2016 IEEE Intelligent Vehicles Symposium (IV)
Gothenburg, Sweden, June 19-22, 2016

# Monocular Vision-Based Obstacle Detection/Avoidance for Unmanned Aerial Vehicles*

Abdulla Al-Kaff[1], Qinggang Meng[2], David Martín[1], Arturo de la Escalera[1] and José María Armingol[1]

***Abstract*— Robust real-time obstacle detection/avoidance is a challenging problem especially for micro and small aerial vehicles due to the limited number of the on-board sensors due to the battery constraint and low payload.Usually lightweight sensors such as CMOS camera are the best choice comparing with laser or radar sensors. For real-time applications, most studies focus on using stereo cameras to reconstruct a 3D model of the obstacles or to estimate their depth. Instead, in this paper, a method that mimics the human behavior of detecting the state of the approaching obstacles using single camera is proposed. During the flight, this method is able to detect the changes of the size area of the obstacles. First, the method detects the feature points of the obstacles, and then extracts the obstacles that has probability of getting close. In addition, by comparing the changes in the area ratios of the obstacle in the image sequence, the method can decide if it is obstacle or not. Finally, by estimating the obstacle 2D position in the image and combining with the tracked waypoints, the UAV can take the action of avoidance.**

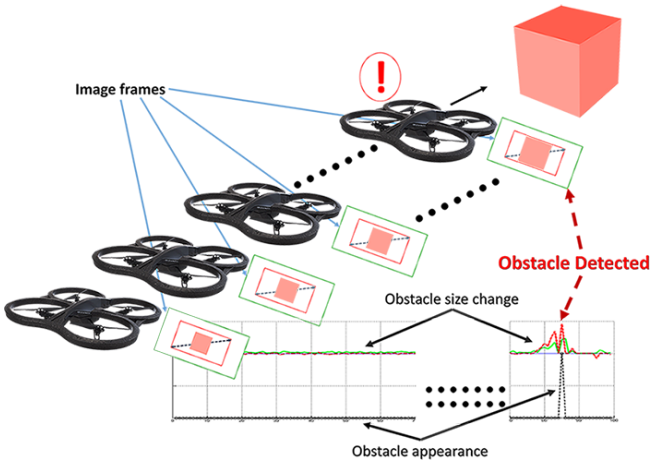


Fig. 1. The concept of approaching obstacle detection

## I. INTRODUCTION

During the last decade, with the developments in micro-electronics and the increase of computing efficiency, the use of Unmanned Aerial Vehicles (UAVs) has no longer restricted to the military purposes only. Recently, with the advent of small and micro aerial vehicles, the requirements of operations and applications in low altitudes are increased.

With the current technology, and the variety/complexity of the tasks, modern UAVs aim at higher levels of autonomy and performing flight stabilization. For the autonomous UAVs, the ability of detection and avoidance of obstacles with high level of accuracy is a challenging problem.

The difficulty appears because of the size of UAVs is getting smaller and the weight is getting lighter. Therefore, with these properties, UAVs are not able to carry heavy sensors such as laser [1], [2], [3] or radar [4]. Hence, the suitable solution is to use the on-board cameras due to its advantage of lightweight and low power consumption.

In addition to its lightweight and low power consumption, the cameras provide very rich information of the environment. Therefore, they are considered as important sensors mounted on the small/micro UAVs.

In vision-based navigation systems, different approaches were presented to solve the problem of obstacle detection/avoidance. Approaches as [5], [6], [7], [8] built a 3D model of the obstacle in the environment. However, they are computationally expensive. While, a bio-inspired (insect/human like) approaches which estimate the presence of the obstacle efficiently without calculating the 3D model, such as using optical flow [9], [10], [11] or perspective cues [12], [13], [14].

In this paper, a bio-inspired approach using a monocular camera is presented to mimic the human behavior of obstacle detection and avoidance applied on UAVs. In which the presence of approaching obstacles is estimated from its size expansion ratios instead of building 3D model of them as shown in Figure 1.

The novelty in this paper is based on the change of the size property of the detected features points, combined with the size ratio of the constructed convex of these detected points from two consecutive frames in the drone motion.

The remainder of this paper is organized as follows; section II introduces the state-of-the-art work related to obstacle detection/avoidance approaches, followed by the proposed obstacle detection algorithm in section III. Section IV presents the avoidance algorithm, then section V discusses the experimental results. Finally, in section VI conclusion is summarized.

## II. RELATED WORK

Obstacle detection/avoidance plays an important role in any autonomous navigation system. Different works were presented to solve this challenging process especially for vision-based systems.

Croon *et al.* [15], presented an approach based on the

[1]Abdulla Al-Kaff, David Martín, Arturo de la Escalera and José María Armingol are with Intelligent Systems Lab, University Carlos III of Madrid C/Butarque 15, Leganés (28911), Madrid, Spain {akaff, dmgomez, escalera, armingol}@ing.uc3m.es

[2]Qinggang Meng is with the Department of Computer Science, Loughborough University, Leicestershire LE11 3TU, UK q.meng@lboro.ac.uk

texture/color variation cue to detect obstacles for indoor environments. However, this approach works with detailed textures.

Working with HaybridMAVs, Green *et al.* proposed an optical flow based approach for lateral collision avoidance mimicking the biological flying insects [16].

In [17], SIFT descriptor and multi-scale-oriented-patches (MOPS) are combined to show 3D information of the obstacles. In which, the edges and corners of the object are extracted by MOPS, then SIFT is used to detect the internal outline information.

Bills *et al.* [14] proposed an approach for indoor environments with a uniform structure characteristics. In this work, Hough Transform is used to detect the edges that are used to classify the essence of the scene based on a trained classifier. however, their experiments was limited to corridors and stairs areas.

In [18], the authors presented an approach of measuring the relative distance of the obstacle. In which, the camera position is estimated by using the Extended Kalman Filter ($EKF$) and a based pose estimation from the Inertial Measurement Unit ($IMU$). Then the 3D position of the obstacle can be calculated by projecting back the detected features of the obstacle from its images.

An expansion segmentation method was presented in [19]. In which a conditional Markov Random Field is used to distinguish if the frontal object is a collision or not. In addition an inertial system is used to estimate the collision time. However the experiments of this work was limited to simulations.

Another approach used feature detection in conjunction with template matching to detect the size expansions of the obstacles [20]. However, the experiments were limited on a like-tree obstacles and did not show results of other shapes.

Kim *et al.* presented a Block-Based Motion Estimation approach for moving obstacles (humans) [21]. in which the input image is divided in smaller blocks and comparing the motion in each block through consecutive images. However, this approach works well with large size obstacles (humans).

In addition, surveys of different approaches of modeling autonomous UAVs, navigation and collision avoidance methods are presented in [22] and [23].

## III. OBSTACLE DETECTION

The proposed obstacle detection algorithm tries to mimic the human behavior of detecting frontal obstacles while motion. In which, the state of the approaching obstacles is estimated instead of building 3D models or calculating the depth of the obstacle.

The novelty and the key of this algorithm is to estimate the size ratios of the approaching obstacles from the consecutive frames during the flight. This is achieved by estimating the change in the size of the feature points and the size of the convex constructed from these points. When the size ratios exceed certain values, it means that the obstacle can cause a danger to the drone as shown in Algorithm 1.

**Algorithm 1:** Obstacle Detection

**Input:** Image frames $F$
**Output:** Obstacle state $Obs$, Obstacle position $\tau$
**Define**: Current frame $F_t$, Previous frame $F_{t-1}$, Current workspace $ROI_t$, Previous workspace $ROI_{t-1}$, Current keypoints $KP_t[\,]$, Previous keypoints $KP_{t-1}[\,]$, Number of keypoints ($N$, $M$), $n \in N$, $m \in M$, $n = m$, $x \in n$, Matched points $pts$, Distance ratio $thresh$, Object of interest $Convex$
**begin**
  **while** *isFlying( )* **do**
    $F_{t-1} \leftarrow$ getNewFrame( )
    **if** *Obstacle Detection isActivated( )* **then**
      $F_t \leftarrow$ getNewFrame( )
      $(ROI_{t-1}, ROI_t) \leftarrow$ DefWrkspc($F_{t-1}$,$F_t$)
      $(KP_{t-1}(N), KP_t(M)) \leftarrow$ DetectKeypoints($F_{t-1}$, $F_t$)
      $(pts_{t-1}(n), pts_t(m)) \leftarrow$ MatchSymKeypoints($KP_{t-1}(N)$, $KP_t(M)$, $thresh$)
      $(pts_{t-1}(x), pts_t(x)) \iff$ size($pts_t(m)$) > size($pts_{t-1}(n)$)
      $(Convex_1, Convex_2) \leftarrow$ Create($pts_{t-1}(x)$, $pts_t(x)$)
      **if** *AreaScale*($Convex_2 : Convex_1$) $\geq$ *1.7 and SizeScale*($pts_t(x) : pts_{t-1}(x) \geq$ *1.2)* **then**
        $Obs \leftarrow$ true
        $\tau \leftarrow$ CalcPos($Obs$)
        **goto** Algorithm 2
    $F_{t-1} \leftarrow F_t$

### A. Feature Detection and Description

In this step, a patch of $62^o$ diagonal field of view ($FOV$) is taken from the whole image to be processed as shown in Figure 2. That is because any detected obstacle out of this area will not cause any danger to the drone. Furthermore, processing the $62^o$ patch instead of the whole $92^o$ image, leads to a significant minimizing in computational time.

Due to flying in unknown environments structures, the captured frames are affected with different conditions (i.e. illumination, noise, etc.). However, the keypoints need to be extracted accurately even under these conditions. Therefore SIFT [24] detector/descriptor is used; because of the ability to identify and localize accurately the feature points even under different image condition specially scale and rotation properties.

All keypoints are detected and extracted from the consecutive frames as shown in Figure 3, then a vector of position $(x,y)$ and size $(s)$ is obtained for each keypoint. Applying Brute-Force algorithm to match the keypoints from the two frames, and only the points (found in both frames) are returned. For more accuracy, the matched keypoints are filtered by eliminating the ones with a distance ratio bigger

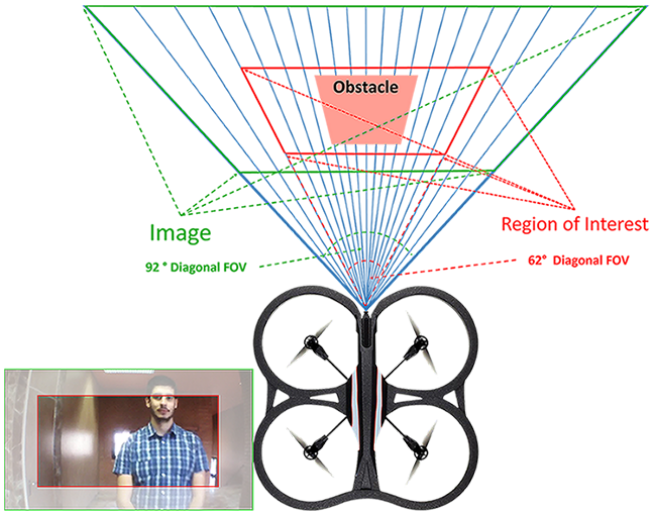


Fig. 2. Define $62^o$ patch from the whole $92^o$ image FOV

than a threshold value as shown as follows:

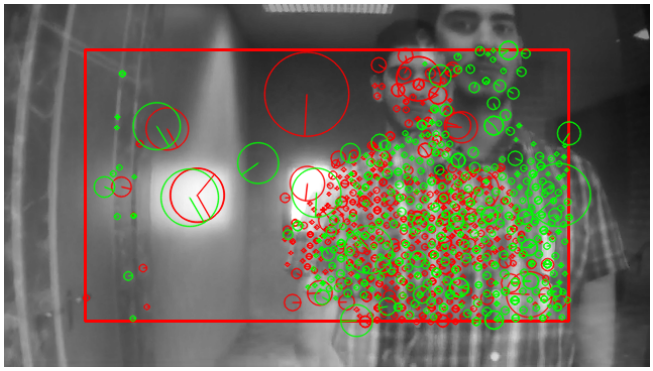

Fig. 3. Detection all keypoints from two consecutive frames

$$mkp(n) = \begin{cases} (x,y,s), & distratio \leq 0.28 \\ 0, & otherwise \end{cases} \quad \forall n \in K \quad (1)$$

where, $mkp$ is the filtered matched keypoint and $K$ is the total number of matched keypoints.

After that, the keypoints obtained by equation 1 are compared from the second to the first frame, and then the algorithm returned the matched keypoints if and only if its size is growing as shown in Figure 4.

$$mkp(i) = \begin{cases} (x,y,s), & Size(mkp_2(i) > mkp_1(i)) \\ 0, & otherwise \end{cases} \quad \forall i \in n \quad (2)$$

### B. Object of Interest

The next step of the detection algorithm is to determine if there is a probability to find a frontal obstacle or not. Hence, from the extracted and filtered keypoints from equation 2, an Object of Interest (OOI) is created around these keypoints in both frames by creating a convex hull of the corresponding points as shown in Figure 5.

Fig. 4. Expanding Size of keypoints from the second frame to the first frame indicates

$$C = \sum_{i=1}^{N} \lambda_i mpk_i | (\forall i : \lambda_i \geq 0) \bigwedge \sum_{i=1}^{N} \lambda_i = 1 \quad (3)$$

where $\lambda_i$ ia a non-negative weight assigned to the keypoints $mpk_i \in N$.

To estimate the size change of the detected obstacles we consider each convex hull as an irregular polygon. So that, for a given C as a convex hull, the area of C can be calculated as follows:

$$C_{area} = \frac{1}{2}[(x_1y_2 + x_2y_3 + x_3y_4 + \cdots + x_ny_1) - (y_1x_2 + y_2x_3 + y_3x_4 + \cdots + y_nx_1)] \quad (4)$$

where $x_{(1:n)}$ and $y_{(1:n)}$ are vertices and $n$ is the number of sides of the polygon.

Finally, the size ratio of both the keypoints and the convex hull from the second to the first frame are calculated as follows:

$$ratio(mkp) = \frac{1}{N} \sum_{i=1}^{N} \frac{Size(mkp_2(i))}{Size(mkp_1(i))} \quad (5)$$

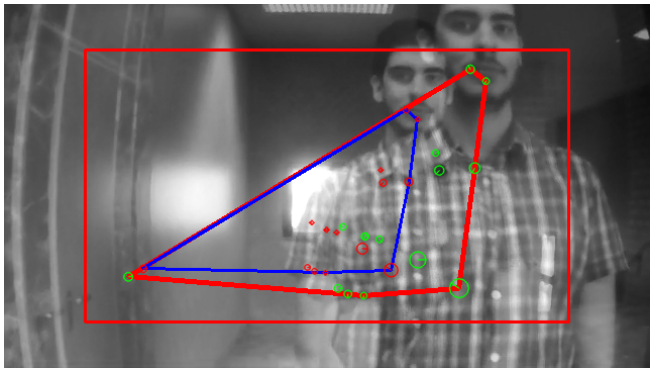

Fig. 5. Convex Hull construction from detected keypoints in both frames

$$ratio(C) = \frac{Size(C_2)}{Size(C_1)} \tag{6}$$

Then, the algorithm can obtain the state, if it is an approaching obstacle or not.

$$State = \begin{cases} 1, & ratio(mkp) \geq 1.2 \bigwedge ratio(C) \geq 1.7 \\ 0, & otherwise \end{cases} \tag{7}$$

Figure 6 shows obstacle detected by the monocular camera.

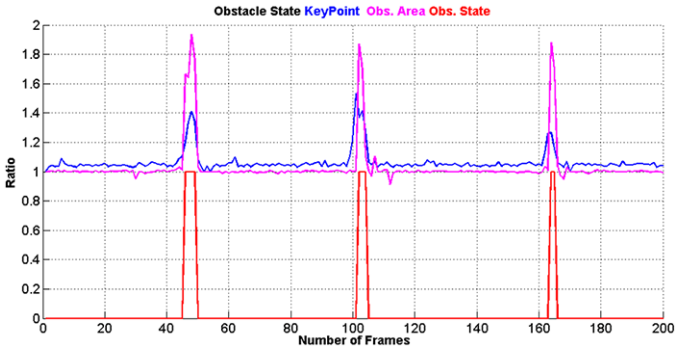


Fig. 6. Obstacle State: Blue: Keypoint size ratio, Magenta: Convex Area Ratio and Red: Obstacle State (0) not fount (1) found.

## IV. OBSTACLE AVOIDANCE

In this section, the combined mission of waypoint tracking and avoidance method is described. The geometrical problem is shown in Figure 7 where the avoidance technique is summarized in Algorithm 2.

To define the problem of waypoint tracking, let the drone $X$ flying at a velocity $V$, where $X = \begin{bmatrix} x_d & y_d & z_d \end{bmatrix}^T$ and $V = \begin{bmatrix} u_d & v_d & w_d \end{bmatrix}^T$ considering the drone flies at a constant velocity along its X-axis.

On the other hand, let a waypoint $WP = \begin{bmatrix} x_w & y_w & z_w \end{bmatrix}^T$, hence the waypoint is assumed to be tracked if $x_d = x_w \pm \mu_x$ is achieved when $y_d = y_w \pm \mu_y$ and $z_d = z_w \pm \mu_z$ is satisfied, where $\mu$ is the position tolerance.

Let a frontal obstacle $\tau = \begin{bmatrix} \tau_x & \tau_y \end{bmatrix}^T$ situated in the drone path and was detected by Algorithm 1.

First, the avoidance algorithm assumed a safety boundary surrounding the obstacle (260 *mm* vertically, 150 *mm* horizontally) as shown in Figure 7. Then, the algorithm reads the position of the predefined next waypoint and calculates the new waypoint out of the path (to avoid the obstacle), and send a control command (*velocity control*) to the drone for maneuvering according to the waypoint position as follows:

- Horizontal maneuver (Right or Left)

$$V = \kappa \frac{WP(n) + X}{2} + \begin{bmatrix} 0 & \pm 260 & 0 \end{bmatrix}^T \tag{8}$$

- Vertical maneuver (Top or bottom)

$$V = \kappa \frac{WP(n) + X}{2} + \begin{bmatrix} 0 & 0 & \pm 150 \end{bmatrix}^T \tag{9}$$

**Algorithm 2:** Obstacle Avoidance Algorithm

**Input:** Obstacle position $\tau$, Drone position $X$
**Output:** Navigation command $Nav(\phi, \theta, \psi, \vartheta)$
**Define**: Image zones ($A, B, C$ and $D$), Waypoint $WP$, Next waypoint $nWP$, Roll $\phi$, Pitch $\theta$, Yaw $\psi$, Vertical speed $\vartheta$, Control gains $K_{ph}$ and $K_{p\phi}$
**Control Commands**:
  $+\Delta H \leftarrow X(z)$ + 150 *mm*
  $-\Delta H \leftarrow X(z)$ - 150 *mm*
  $+\Delta Y \leftarrow X(y)$ + 260 *mm*
  $-\Delta Y \leftarrow X(y)$ - 260 *mm*
**begin**
  **while** *!atGoal* **do**
    **if** *AreaScale*$\leq$ *2* && *SizeScale*$\leq$ *1.5* **then**
      **if** $\tau \in A \,||\, B \,||\, (AB)$ **then**
        $\vartheta \leftarrow K_{ph}\ (-\Delta H$ - $X(z))$
      **else if** $\tau \in C \,||\, D \,||\, (CD)$ **then**
        $\vartheta \leftarrow K_{ph}\ (+\Delta H$ - $X(z))$
      **else if** $\tau \in A \,||\, C \,||\, (AC)$ **then**
        $\phi \leftarrow K_{p\phi}\ (+\Delta Y$ - $X(y))$
      **else if** $\tau \in B \,||\, D \,||\, (BD)$ **then**
        $\phi \leftarrow K_{p\phi}\ (-\Delta Y$ - $X(y))$
      **else if** $\tau \in (ABC)$ **then**
        $\vartheta \leftarrow K_{ph}\ (-\Delta H$ - $X(z))$
        $\phi \leftarrow K_{p\phi}\ (+\Delta Y$ - $X(y))$
      **else if** $\tau \in (ABD)$ **then**
        $\vartheta \leftarrow K_{ph}\ (-\Delta H$ - $X(z))$
        $\phi \leftarrow K_{p\phi}\ (-\Delta Y$ - $X(y))$
      **else if** $\tau \in (ACD)$ **then**
        $\vartheta \leftarrow K_{ph}\ (+\Delta H$ - $X(z))$
        $\phi \leftarrow K_{p\phi}\ (+\Delta Y$ - $X(y))$
      **else if** $\tau \in (BCD)$ **then**
        $\vartheta \leftarrow K_{ph}\ (+\Delta H$ - $X(z))$
        $\phi \leftarrow K_{p\phi}\ (-\Delta Y$ - $X(y))$
      **else if** $\tau \in (ABCD)$ **then**
        $nWP \leftarrow Read()$
        **if** $WP(y) > X(y)$ **then**
          $\phi \leftarrow K_{p\phi}(+\Delta Y$ - $X(y))$
        **else**
          $\phi \leftarrow K_{p\phi}(-\Delta Y$ - $X(y))$
      $X \leftarrow$ Nav($\phi$, $\theta$, $\psi$, $\vartheta$)
    **else**
      $X \leftarrow$ Nav(0, 0, 0, 0)
    **goto** Algorithm 1

where $WP(n)$ defines the new waypoint position and $\kappa$ is a control coefficient. Finally, by estimating the new drone position after avoidance, the algorithm recalculates the new waypoints in order the drone to be able to return back to its predefined path and activate the detection process.

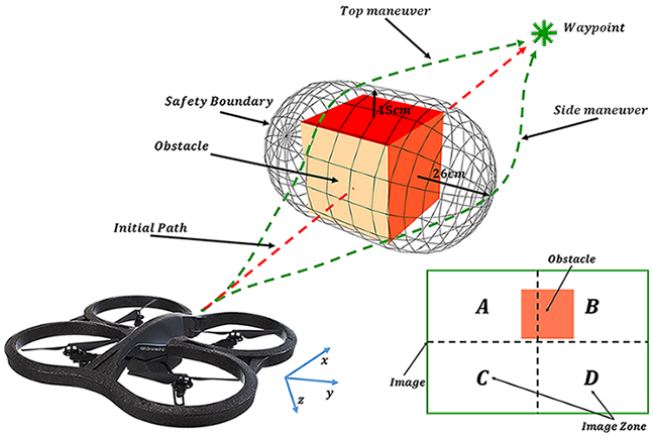


Fig. 7. Obstacle Avoidance Path

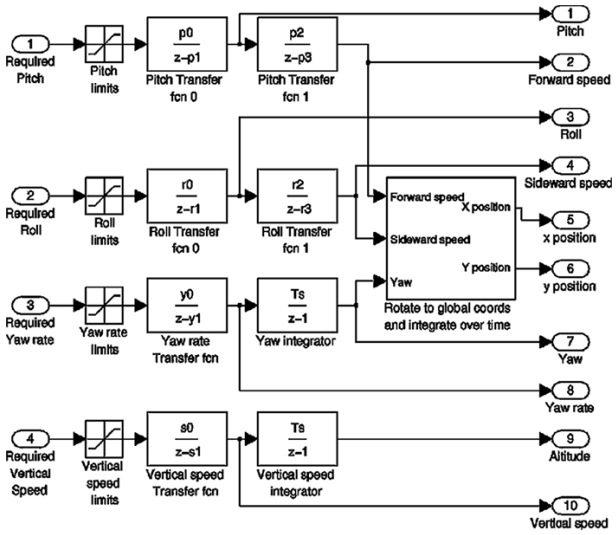


Fig. 8. AR-Drone internal control structure [25]

In the case if the *AreaScale* is greater than 2 and the keypoint *SizeScale* is greater than 1.5, a "*Hover*" command is sent to the drone. That is because if the ratios exceed these limits, this means that the obstacle is very close to the drone (less than 50 cm).

## V. EXPERIMENTAL RESULTS

The proposed algorithms have been tested and verified with data gathered from real flights. The processing in the ground station is performed in an Intel i7-3770 at 3.4 GHz CPU.

### A. Platform

The experiments have been performed with a Parrot AR.Drone 2.0 quadcopter [25]. One of the most important aspects in the avoidance phase is based on the robust control system for the drone, in which is necessary to know its dynamic model. However, to avoid the complexity in modeling, the control was applied on the systems internal control. Figure 8 shows the internal control of the drone which is governed by the inputs (roll, pitch, yaw angles, and vertical speed), therefore the implemented controller realize the drone actual position, orientation and velocity.

### B. Results

In the experiments, sixty real flights have been preformed with different types of obstacles (people, obstacles, pillars and walls), (statics and dynamic) of a total number of 285 obstacles. All the flight were in indoor environments, two different scenarios were tested to evaluate the performance of the proposed algorithms in both motion and stability.

The first scenario was a predefined straight flight from a starting point to an end point. Different types of obstacles were situated in the drone path. The goal of this scenario is to evaluate the accuracy and robustness in detecting and avoiding the obstacles in motion.

The second scenario was a hover flight. In which, the drone enters to the hover mode and different obstacle were approached to it. Once an obstacle is detected, the drone flights in the the opposite direction of the obstacle (Backward).

From the experiments, the algorithm is able to detect the obstacles with different sizes ($8500 \leq area \geq 200000$ in pixels units) at a distance between 90-120 *cm*. It is shown that the accuracy of the algorithm is more than 92.5%.

Table I summarized the accuracy of the detection algorithm. The table shows the total number of the obstacles either situated in the drone path (*first scenario*) or moving towards the drone (*second scenario*), the number of the detected obstacles and the number of fails.

Two main reasons for the fail of detection; the first one is the disability of extracting sufficient number of keypoints, and that is either because of the low light conditions or because of the absence of the texture on the obstacle surface such as in the case of some pillars and walls.

The second reason is the motion of the obstacle, the algorithm is able to detect the moving obstacle if the motion is towards the drone. However, in most cases of the moving obstacles according to Table I, the algorithm could not detect the appearance of the obstacles if the motion is around the drone such as in the case of the people and obstacles.

On the other hand, the processing time is estimated around 52.4*ms* as shown in Figure 9. This is due to work with patches of $62^o$ FOV which decreases the processing time from 106.1*ms* if working with the whole frames $92^o$ FOV.

TABLE I
ACCURACY OF DETECTION ALGORITHM

| | **People** | **Obstacle** | **Pillar** | **Wall** |
|---|---|---|---|---|
| **Total** | 153 | 72 | 20 | 40 |
| **Detected** | 148 | 67 | 19 | 37 |
| **Fail** | 5 | 5 | 1 | 3 |
| **Accuracy (%)** | **96.7** | **93.1** | **95** | **92.5** |

## VI. CONCLUSIONS

In this paper, two proposed algorithms of detection and avoidance obstacles are proposed and applied on a small

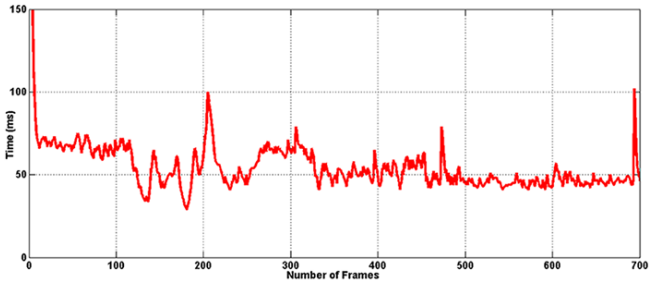


Fig. 9. Processing Time in milliseconds for the detection step.

UAV. Based on monocular processing, the relative size expansion of the obstacles is estimated and the approaching obstacles are detected from a distance between 90-120 *cm* with accuracy more than 92.5%. Then the avoidance algorithm sends a maneuver command to the drone based on the obstacle and drone positions.

## ACKNOWLEDGMENT

This research is supported by the Spanish Government through the CICYT projects (TRA2015-63708-R and TRA2013-48314-C3-1-R), Comunidad de Madrid through SEGVAUTO-TRIES (S2013/MIT-2713) and DGT through project (SPID 2015-01802).

## REFERENCES

[1] D. Shim, H. Chung, and S. Sastry, "Conflict-free navigation in unknown urban environments," *IEEE Robotics Automation Magazine*, vol. 13, no. 3, pp. 27–33, Sept. 2006.

[2] D. Luo, F. Wang, B. Wang, and B. Chen, "Implementation of obstacle avoidance technique for indoor coaxial rotorcraft with Scanning Laser Range Finder," in *Control Conference (CCC), 2012 31st Chinese*, July 2012, pp. 5135–5140.

[3] E. Shang, X. An, J. Li, and H. He, "A novel setup method of 3d LIDAR for negative obstacle detection in field environment," in *2014 IEEE 17th International Conference on Intelligent Transportation Systems (ITSC)*, Oct. 2014, pp. 1436–1441.

[4] K. Ariyur, P. Lommel, and D. Enns, "Reactive inflight obstacle avoidance via radar feedback," in *American Control Conference, 2005. Proceedings of the 2005*, June 2005, pp. 2978–2982 vol. 4.

[5] S. Hrabar, "3d path planning and stereo-based obstacle avoidance for rotorcraft UAVs," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. IROS 2008*, Sept. 2008, pp. 807–814.

[6] Y. Gao, X. Ai, J. Rarity, and N. Dahnoun, "Obstacle detection with 3d camera using U-V-Disparity," in *2011 7th International Workshop on Systems, Signal Processing and their Applications (WOSSPA)*, May 2011, pp. 239–242.

[7] I. Na, S. H. Han, and H. Jeong, "Stereo-based road obstacle detection and tracking," in *2011 13th International Conference on Advanced Communication Technology (ICACT)*, Feb. 2011, pp. 1181–1184.

[8] A. Broggi, S. Cattani, M. Patander, M. Sabbatelli, and P. Zani, "A full-3d voxel-based dynamic obstacle detection for urban scenario using stereo vision," in *2013 16th International IEEE Conference on Intelligent Transportation Systems - (ITSC)*, Oct. 2013, pp. 71–76.

[9] P. C. Merrell, D.-J. Lee, and R. W. Beard, "Obstacle avoidance for unmanned air vehicles using optical flow probability distributions," *Mobile Robots XVII*, vol. 5609, no. 1, pp. 13–22, 2004.

[10] S. Hrabar, G. Sukhatme, P. Corke, K. Usher, and J. Roberts, "Combined optic-flow and stereo-based navigation of urban canyons for a UAV," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005. (IROS 2005)*, Aug. 2005, pp. 3309–3316.

[11] A. Beyeler, J.-C. Zufferey, and D. Floreano, "3d Vision-based Navigation for Indoor Microflyers," in *2007 IEEE International Conference on Robotics and Automation*, Apr. 2007, pp. 1336–1341.

[12] K. Celik, S.-J. Chung, M. Clausman, and A. Somani, "Monocular vision SLAM for indoor aerial vehicles," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009. IROS 2009*, Oct. 2009, pp. 1566–1573.

[13] A. Chavez and D. Gustafson, "Vision-based obstacle avoidance using SIFT features," in *Advances in Visual Computing*. Springer, 2009, pp. 550–557.

[14] C. Bills, J. Chen, and A. Saxena, "Autonomous MAV flight in indoor environments using single image perspective cues," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, May 2011, pp. 5776–5783.

[15] G. de Croon, E. de Weerdt, C. De Wagter, and B. Remes, "The appearance variation cue for obstacle avoidance," in *2010 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec. 2010, pp. 1606–1611.

[16] W. Green and P. Oh, "Optic-Flow-Based Collision Avoidance," *IEEE Robotics Automation Magazine*, vol. 15, no. 1, pp. 96–103, Mar. 2008.

[17] J.-O. Lee, K.-H. Lee, S.-H. Park, S.-G. Im, and J. Park, "Obstacle avoidance for small UAVs using monocular vision," *Aircraft Engineering and Aerospace Technology*, vol. 83, no. 6, pp. 397–406, 2011.

[18] S. Saha, A. Natraj, and S. Waharte, "A real-time monocular vision-based frontal obstacle detection and avoidance for low cost UAVs in GPS denied environment," in *2014 IEEE International Conference on Aerospace Electronics and Remote Sensing Technology (ICARES)*, Nov. 2014, pp. 189–195.

[19] J. Byrne and C. J. Taylor, "Expansion segmentation for visual collision detection and estimation," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 875–882.

[20] T. Mori and S. Scherer, "First results in detecting and avoiding frontal obstacles from a monocular camera for micro unmanned aerial vehicles," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1750–1757.

[21] J. Kim and Y. Do, "Moving Obstacle Avoidance of a Mobile Robot Using a Single Camera," *Procedia Engineering*, vol. 41, pp. 911–916, 2012.

[22] B. M. Albaker and N. Rahim, "A survey of collision avoidance approaches for unmanned aerial vehicles," in *Technical Postgraduates (TECHPOS), 2009 International Conference for*, 2009, pp. 1–7.

[23] F. Kendoul, "A Survey of Advances in Guidance, Navigation and Control of Unmanned Rotorcraft Systems," vol. 29, no. 2, pp. 315–378, 2012.

[24] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[25] T. Krajnik, V. Vonasek, D. Fiser, and J. Faigl, "Ar-drone as a platform for robotic research and education," *Research and Education in Robotics-EUROBOT. Springer Berlin Heidelberg*, pp. 172–186, 2011.