virtual simulation was conducted through the Unity game engine. The results showed that the UAV could successfully detect fire and flame, autonomously fly towards and hover around it, communicate with the ground station and simultaneously generate a map of the environment. There was a slight error between the real and virtual UAV calibration due to the ground truth data and the correlation complexity of tracking real and virtual camera coordinate frames.

1. Introduction

Recent advances in robot autonomy using lightweight drones and remote sensing provides the ability to perform tasks based on the current situation and sensing without human intervention. There are various types of drones which differ in size from large fixed-wing unmanned air vehicle (UAV) to smart dust [1] with several micro-electromechanical systems [2,3]. They have several capabilities based on the size and the payload with the goal of examining an environment, report to the human engineer based on obtained data and achieve full coverage of the state. The onboard sensor blendings facilitate the cognitive intelligent quadrotor to operate in cluttered, GPS-denied environments safely. The SLAM or concurrent mapping and localization (CML) facilitates the synchronous estimate of the state of a robot. In that, the system constructs a map of the environment to locate the robot in it. The cognitive autonomy of robots requires reactive independence to perform SLAM/CML and manage contrary information. Operating UAV's with a camera and streaming color video inputs could have many interests particularly in large and open spaces (i.e., fire detection) [4]. The smoke and flame of inaccessible areas could be recognized quicker and earlier before developing [5,6]. A video is a volume sensor rather than a point sensor that conceivably monitors a larger range and has a much higher probability of successful early exposure [7]. The use of color video inputs from a non-stationary camera could detect fire at a speed of 30 frames per second (fps) [8]. The progress on fire detection technology has been substantial due to advance in sensors, microelectronics, information technologies (e.g., computer vision system) and a numerous knowledge of a fire physics [9–13].

In recent years, UAVs have successfully been utilized both in research and for commercial applications, and there has been significant progress in the design of robust control software and hardware. Several researchers [14–16] have carried the prototype of UAV systems and its simulation based on the Robot Operating System (ROS) and Gazebo. The gazebo offers the ability to simulate robots in complex indoor and

E-mail address: shabnam.sadeghi-esfahlani@anglia.ac.uk.

https://doi.org/10.1016/j.jii.2019.04.006
Received 5 June 2018; Received in revised form 3 November 2018; Accepted 28 April 2019
Available online 01 May 2019

outdoor environments.[1] Motivated by this, we propose the use of Mixed Reality (MR) as a tool to visualize and test the performance of our algorithm in a virtual environment with the capability to be transferred to a physical world. Our main contribution resides in the implementation of computer vision algorithms, and the build of a general simulation platform based on the Unity game engine and ROS. It is done to control and test the UAV and incorporate the SLAM system and fire detection based on visualization algorithm. Colour and motion information computed from video sequences adjusted through computer vision to be insensitive to camera motion. A temporally extended normalized covariance descriptor (TENCD) is designed to describe spatiotemporal video blocks. Spurious fire pixels are automatically eliminated employing an erode operation. Missing fire pixels are attained applying region growing method. The inspection drone in this study is an open source nano aerial vehicle (Crazyflie 2.0) that is paired with a mini ArduCAM camera. The Crazyflie weighs only 27 g that fits in the palm of a hand with a wingspan of 9 cm and two Micro-electro-mechanical.[2] Due to its limited payload (15 g) a mini ArduCAM camera is used for video streaming (the resolution of 320 × 240 pixels) and SLAM generation via object and place recognition method. Open-source Robot Operating System (ROS)[3] is used and interfaced with the Unity game engine. Our algorithm is examined in a Mixed Reality (MR) environment for fire detection, autonomous flight, and SLAM performance. Fig. 1(a) illustrates the Crazyflie 2.0 with the mini ArduCAM that is employed in this study.

2. Background

SLAMs are navigation solutions which are less reliant on external navigation systems (i.e., GPS) and provide the solution by estimating navigation states and some properties of the environment [17]. A solution to real-time structure from motion (SFM) or monocular SLAM can be categorized into two main paradigms: filtering and optimization based approaches [18–23]. Some of the filtering methods are; Extended Kalman Filter (EKF), Extended Kalman Filter (EKF-SLAM), and Rao–-Blackwellized particle filters (FastSLAM) [24–26]. Visual bundle adjustment (BA) or full SLAM is an optimisation method that uses Levenberg–Marquardt or the Gauss–Newton methods [27–30]. It is the optimal non-linear least-squares formulation of SLAM problem. It improves a visual reconstruction to produce collectively optimal structure with a minimum reprojection error MLPE (maximum likelihood parameter estimate) of all landmark observations in keyframes [30–35]. The goal of BA is to reduce an error between perceived and predicted measurements of various landmarks sensed from sensor frames [36]. Visual and inertial bundle adjustment (IMUBA) is also an optimization method that is linked with the inertial measurement unit (IMU) [27,28]. For BA in real-time a local approach could be adopted whereas, in IMUBA a reasonable approach is to actively slide the frame of the most recent poses (landmarks) and marginalize the rest into a former distribution [36–38]. Carrying prior distributions influenced by marginalization necessitates an expensive global optimization at loop-closure (often all parameters in the loop must be adjusted) to achieve the correct results [29]. Keivan et al. [28] used adaptive asynchronous conditioning instead of marginalization and showed that this procedure is more robust and avoids locking in unreliable parameter estimations when used adaptively. Sibley [39] has proposed the pose graph optimization (a graph of relative poses with landmarks specified about the poses) to solve SLAM problems with too many loops. They converted SLAM into a sparse set of pose constraints (marginalize landmark parameters onto pose parameters and optimize a small confined subset of a map to approximate the full solution with various forms of marginalization without overlapping estimations. The graph defines a connected Riemannian manifold with a distance metric based on shortest paths. Sparse BA is the standard method for optimizing a structure-from-motion (SFM) that marginalizes landmark parameters onto pose parameters and solves for the optimal path estimation that iterates the procedure to convergence. To overcome computational complexity issues in MLPE, Sibley et al. [36] proposed to adjust all parameters in a loop by using Adaptive-Relative BA (ARBA). ARBA is defined by a connected Riemannian manifold using an adaptive optimization strategy in that instead of optimization in single Euclidean space; a metric-space is defined based on shortest paths to solve for the full ML in constant time and loop closure. Forster et al. [40] have shown that visual-inertial odometry (Visual inertial navigation and mapping (VINAM)) approach can obtain highly accurate state estimation via nonlinear optimization. Although, real-time optimization grows infeasible as the trajectory develops over time but leads to a quick maturity of the number of variables in the optimization. Forster et al. [27,40] proposed the preintegration theory to address the manifold structure of the rotation group. This is an inertial measurement between selected keyframes into unique relative motion constraints that derive the expression for the maximum a posteriori state estimator. This approach enables the analytical correction of a posteriori bias and avoids optimizing over the 3D points that result in accelerating the computation. Huang et al. [41] used RGB-D cameras to capture RGB color images augmented with depth data at each pixel that provided an autonomous micro air vehicle with fast and reliable state estimates. They used an onboard RGB-D camera and an inertial measurement unit (IMU) for autonomous flight. Zhang and Singh [42] presented a general framework for combining visual odometry and lidar odometry. Visual odometry handles rapid motion detection, local position control, and stability that typically consists of estimating its relative motion at each time step by aligning successive sensor measurements [43]. It suffers from long-term drift, thus, is not suitable for building large-scale maps. To address the issue, Zhang and Singh [42] incorporated loop closure technique. This technique enables pose (position and orientation) detection which is a motion estimation as point clouds are generated and simultaneously corrects accumulated drifts over the history of frames.

Crazyflie has two micro-controller units known as; (MCU) STM32F405 and nRF51822. The former controls the flight of the quadrotor through the power to the motor driver and reads inertia data at 168 *MHz*. Tri-axis gyroscope, accelerometer, gravity, magnetic angular rate and a high precision pressure sensor. Nordic Semiconductor (nRF51822) handles power management, radio communication, the Bluetooth Low Energy (BLE) and Compressed Real-Time Protocol (CRTP) [44–46]. The MCUs are combined into a point cloud using transformation library in ROS to keep track of coordinate frames and to generate compatible data between the quadrotor's IMU and Arducam's coordinate frames. A Crazyradio (Crazyradio PA) was used to provide wireless radio communication to the quadrotor. We implement ten DOF IMU sensor fusion algorithm[4] (nav-msgs/Odometry) in ROS [47]. A virtual environment was simulated in Unity with firefighting scenario. Colour detection algorithm was developed in OpenCV [48] to enable the drone to fly autonomously towards the fired hut in the virtual world. It flies towards the fire and smoke, hovers around it and sends the video streams to the ground station. The dynamic bag-of-words were utilized for place identification. We performed monocular SLAM based on VINAM where the inputs are collected from the Arducam, Crazyflie and mixed reality (MR) environment to examine the algorithm.

3. Methodology

SLAM facilitates the simultaneous estimation of a robot's state using

[1] http://gazebosim.org/.
[2] https://wiki.bitcraze.io/projects:crazyflie2:index.
[3] http://wiki.ros.org/ROS/Introduction.
[4] http://x-io.co.uk/open-source-imu-and-ahrs-algorithms/.

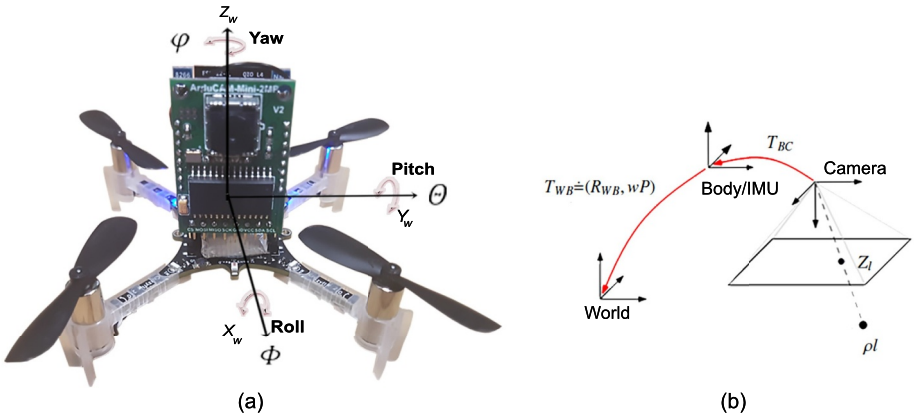

Fig. 1. (a) The Crazyflie 2.0 with ArduCAM-Mini-V2 its ESP8266 modules. (b) T_{WB} is the pose of the body frame B to the world frame W where B correlates to the inertial frame. T_{BC} is the pose of the camera in the B, taken from former calibration [40].

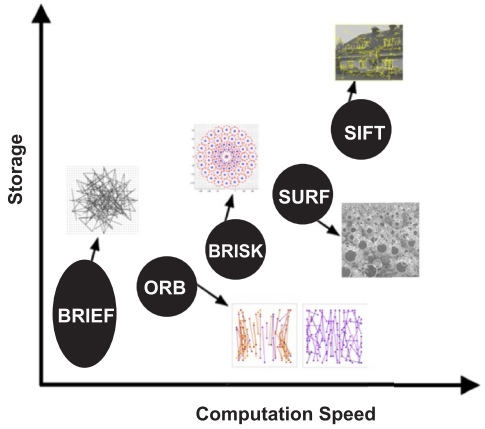

Fig. 2. The computation speed and memory requirement for Real Value Parameterization of front-end features (SURF [56] and SIFT [57]) and the binary descriptors (BRIEF, ORB and BRISK [51]) with an example of their pattern.

onboard sensors in the front-end, and a map is generated in the back-end as the robot perceives in an environment. The front-end of SLAM comprises of the algorithms and data structures to detect, match and estimate keypoints, edges and orientations. The back-end uses the information provided in the front-end to match over the previous frames to estimate poses and keypoint description and matching [49]. In this study, the UAV's state is reported by the onboard camera and an inertial measurement unit. The camera supports the measurement of geometry and appearance of 3D scenes. The IMU renders metric scale of monocular vision and gravity to enable motion estimation such as; UAV's position, orientation, a position of landmarks, point cloud and obstacles. Fig. 2 illustrates the performance of feature-based computation cost (detection, description or matching) and the cost of memory required to store and manage data. Binary feature pointers and descriptors have enhanced performance and compact representation that makes them an appealing explanation for many current purposes [49]. BRISK, SURF and SIFT require larger computational and memory corresponding to real value parameterization. SIFT algorithm [30] or at higher speed SURF [50] rely on 2D image key points that are invariant to many transformations and are tolerant to any distortions caused by viewpoint change, sudden movement or contrast variation [31]. BRISK detects key points in a scale-space pyramid, performing non-maxima suppression and interpolation across all scales [51].

3.1. Compound measurement of Arducam and Crazyflie

In this study, we use the optimal maximum a posteriori (MAP) estimate and a preintegration theory to combine Crazyflie's inertial estimations and Arducam's frames between two keyframes into a single contingent motion control [52,53]. The inertial measurements are integrated among two frames where the initial condition at the first frame is done using the state estimation. MAP estimation leads to a nonlinear optimization problem that involves quantities such as rotation and poses on smooth multiplication matrix [40]. At every iteration, IMU preintegration (reparametrization of the relative motion constraints) is utilized to avoid duplicating integration between all frames. IMU preintegration theory addresses the multiplication composition of the Special Orthogonal Group or rotation group $SO(3)$ [52]. The general derivation of the MAP estimator determines analytic expressions for the Jacobian optimization. $SO(3)$ represents the group of three-dimensional rotation matrices that form a smooth multiplication. It is formulated mathematically as; $SO(3) \doteq \{R \in R^{3\times3}: \ R^TR = I, \ det(R) = 1\}$.

$SO(3)$ is the usual matrix multiplication where its inverse is the matrix transpose $R^{-1} = R^T$. The tangent space to the product (at the identity) is called the Lie algebra ($\hat{SO}(3)$), which is the group of all rotations about the origin of space. The Crazyflie's IMU measurements (the rotation rate and the acceleration) concerns with inertial frame and the exponential map exp: $\hat{SO}(3) \rightarrow SO(3)$ associates an element of $\hat{SO}(3)$ to a rotation and corresponds to standard matrix exponential. Where its first-order approximation of the exponential map is formulated as:

$$exp(\hat{\phi}) \approx I + \hat{\phi} \tag{1}$$

The right hand Jacobian of the Special Orthogonal Group ($J_r(\varphi)$) relates additive increments in the tangent space to multiplicative increments is:

$$J_r(\phi) = I - \left(\frac{1 - cos(\|\phi\|)^2}{\|\phi\|}\right)\hat{\phi} + \left(\frac{\|\phi\| - sin(\|\phi\|)}{\|\phi\|^3}\right)\hat{\phi}^2 \tag{2}$$

In Fig. 1(b), it is assumed that the quadrotor's IMU frame (B) coincides with the tracked body frame. The transformation between the Arducam and Crazyflie is fixed and calculated from prior calibration. Fig. 1(b) illustrates that a front-end provides image measurements of landmarks at unknown position and selects a subset of images (keyframes) to compute a pose estimates [40].

$SE(3)$ or Special Euclidean Group defines the group of rigid motion in 3D [54]. It is the semi-direct product of Special Orthogonal Group and R^3 defined mathematically as;

$SE(3) \doteq (R, p)$: $R \in SO(3)$, $p \in R^3$. The optimisation of the matrix is performed using a standard approach (a retraction). A retraction (R_x) is

a map between an element δx of the tangent space at x and a neighborhood of $x \in M$ with M multiplications. The retraction for Special Orthogonal Group is defined as; $\boldsymbol{R}_R(\varphi) = RExp(\delta\varphi)$ and for Special Euclidean Group at $T \doteq (R, p)$ is formulated as; $\boldsymbol{R}_T(\delta\varphi, \delta p) = (RExp(\delta\varphi), p + R\delta p)$. The state of the system at time i is written as; $x_i \doteq [R_i, p_i, v_i, b_i^g, b_i^a]$. Where (R_i) is the Crazyflie's orientation, (p_i) is position, $(v_i \in R^3)$ is velocity, $(b_i^g, b_i^a \in R^3)$ are gyroscope and accelerometer biases respectively. If κ_L denote the set of all keyframes up to time L, the state of all keyframes is χ_L. C_i is the Arducam measurements at time (keyframe) i and it can observe numerous landmarks l with several pixel z_{il}. $\Im_{i,j}$ is the set of Crazyflie's IMU measurements between two consecutive keyframes i and j. A factor graph encodes the posterior probability of the variables χ_L, given the available measurement (Z_L) and prior measurement $(p(\chi_0))$ [53]:

$$p(\chi_L|Z_L) \propto p(\chi_0)p(Z_L|\chi_L) = p(\chi_0)\prod_{(i,j)\in\kappa_L} p(\Im_{i,j}|x_i, x_j)\prod_{i\in\kappa_L}\prod_{l\in C_i} p(z_{il}|x_i) \tag{3}$$

The Crazyflie and Arducam's calculated data are synchronized at discrete times intervals (L). The relative motion increments are independent of the pose and velocity at t_i. The preintegrated inertia measurements $\Delta\tilde{R}_{ij}$, $\Delta\tilde{v}_{ij}$ and $\Delta\tilde{p}_{ij}$ as a function of their noise $\delta\phi_{ij}$, δv_{ij} and δp_{ij} for orientation, velocity and position, respectively, are formulated as following [55];

$$\Delta\tilde{R}_{ij} = R_i^T R_j Exp(\delta\phi_{ij}) \tag{4}$$

$$\Delta\tilde{v}_{ij} = R_i^T(v_j - v_i - g\Delta t_{ij}) + \delta v_{ij} \tag{5}$$

$$\Delta\tilde{p}_{ij} = R_i^T\left(p_j - p_i - v_i\Delta t_{ij} - \frac{1}{2}g\Delta t_{ij}^2\right) + \delta p_{ij} \tag{6}$$

3.2. Binary descriptor

The front-end feature BRIEF (Binary Robust Independent Elementary Features) is a binary vector in that each segment is the result of a test between two of the pixels of the patch. The patches are smoothed with a Gaussian kernel to reduce noise. BRIEF is proposed by Calonder et al. [58], uses a sampling pattern, with sample points selected randomly from an isotropic Gaussian distribution centered at the feature location. Given its simple construction and compact storage, BRIEF requires the lowest computing and storage. Given a point P in an image, the BRIEF descriptor vector $B_i(P)$ is the i^{th} segment of the vector, x_i and y_i that are the offset of the randomly selected test points (the pair of pixels) based on a normal distribution $\mathcal{N}(0, \frac{1}{25}S_b^2)$ whose value must fall in $[-\frac{S_b}{2},...,\frac{S_b}{2}] \times [-\frac{S_b}{2},...,\frac{S_b}{2}]$:

$$B_i(P) = \begin{cases} 1, & \text{if } I(P + x_i) < I(P + y_i) \\ 0, & \text{otherwise.} \end{cases} \quad \forall \; i \in [1, ...L_b] \tag{7}$$

Where S_b is the size of the patch and L_b is the number of tests to perform (length of the descriptor) and $I(.)$ is the intensity of the pixel in the smoothed image. Each coordinate j of the pairs are selected by sampling the distributions $x_i^j \sim \mathcal{N}(0, \frac{1}{25}S_b^2)$ and $y_i^j \sim \mathcal{N}(a_i^j, \frac{4}{625}S_b^2)$ [58].

ORB (Oriented FAST and Rotated BRIEF) was proposed by Rublee et al. [59] and overcomes the lack of rotation invariance of BRIEF. ORB is a very fast binary descriptor based on BRIEF descriptor [58] and built upon the well-known FAST keypoints (Features from accelerated segment test) feature extraction methods. FAST corner detector uses a circle of 16 pixels (a Bresenham circle of radius 3) to classify whether a candidate point p is a corner. Each pixel in the circle is labeled by integer number 1 to 16 clockwise. If a set of N contiguous pixels in the circle are all brighter than the intensity of candidate pixel Ip plus a threshold value t or all darker than the intensity of candidate pixel p minus threshold value t, then p is classified as the corner. A local orientation is computed using an intensity centroid, which is a weighted averaging of pixel intensities in the local patch assumed not to be coincident with the center of the feature [60]. The orientation is the vector between the feature location and the centroid. The sampling pattern employed in ORB uses 256 pairwise intensity comparisons, but in contrast to BRIEF, is constructed via machine learning, maximizing the descriptor's variance and minimizing the correlation under various orientation changes. ORB matches visual features using PTAM (split tracking and mapping into two separate tasks) for real-time keypoint detection. Parallel thread optimization is used based on tracking and mapping dual architecture in [20]. Large SLAM problems with too many loops could be solved using pose-graph optimization algorithms by optimizing a set of relative pose constraints [39] considering 6 Degrees of Freedom pose-graph optimization. Inspired by Mur-Artal and Tardós [61] we adopted the ORB-SLAM to build a visibility graph and facilitate local track/map recoveries (re-localization).

3.3. Place recognition method

The bag-of-words technique utilizes a visual vocabulary to convert an image into a scattered numerical vector to create an image database [62,63]. The visible vocabulary is structured as a tree in an offline step over a large set of descriptors and is derived from a training image dataset. It discretizes the descriptor space into different visual words to detect revisited places. The search for initial correspondences is performed exhaustively between those features that belong to the same node of the tree. The database uses a direct index to store elements of an image in a node. The BDoW2 bag-of-words library is an efficient, opensource binary feature that uses FAST and BRIEF features (like ORB feature) [31]. This iterative method estimates the parameters of the mathematical model from a set of observed data [64]. The default parameters in BDoW2's for BRIEF are set to 500 keypoints, at 8 different scales with a scale factor of 1.2 and no non-maximum suppression. The ORB feature's defaults are increased to extract 1000 key points with non-maximum suppression to improve the keypoint distribution. Non-maximum suppression method reduces the descriptor matching performance at the cost of extracting more key points and more computation. The Keyframe-based SLAM performs map reconstruction over selected frames (keyframes) using batch optimization techniques, as mapping is not tied to frame-rate, providing exact results of the real and simulated environment. It is also designed in a way in that tracking is no longer probabilistically Slaved to the map-making procedure. It is done separately using coarse-to-fine robust estimator approach [20].

3.4. Smoke and fire detection base on video processing

Frequent and detailed updates of the development of fire are essential for effective and safe firefighting. Since fire is typically inaccessible by ground vehicles, firefighters often have to enter a fire region with little knowledge of how and where the fire is propagating, placing their lives at risk [65]. UAVs are emerging as a promising solution for monitoring large areas which could capture high-resolution imagery and broadcast frequent updates to fire crews [66]. The ongoing research projects of NASA looking at tracking the growth of fires using Low-altitude, short-endurance UAVs [67]. Yuan et al. [66] used an effective path planning algorithm for UAVs tasked to monitor a forest fire in that EMBYR (Ecological Model for Burning in the Yellowstone Region) in Simulink was utilized to simulate the time-evolution of a forest fire.

Smoke is produced much quicker than other fire signs during the scenes of fire extension and development. The rapid detection of smoke at deficient levels can maximize the likelihood of successful fire suppression, rescue, and survivability. The mass concentration, volume fraction and size distribution of the smoke are recognized as key parameters for smoke detection. There are two types of flame detectors based on the measurement of the range of flame radiation: infra-red detector and ultra-violet detector. Infra-red detectors (IR) detect fires when a particular flame flicker is produced. Ultra-violet detectors (UV) recognize fires when any ultra-violet radiation produced by flaming

combustion is exposed. With the introduction of artificial intelligence techniques (computer-based technology), the effectiveness of fire detection technology is significantly improved. There are intelligent algorithms for fire detection, including cross-correlation, algorithmic comparison, neural networks, fuzzy reasoning and Hidden Markov Model [68]. In intelligent fire detection systems, data processing and decision making are carried out in a ground station. It allows the system to perform complex and advanced algorithms. Video-based Fire Detection (VFD) is another method that helps to reduce the detection time compared to the currently available sensors in both indoors and outdoors [69]. VFD using cameras can monitor volumes and do not have a transport delay. These systems can provide crucial information about the size and growth of the fire, direction of smoke propagation and Colour detection (RGB, HSI and HSV saturation) [12,13,70–74]. Various algorithms could be integrated to determine if a motion in a scene is due to smoke or an ordinary moving object. Background subtraction methods, temporal differencing and optical flow analysis [75–78]. When a fire is detected and confirmed, the fire suppression system could be activated, or fire information could be communicated directly to the appropriate fire department. Marbach et al. [79] adopted an image processing technique for automatic real-time fire detection in video images with the underlying algorithm is based on the temporal variation of fire intensity. A Gaussian-smoothed color histogram was introduced in [80] that was used to detect the fire-colored pixels and temporal variation of pixels in each frame. To optimize the detection performance, an erode operation and region growing method could be integrated [5]. The pixel intensity may vary due to global motion and fire flicker. Thus, the pixel-by-pixel intensity difference for non-fire color pixels, by first correcting for the temporal variation of non-fire pixels, it is possible by deciding which pixels are fire candidates using Colour, finding the average change in intensity of all non-fire candidate pixels and subtracting this average value from the pixel differences at each location. Healy et al. [6] proposed a real-time system for automatic fire detection using color video input. This system has significant advantages over traditional ultraviolet and infrared fire detectors. These advantages include improved detection, fewer false alarms, and additional descriptive information about fire location, size, and growth rate. Their algorithm is developed based on the spectral, spatial, and temporal properties of fire events. The algorithm in [80] creates a threshold Gaussian-smoothed color histogram for increased accuracy of training sequences. Töreyin et al. [81] adopted the spatial wavelet transforms and static camera monitoring system to analyze periodic behavior in smoke boundaries and convexity of smoke regions. Edges are considered important due to their sharpness that leads to a decrease in the high-frequency content of the image and producing local extrema in the wavelet domain. Töreyin et al. [81] illustrated that a decrease in values of local extrema is an indicator of smoke where a scene becomes greyish that leads to a decrease in chrominance values of pixels. Qi and Ebert [71] proposed the spatial difference analysis using a histogram based approach, which focuses on the standard deviation of the green color band. They found that the green color band is the most discriminative for recognizing the spatial color variation of flames. This can also be seen by analyzing the histograms: green values vary more than red and blue values. If the standard deviation of the green color exceeds 50, the region is labeled as candidate flame. For smoke detection, this technique is not always applicable, because smoke regions often do not show as high spatial color variation and can cause false detections. Temporal Fourier analysis is an alternative approach to detect flickering flames in that any increase in Fourier domain energy between $[5-10]$ Hz is an indicator of flames. To extract fire features from video sequences we have adopted temporally extended normalized covariance descriptors (TENCD) by Verstockt [82]. TENCD are designed to describe spatiotemporal video blocks in that $I(i, j, n)$ is the intensity of (i, j) th pixel of the nth image frame of a spatiotemporal block in video. Some property parameters are defined to form a covariance matrix representing spatial information [69]. The video is divided into blocks of size $10 \times 10 \times F_{rate}$ where F_{rate} is the video's frame rate. To increase the efficiency of computations, the normalized covariance parameters are calculated for pixels with non-zero values of the mask defined for blocks [82].

$$\Phi(i, j, n) = \begin{cases} 1 & \text{for } M(i, j, n) = 1 \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

where $M(\cdots, \cdots, n)$ is the binary mask obtained from color detection and moving object detection algorithms. Property parameters are used for each pixel satisfying the color condition. To reduce the computational cost, the normalized covariance values of the pixel property vectors are computed separately. During the implementation of the correlation method, the first derivative of the image is computed by filtering the image with $[1 \quad 0 \quad 1]$ and the second derivative is found by filtering the image with $[1 \quad -2 \quad 1]$ filters, respectively. The fire and smoke detection algorithm based on the pixels and frame-by-frame basis. It was developed based on RGB (Red, Green, Blue), HSV (Hue, Saturation, Value) and YCbCr (non-linear RGB or Luminance, Chrominance) colour space [6,71,83] as well as motion information from video sequences streamed via mini Arducam.

3.5. Mixed reality simulation

Mixed Reality (MR) creates a space in which both physical and virtual elements co-exist [84–90]. This enables elements in one world to react directly to elements in the other world via direct and real-time data communication. The quadrotors performance and examination in real-world could be a cumbersome task and limited to specific scenarios. As such, we proposed a system that adopts MR to enable the physical and virtual world to communicate in a real-time. Using the Unity game engine to provide a safe and simple testbed with the real dimensions of objects, quadrotor and environmental measurements in SI units (meter and seconds). The data are transferred to the real world in the real-time.

We have neglected the wind effects in both virtual and physical due to the high precision pressure sensor (LPS25H) of the Crazyflie. The system allows a gradual transition of virtual training into the physical system in robotics or any other fields. We tested our proposed system in a fire detection scenario. The data are collected and forwarded to the ground station to generate the SLAM and map of the environment. We used the bag-of-words technique to convert images taken from both physical and virtual camera into sparse numerical vectors and to create an image database. The Unity and ROS systems are interfaced using a yaml-based[5] communication, Ros-bridge and HTTP protocol [91]. Ros-bridge enabled data streaming and communication between two operations. We mapped messages to events which are processed as part of the basic rendering loop in the visualized system to achieve real-time interface. The target (fire and smoke) is simulated in Unity which is visualized by the virtual camera and IMU data and communicated to the real quadrotor for teleoperation. The real quadrotor imitates the virtual drone's behavior as it detects fire and smoke (in the virtual scene) while the SLAMs are generated independently. Our system tracks the physical quadrotor's sensor pose, velocity, and IMU data to compare with the received data at frame-rate. This enables us to achieve reliable poses to match, and project mapped points to the key points on the frame. Each frame j is optimized by minimizing the feature reprojection error of all matched points and IMU errors. After a map update the IMU error is calculated from:

$$E_{IMU} = \rho\left([e_R^T e_v^T e_R^T] \sum_I [e_R^T e_v^T e_R^T]^T\right) + \rho\left(e_b^T \sum_R e_b\right) \tag{9}$$

Where Σ_I is the information matrix of the preintegration and Σ_R of

[5] http://www.yaml.org/.

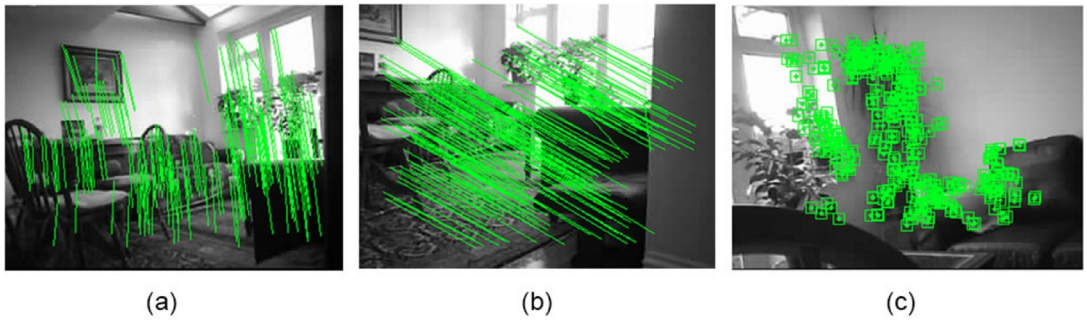

Fig. 3. SLAM initialization in an indoor environment at (a) 0.03 s, (b) 0.06 s and (c) 0.09 s to find edges, sharp corners and to match it with the bag-of-word visual vocabulary.

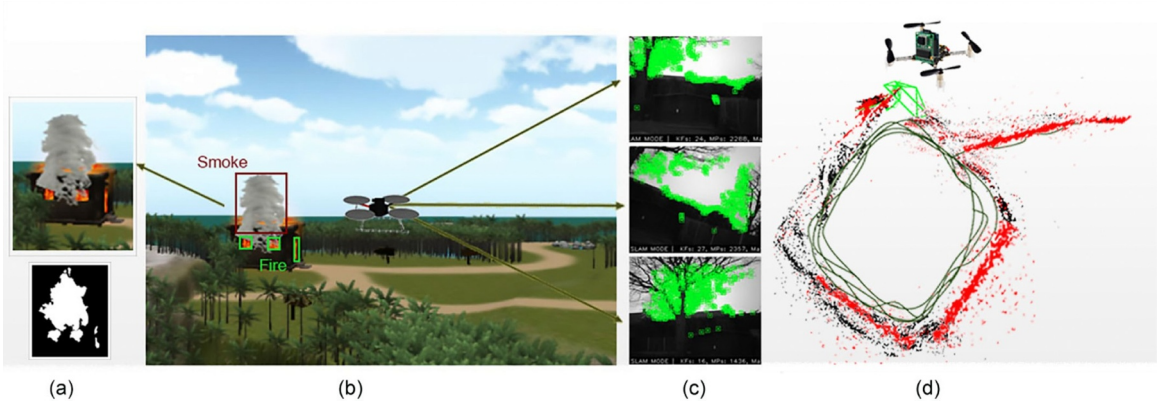

Fig. 4. The images collected in-indoor environment by the mini ArduCAM with the resolution of 320 × 240 pixels (a). Simulated environment using Unity game engine (b). The Keyframes generated using Place recognition algorithm (c). The SLAM for map reuse, loop closing, relocalization and scale factor between the quadrotor's inertia and Arducam coordinates generated while the quadrotor detected the fire and hovered around it (d).

Table 1
Crazyflie's pose RMSEs of the trajectories of autonomous flight in the physical and virtual world, with pose estimation errors in *mm* and attitude errors in degrees.

RMSEs	X	Y	Z	Roll	Pitch	Yaw
Physical world	21.7	38.5	13.2	0.91	0.89	1.03
Virtual world	19.4	31.7	10.9	1.43	1.37	1.81

the random bias, and ρ is the Huber robust cost function [29]. This is done using the optimization problem with the Gauss–Newton algorithm in g^2o [92]. The resulting estimation and Hessian matrix performed before next frame optimization. Adaptive relative bundle adjustment (ARBA) loop closure reduced the drift accumulated during exploration. The place recognition module and a matching keyframes procedure were conducted to match an already mapped and revisited area. Gyroscope and acceleration biases are estimated from the known orientation of two consecutive keyframes. The scale factor is utilized for transforming the coordinates between a camera and IMU data in the real and virtual world. Our results demonstrated that the simulation in Unity provided the flexibility of deploying the data in the ROS platform to use it for teleoperation in the real-time. The SLAM minimized the difference between IMU combination and relative orientation for all pairs of consecutive keyframes and pre-integrating measures. The system enabled us to blend MR intercommunication suit to test our SLAM algorithm as well.

4. Experimental evaluation

The operating system was set up on the ground-based station and used to communicate and exchange information via the WiFi ad-hocs to two host machines (a laptop and a desktop computer) running Ubuntu 16.04 LTS at the same time. Using a target of known circumstances allowed us to indicate a precise scale to the estimated map, motion and initialize features mainly due to the lack of depth information of our cameras. The quadrotor's position is indicated with a vector of (*x, y, z*) that describes how far an object translated along the axis relative to the origin. Similarly, orientation is specified via a vector of three numbers (*roll, pitch, yaw*) that describes how far an object has rotated relative to each axis where the position and orientation pair is known as 6D-pose. We built a node for learning a map of the environment and simultaneously estimated the platform's 2D pose at Arducam's frame rate. Visual odometry evaluated a continuous camera trajectory by examining the changes motion induce on the images, and the locations of obstacles in each frame are utilized to construct a global map. The loop closing technique was utilized for revisited locations since, without loop closures the world is represented as an infinite corridor in which the quadrotor keeps exploring new areas continuously. Fig. 3 shows three images of the camera's regular tracking performance for SLAM at different times which is followed by the image matching using the bag-of-words system. The trajectory poses are registered to its ground truth counterpart (information provided by direct observation). The localization performed for the corresponding trajectory to compute each frame's pose in the coordinate system of the root frame. The total trajectory recording error calculated as the average Euclidean distance between the ground truth and the localized frames. We implemented the virtual environment to include the live video for fire detection, where the SLAMs are generated by both cameras (real and virtual). The virtual Crazyflie's trajectory in the virtual environment is compared to its equivalent in the physical world. Tracking was maintained throughout several minutes and frames as new features generated by SLAM. The errors are minimized in the SLAM as uncertainties in the mapped features became small. The fire detection became stable and

locked the map into a drift-free state. In general terms, the algorithm gave a robust real-time performance within a room with relatively few obstacles and constraints and long time periods of localization. The root-mean-square errors (RMSEs) of the pose estimates showed that there is a potential scale drift in the SLAMs in our systems. The real camera calibration errors with the virtual one happened due to the ground truth data since it is difficult to set the tracking operation coordinate frame to coincide with the virtual coordinate frame. Table 1 outlines the Crazyflie's pose RMSEs of the trajectories of autonomous flight in the physical and virtual environment, with position errors in mm and orientation errors in degrees. Tracking proceeded throughout several minutes and frames with SLAM initializing new features as needed. Fig. 4(a) and Fig. 4(b) shows the simulated environment in the Unity game engine with the masks created. Fig. 4(c) illustrates the Keyframes generated by the place recognition algorithm. Fig. 4(d) plots the SLAM for map reuse, loop closing, relocalization and scale factor between the quadrotor's inertia and Arducam coordinates. It overlaps the maps created in the virtual and physical environment as the UAV maneuvered.

5. Conclusion

SLAM system was implemented that utilizes feature measurements from the onboard sensors. It demonstrated the efficiency of the method through the open source quadrotor with a camera and IMU to navigate autonomously toward a hut in the fire in a virtual world where the physical drone imitates its behavior in the real environment. The visual data and SLAMs are taken from virtual and physical cameras within the virtual and physical world. The experimental data prove that the proposed method is efficient and resistant to tracking failure. It provided the possibility to test the algorithm in a mixed reality environment. Long term poses tracking and IMU data adjusted the metric scale. Adaptive-Relative BA loop closing method was Integrated to form a full SLAM system for navigating the MAV in any size environments. The virtual and physical quadrotors are synchronized in that video streaming for fire detection happened inside the virtual world. Too many loops in SLAM was solved with the pose-graph optimization algorithms. Instead of explaining the full SLAM problem, a set of relative pose constraints was integrated for optimization. The future work will look into using drone flocks in a range of applications.

References

[1] D. Floreano, R.J. Wood, Science, technology and the future of small autonomous drones, Nature 521 (7553) (2015) 460–466.
[2] M. Hassanalian, A. Abdelkefi, Classifications, applications, and design challenges of drones: a review, Prog. Aerosp. Sci. (2017).
[3] R.A. Zahawi, J.P. Dandois, K.D. Holl, D. Nadwodny, J.L. Reid, E.C. Ellis, Using lightweight unmanned aerial vehicles to monitor tropical forest recovery, Biol. Conserv. 186 (2015) 287–295.
[4] D. Kim, Y.-F. Wang, Smoke detection in video, Computer Science and Information Engineering, 2009 WRI World Congress on, 5 IEEE, 2009, pp. 759–763.
[5] W. Phillips Iii, M. Shah, N. da Vitoria Lobo, Flame recognition in video, Pattern Recognit. Lett. 23 (1–3) (2002) 319–327.
[6] G. Healey, D. Slater, T. Lin, B. Drda, A.D. Goedeke, A system for real-time fire detection, CVPR, 93 (1993), pp. 15–17.
[7] Z. Xiong, R. Caballero, H. Wang, A.M. Finn, M.A. Lelic, P.-Y. Peng, Video-based smoke detection: possibilities, techniques, and challenges, IFPA, fire suppression and detection research and applications technical working conference (SUPDET), Orlando, FL, (2007).
[8] A. Kushleyev, D. Mellinger, C. Powers, V. Kumar, Towards a swarm of agile micro quadrotors, Auton. Robots 35 (4) (2013) 287–300.
[9] Z. Liu, A.K. Kim, Review of recent developments in fire detection technologies, J. Fire Prot. Eng. 13 (2) (2003) 129–151.
[10] B. Jiang, Y. Lu, X. Li, L. Lin, Towards a solid solution of real-time fire and flame detection, Multim. Tools Appl. 74 (3) (2015) 689–705.
[11] G. Pajares, Overview and current status of remote sensing applications based on unmanned aerial vehicles (UAVS), Photogramm. Eng. Remote Sens. 81 (4) (2015) 281–330.
[12] B.U. Töreyin, Y. Dedeoğlu, U. Güdükbay, A.E. Cetin, Computer vision based method for real-time fire and flame detection, Pattern Recognit. Lett. 27 (1) (2006) 49–58.
[13] B.C. Ko, K.-H. Cheong, J.-Y. Nam, Fire detection based on vision sensor and support vector machines, Fire Safety J. 44 (3) (2009) 322–329.
[14] Q. Bu, F. Wan, Z. Xie, Q. Ren, J. Zhang, S. Liu, General simulation platform for vision based UAV testing, Information and Automation, 2015 IEEE International Conference on, IEEE, 2015, pp. 2512–2516.
[15] J. Meyer, A. Sendobry, S. Kohlbrecher, U. Klingauf, O. Von Stryk, Comprehensive simulation of quadrotor UAVS using ROSand gazebo, International conference on simulation, modeling, and programming for autonomous robots, Springer, 2012, pp. 400–411.
[16] M. Sokolov, R. Lavrenov, A. Gabdullin, I. Afanasyev, E. Magid, 3d modelling and simulation of a crawler robot in ros/gazebo, Proceedings of the 4th International Conference on Control, Mechatronics and Automation, ACM, 2016, pp. 61–65.
[17] P.-J. Bristeau, F. Callou, D. Vissiere, N. Petit, The navigation and control technology inside the AR drone micro UAV, IFAC Proc. Vol. 44 (1) (2011) 1477–1484.
[18] A.J. Davison, Real-time simultaneous localisation and mapping with a single camera, null, IEEE, 2003, p. 1403.
[19] E. Eade, T. Drummond, Monocular slam as a graph of coalesced observations, Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, IEEE, 2007, pp. 1–8.
[20] G. Klein, D. Murray, Parallel tracking and mapping for small ar workspaces, Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on, IEEE, 2007, pp. 225–234.
[21] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, P. Sayd, Real time localization and 3d reconstruction, Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, 1 IEEE, 2006, pp. 363–370.
[22] D. Nistér, O. Naroditsky, J. Bergen, Visual odometry, Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, 1 Ieee, 2004. I–I
[23] G. Younes, D. Asmar, E. Shammas, J. Zelek, Keyframe-based monocular slam: design, survey, and future directions, Robot. Auton. Syst. 98 (2017) 67–88.
[24] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, J. Leonard, Past, present, and future of simultaneous localization and mapping: towards the robust-perception age, IEEE Trans. Robot. 32 (6) (2016) 1309–1332.
[25] B. Siciliano, O. Khatib, Springer Handbook of Robotics, Springer, 2016.
[26] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics, MIT press, 2005.
[27] C. Forster, L. Carlone, F. Dellaert, D. Scaramuzza, IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation, Georgia Institute of Technology, 2015.
[28] N. Keivan, A. Patron-Perez, G. Sibley, Asynchronous adaptive conditioning for visual-inertial slam, Experimental Robotics, Springer, 2016, pp. 309–321.
[29] R. Mur-Artal, J.D. Tardós, Visual-inertial monocular slam with map reuse, IEEE Robot. Autom. Lett. 2 (2) (2017) 796–803.
[30] H. Strasdat, J. Montiel, A.J. Davison, Real-time monocular slam: why filter? Robotics and Automation (ICRA), 2010 IEEE International Conference on, IEEE, 2010, pp. 2657–2664.
[31] D. Gálvez-López, J.D. Tardos, Bags of binary words for fast place recognition in image sequences, IEEE Trans. Robot. 28 (5) (2012) 1188–1197.
[32] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge university press, 2003.
[33] M. Kaess, A. Ranganathan, F. Dellaert, Isam: incremental smoothing and mapping, IEEE Trans. Robot. 24 (6) (2008) 1365–1378.
[34] K. Konolige, W. Garage, Sparse sparse bundle adjustment, BMVC, 10, Citeseer, 2010, pp. 102.1–102.11 http://www.willowgarage.com/sites/default/files/ssba.pdf.
[35] E. Olson, J. Leonard, S. Teller, Fast iterative alignment of pose graphs with poor initial estimates, Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on, IEEE, 2006, pp. 2262–2269.
[36] D. Sibley, C. Mei, I.D. Reid, P. Newman, Adaptive relative bundle adjustment. Robotics: science and systems, 32 (2009), p. 33.
[37] G. Grisetti, C. Stachniss, S. Grzonka, W. Burgard, A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. Robotics: Science and Systems, 3 (2007), p. 9.
[38] B. Triggs, P.F. McLauchlan, R.I. Hartley, A.W. Fitzgibbon, Bundle adjustment modern synthesis, International workshop on vision algorithms, Springer, 1999, pp. 298–372.
[39] G. Sibley, Relative bundle adjustment, Technical Report, Department of Engineering Science, Oxford University, 2009.
[40] C. Forster, L. Carlone, F. Dellaert, D. Scaramuzza, On-manifold preintegration for real-time visual-inertial odometry, IEEE Trans. Robot. 33 (1) (2017) 1–21.
[41] A.S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, N. Roy, Visual odometry and mapping for autonomous flight using an rgb-d camera, Robotics Research, Springer, 2017, pp. 235–252.
[42] J. Zhang, S. Singh, Visual-lidar odometry and mapping: Low-drift, robust, and fast, Robotics and Automation (ICRA), 2015 IEEE International Conference on, IEEE, 2015, pp. 2174–2181.
[43] M.C. Deans, M. Hebert, Bearings-only localization and mapping, Carnegie Mellon University, The Robotics Institute, 2005 Ph.D. thesis.
[44] A. Bry, C. Richter, A. Bachrach, N. Roy, Aggressive flight of fixed-wing and quadrotor aircraft in dense indoor environments, Int. J. Robot. Res. 34 (7) (2015) 969–1002.
[45] C. Richter, A. Bry, N. Roy, Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments, Robotics Research, Springer, 2016, pp. 649–666.
[46] S.O. Madgwick, A.J. Harrison, R. Vaidyanathan, Estimation of IMU and marg orientation using a gradient descent algorithm, Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on, IEEE, 2011, pp. 1–7.
[47] O. Amidi, T. Kanade, K. Fujita, A visual odometer for autonomous helicopter flight,

Robot. Auton. Syst. 28 (2–3) (1999) 185–193.
[48] M.A. Turk, A.P. Pentland, Face recognition using eigenfaces, Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on, IEEE, 1991, pp. 586–591.
[49] J. Heinly, E. Dunn, J.-M. Frahm, Comparative evaluation of binary features, Computer Vision–ECCV 2012, Springer, 2012, pp. 759–773.
[50] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, Speeded-up robust features (surf), Comput. Vision Image Underst. 110 (3) (2008) 346–359.
[51] S. Leutenegger, M. Chli, R.Y. Siegwart, Brisk: binary robust invariant scalable keypoints, Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE, 2011, pp. 2548–2555.
[52] T. Lupton, S. Sukkarieh, Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions, IEEE Trans. Robot. 28 (1) (2012) 61–76.
[53] V. Indelman, S. Williams, M. Kaess, F. Dellaert, Information fusion in navigation systems via factor graph based incremental smoothing, Robot. Auton. Syst. 61 (8) (2013) 721–738.
[54] Y. Wang, G.S. Chirikjian, Nonparametric second-order theory of error propagation on motion groups, Int. J. Robot. Res. 27 (11–12) (2008) 1258–1273.
[55] P. Furgale, J. Rehder, R. Siegwart, Unified temporal and spatial calibration for multi-sensor systems, Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on, IEEE, 2013, pp. 1280–1286.
[56] H. Bay, T. Tuytelaars, L. Van Gool, Surf: speeded up robust features, Comput. Vision–ECCV 2006 (2006) 404–417.
[57] D.G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vision 60 (2) (2004) 91–110.
[58] M. Calonder, V. Lepetit, C. Strecha, P. Fua, Brief: binary robust independent elementary features, Comput. Vision–ECCV 2010 (2010) 778–792.
[59] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, Orb: an efficient alternative to sift or surf, Computer Vision (ICCV), 2011 IEEE international conference on, IEEE, 2011, pp. 2564–2571.
[60] P.L. Rosin, Measuring corner properties, Comput. Vision Image Underst. 73 (2) (1999) 291–307.
[61] R. Mur-Artal, J.D. Tardós, ORB-SLAM2: an open-source slam system for monocular, stereo, and rgb-d cameras, IEEE Trans. Robot. 33 (5) (2017) 1255–1262.
[62] J. Sivic, A. Zisserman, Video google: a text retrieval approach to object matching in videos, null, IEEE, 2003, p. 1470.
[63] D. Nister, H. Stewenius, Scalable recognition with a vocabulary tree, Computer vision and pattern recognition, 2006 IEEE computer society conference on, 2 IEEE, 2006, pp. 2161–2168.
[64] M.A. Fischler, R.C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM 24 (6) (1981) 381–395.
[65] D.W. Casbeer, R.W. Beard, T.W. McLain, S.-M. Li, R.K. Mehra, Forest Fire Monitoring with Multiple Small UAVS, (2005).
[66] C. Yuan, Y. Zhang, Z. Liu, A survey on technologies for automatic forest fire monitoring, detection, and fighting using unmanned aerial vehicles and remote sensing techniques, Can. J. forest Res. 45 (7) (2015) 783–792.
[67] White paper on UAV over-the-horizon disaster management demonstration projects, 2000, (http://geo.arc.nasa.gov/sge/UAVFiRE/whitepaper.html). Accessed: 2018.
[68] H.-C. Muller, et al., A new approach to fire detection algorithms based on the hidden Markov model, NIST Special Publication SP (2001) 129–138.
[69] A.E. Çetin, K. Dimitropoulos, B. Gouverneur, N. Grammalidis, O. Günay, Y.H. Habibolu, B.U. Töreyin, S. Verstockt, Video fire detection–review, Digital Signal Process. 23 (6) (2013) 1827–1843.
[70] T.-H. Chen, P.-H. Wu, Y.-C. Chiou, An early fire-detection method based on image processing, Image Processing, 2004. ICIP'04. 2004 International Conference on, 3 IEEE, 2004, pp. 1707–1710.
[71] X. Qi, J. Ebert, A computer vision based method for fire detection in color videos, Int. J. Imaging 2 (S09) (2009) 22–34.
[72] J. Chen, Y. He, J. Wang, Multi-feature fusion based fast video flame detection, Build. Environ. 45 (5) (2010) 1113–1122.
[73] J. Gubbi, S. Marusic, M. Palaniswami, Smoke detection in video using wavelets and support vector machines, Fire Safety J. 44 (8) (2009) 1110–1115.
[74] O. Günay, K. Taşdemir, B.U. Töreyin, A.E. Çetin, Fire detection in video using lms based active learning, Fire Technol. 46 (3) (2010) 551–577.
[75] B. Ko, K.-H. Cheong, J.-Y. Nam, Early fire detection algorithm based on irregular patterns of flames and hierarchical Bayesian networks, Fire Safety J. 45 (4) (2010) 262–270.
[76] P. Piccinini, S. Calderara, R. Cucchiara, Reliable smoke detection in the domains of image energy and color, Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on, IEEE, 2008, pp. 1376–1379.
[77] B. Lee, D. Han, Real-time fire detection using camera sequence image in tunnel environment, International Conference on Intelligent Computing, Springer, 2007, pp. 1209–1220.
[78] I. Kolesov, P. Karasev, A. Tannenbaum, E. Haber, Fire and smoke detection in video with optimal mass transport based optical flow and neural networks, Image Processing (ICIP), 2010 17th IEEE International Conference on, IEEE, 2010, pp. 761–764.
[79] G. Marbach, M. Loepfe, T. Brupbacher, An image processing technique for fire detection in video images, Fire Safety J. 41 (4) (2006) 285–289.
[80] R. Kjeldsen, J. Kender, Finding skin in color images, Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference on, IEEE, 1996, pp. 312–317.
[81] B.U. Töreyin, Y. Dedeoğlu, A.E. Cetin, Wavelet based real-time smoke detection in video, Signal Processing Conference, 2005 13th European, IEEE, 2005, pp. 1–4.
[82] S. Verstockt, Multi-modal video analysis for early fire detection, Ghent University, 2011 Ph.D. thesis.
[83] M.J. Swain, D.H. Ballard, Color indexing, Int. J. Comput. Vision 7 (1) (1991) 11–32.
[84] F.L. Peña, P. Caamaño, G. Varela, F. Orjales, A. Deibe, Setting up a mixed reality simulator for using teams of autonomous UAVS in air pollution monitoring, Int. J. Sustain. Dev. Plan. 11 (4) (2016) 616–626.
[85] I.Y.-H. Chen, B. MacDonald, B. Wünsche, Evaluating the effectiveness of mixed reality simulations for developing UAV systems, International Conference on Simulation, Modeling, and Programming for Autonomous Robots, Springer, 2012, pp. 388–399.
[86] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, B. MacIntyre, Recent advances in augmented reality, IEEE Comput. Graphics Appl. 21 (6) (2001) 34–47.
[87] F. Ghiringhelli, J. Guzzi, G.A. Di Caro, V. Caglioti, L.M. Gambardella, A. Giusti, Interactive augmented reality for understanding and analyzing multi-robot systems, Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on, IEEE, 2014, pp. 1195–1201.
[88] J.T. Hing, J. Menda, K. Izzetoglu, P.Y. Oh, An indoor study to evaluate a mixed-reality interface for unmanned aerial vehicle operations in near earth environments, Proceedings of the 10th Performance Metrics for Intelligent Systems Workshop, ACM, 2010, pp. 214–221.
[89] P. Milgram, H. Takemura, A. Utsumi, F. Kishino, Augmented reality: a class of displays on the reality-virtuality continuum, Telemanipulator and telepresence technologies, 2351 International Society for Optics and Photonics, 1995, pp. 282–293.
[90] Z. Pan, A.D. Cheok, H. Yang, J. Zhu, J. Shi, Virtual reality and mixed reality for virtual learning environments, Computers & Graphics 30 (1) (2006) 20–28.
[91] D. Crockford, The Application/json Media Type for Javascript Object Notation (json), (2006).
[92] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, W. Burgard, g 2 o: a general framework for graph optimization, Robotics and Automation (ICRA), 2011 IEEE International Conference on, IEEE, 2011, pp. 3607–3613.