2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Autonomous Obstacle Avoidance for UAV based on Fusion of Radar and Monocular Camera

Hang Yu, Fan Zhang, *Member, IEEE*, Panfeng Huang, *Senior Member, IEEE,* Chen Wang, and Li Yuanhao

***Abstract*—UAVs face many challenges in autonomous obstacle avoidance in large outdoor scenarios, specifically the long communication distance from ground stations. The computing power of onboard computers is limited, and the unknown obstacles cannot be accurately detected. In this paper, an autonomous obstacle avoidance scheme based on the fusion of millimeter wave radar and monocular camera is proposed. The visual detection is designed to detect unknown obstacles which is more robust than traditional algorithms. Then extended Kalman filter (EKF) data fusion is used to build exact real 3D coordinates of the obstacles. Finally, an efficient path planning algorithm is used to obtain the path to avoid obstacles. Based on the theoretical design, an experimental platform is built to verify the UAV autonomous obstacle avoidance scheme proposed in this paper. The experiment results show the proposed scheme cannot only detect different kinds of unknown obstacles, but can also take up very little computing resources to run on an onboard computer. The outdoor flight experiment shows the feasibility of the proposed scheme.**

I. Introduction

Unmanned aerial vehicles (UAV) can replace humans to perform tasks in complicated and harsh environments, with the advantages of flexible flight and easy to use. But in the flight environment, there are unknown and random obstacles everywhere. Thus, autonomous obstacle avoidance navigation system is the safety guarantee for the successful completion of flight missions. In recent years, the obstacle avoidance of UAV has become a research focus. Autonomous obstacle avoidance plays a decisive role in the trajectory planning of the drone design, which reflects the safety, stability and intelligence of the drone to a certain extent. The use of computer vision technology enables UAV to obtain a large amount of environmental information. Vision-based simultaneous localization and mapping (SLAM) refers to the robot's positioning based on vision and creating maps in completely unknown environments [1-3]. SLAM can provide positioning information and obstacle map information for the obstacle avoidance, but the technology needs amount of data storage during the navigation process. Therefore, SLAM is limited by the computing performance and storage space of the onboard computer, so it is difficult to use in large outdoor environment.

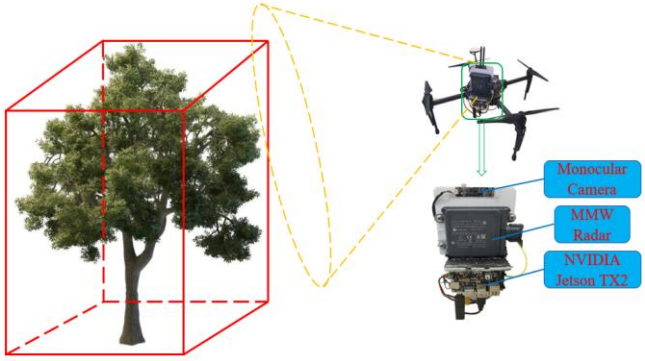

Figure 1. UAV autonomous obstacle avoidance scheme experimental verification platform. It use DJI Matrix 100 UAV with a monocular camera and a millimeter wave radar. One NVIDIA Jetson TX2 is as a processor.

In order to perform obstacle detection, localization, and avoidance trajectory in the flight missions in large outdoor scenario, there are three main technical difficulties to solve: firstly, various unknown obstacles outdoors should be quickly detected by the drone; secondly, to achieve autonomous navigation, it is necessary to build the coordinates of the obstacle in 3D space; thirdly, based on the precise localization and shape detection of the obstacle, UAV can independently plan flight paths to avoid obstacles.

In this paper, an autonomous obstacle avoidance scheme based on the fusion of millimeter wave radar and monocular vision is proposed. As shown in Figure 1, we built an experiment based on verification platform for autonomous obstacle avoidance of UAV. A monocular camera and millimeter wave radar are used to detect the spatial information of obstacles, and then a flight path is generated through a path planning algorithm. A DJI matrix 100 UAV is used for flight verification, and the entire algorithm runs on NVIDIA Jetson TX2. Firstly, the monocular camera is used to identify obstacles and obtain the most dominant information, such as the outlines of the obstacle and the position of the center point. Then the information is fused with a millimeter wave radar to acquire the spatial position of obstacles and establish a local map. Finally, the path planning algorithm will generate a reasonable path to avoid obstacles.

This scheme has three significant advantages: 1) vision can identify as many unknown obstacles as possible, and it is

*Research supported in part by the National Natural Science Foundation of China under Grant Nos.61803313, 91848205, 61725303, in part by the Fundamental Research Funds for the Central Universities under Grant No. 3102019HTQD003, and in part by the Young Talent Fund of University Association for Science and Technology in Shaanxi, China under Grant No. 20190102, and in part by the Natural Science Basic Research Plan in Shaanxi Province of China (No. 2019JQ-345, 2019JQ-411, 2019JM-406). (corresponding author: Fan Zhang, phone: (+86)029-88460366; fax: (+86)029-88460366-4; e-mail: fzhang@nwpu.edu.cn)

Hang Yu, Fan Zhang, Panfneg Huang and Li Yuanhao are with the National Key Laboratory of Aerospace Flight Dynamics, Research Center for Intelligent Robotics, School of Astronautics, Northwestern Polytechnical University, 127 West Youyi Road, Xi'an, China (e-mail: yuhang@mail.nwpu.edu.cn, fzhang@nwpu.edu.cn, pfhuang@nwpu.edu.cn, 327454122@qq.com).

Chen Wang is with the Key Laboratory for Highway Construction Technology and Equipment of Ministry of Education, Chang'an University, middle-section of Nan'er Huan Road, Xi'an, ShaanXi Province, 710064, China (email: wangchenjustin@chd.edu.cn).

not affected by the lighting environment; 2) combining the advantages of camera and millimeter wave radar, specifically obstacles' information in the pixel plane and radar's two-dimensional coordinates, stable and accurate 3D coordinates of obstacles is obtained by using EKF data fusion; 3) The whole navigation process can be run in real time on the onboard computer used without ground stations.

II. Related Work

The detection algorithm of the vision-based obstacle detection technology is the key, and deep learning has been the focus. In 2014, the R-CNN [4] proposed by Ross Girshick achieved good results. R-CNN opened up new ideas for target detection. In 2015, the YOLO algorithm proposed by Redmon et al. who modeled object detection as a regression problem. The algorithm is designed by using a simple neural network to directly predict the bounding box and class probability for the entire images [5]. The YOLO algorithm has outstanding advantages, especially in terms of detection speed, and can maintain considerable accuracy and strong generalization ability. However, the biggest problem of deep learning algorithms is the large storage of data sets. Deep learning methods are more suitable for autonomous driving because obstacles that cars encounter on the road are more predictable. But drones face more unknown aerial obstacles for avoidance navigation in large outdoor scenarios. At this time, target detection algorithms based on deep learning cannot detect obstacles.

Multi-sensor fusion technology is important in the study of environmental perception and obstacle avoidance. Wang et al. proposed an indoor navigation scheme of drones based on lidar and vision [6], using inertial measurement unit (IMU) and vision for state estimation, and then using EKF-based data fusion with lidar to sense the environment, and finally achieving navigation. Bachrach et al. used lidar, IMU and monocular camera for obstacle avoidance research of rotorcraft drones, in which lidar is used to measure the distance and orientation of obstacles; combined environmental details of images are used to perform obstacle avoidance and navigation [7]. Both the two methods take a lot of computation to run on the onboard computer, and thus the communication between the ground station and the onboard computer requires high technique to be feasible. Due to the limitation of data transmission volume and distance of the wireless communication, UAV cannot complete autonomous obstacle avoidance tasks in the large outdoor scenario.

Traditional path planning algorithms, for example the artificial potential field method proposed by Khatib [8] in 1986, need to model obstacles in a certain space. The computational complexity and the dimension have an exponential relationship, and these algorithms are not appropriate to solve the path planning of UAV in 3D space. Sampling-based methods such as PRM [9] and RRT [10] have achieved excellent results in real-time performance. Although PRM and RRT algorithms can generate feasible solutions in a short time, the obtained paths are not optimal. Sertac et al. [11] proposed a real-time path planning algorithm called RRT*. This algorithm cannot only find the feasible solution as quickly as RRT, but also can converges to the optimal solution.

III. Approach

A. System Framework

As shown in Figure 2, the UAV autonomous obstacle avoidance system consists of three subsystems: hardware subsystem, communication subsystem and algorithm subsystem. The hardware subsystem contains a DJI matrix 100 UAV, a monocular camera, a millimeter wave radar and an onboard computer (NVIDIA Jetson TX2). The UAV and sensors are connected with onboard computer. Robot Operating System (ROS) runs on the onboard computer which is installed Ubuntu 16.04. Hardware devices are published as messages through ROS: DJI provides developers with application programming interface (API) based on ROS to transfer data through the serial port; based on the CAN bus provided by a millimeter wave radar, data is parsed and published into custom radar messages via communication protocols; with a common monocular USB interface camera, OpenCV is used to obtain image information and convert it into ROS standard messages.

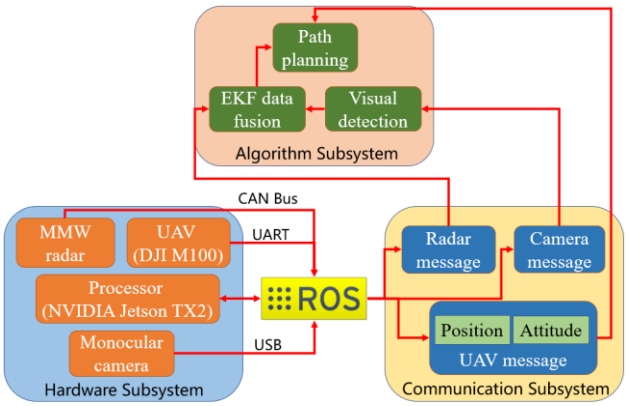

Figure 2. UAV autonomous obstacle avoidance system which including three subsystem: hardware, communication and algorithm.

With the obtained information of UAV and sensors, it enters the algorithm subsystem. The algorithm subsystem is composed of three modules, including a visual detection, an EKF data fusion, and a path planning module. In this paper, an obstacle detection algorithm based on point cloud clustering is proposed. This algorithm can be used to detect unknown obstacles in real time, with a small amount of calculation and without point cloud information. The information output by visual detection is fused with radar data to obtain detailed spatial coordinates of obstacles. The path planning algorithm based on RRT* generates a 3D path based on the fused obstacle information. The entire autonomous obstacle avoidance scheme can satisfy the real-time requirement for outdoor missions.

B. Visual Detection

As shown in Figure 3, each input image needs to be detected with ORB feature points, and then optical flow tracking is used to calculate the displacement of feature points between different frames. In this paper, optical flow tracking has two main functions: 1) calculating the disparity between the current frame and previous key frame according to the results of optical flow tracking, and determining whether the current frame is a key frame; 2) if current frame is not a key

frame, using the results of optical flow tracking of obstacles feature points as the position information. The advantage of flow tracking is fast. In order to ensure the accuracy of motion recovery, ORB feature matching is used to determine the displacement of feature points between key frames, so the visual detection algorithm in this paper gives a certain balance between the speed and accuracy.

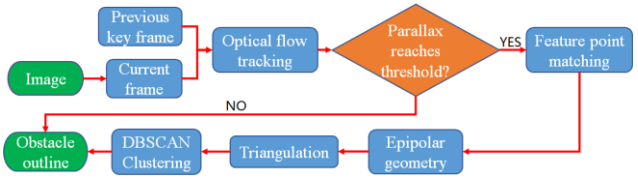

Figure 3. Visual detection algorithm based on motion recovery and point cloud clustering.

The corresponding feature points of a point in space on the previous key frame and the current frame are defined as $P_i(u_i, v_i)$ and $P_{i+1}(u_{i+1}, v_{i+1})$, respectively. Then the parallax of this point can be expressed as:

$$d_i = \sqrt{(u_{i+1} - u_i)^2 + (v_{i+1} - v_i)^2} \tag{1}$$

The parallax d_i is compared to a parallax threshold d_{thre}. If $d_i > d_{thre}$, points exceeding the parallax threshold will be increased by one. If the number of points exceeding the parallax threshold is more than another point threshold N_{thre}, then the disparity of the two frames is considered to be sufficient, and the current frame will be set as the current key frame. Then the motion of the two frames will be restored.

Epipolar geometry is used to restore the motion of two key frames. As shown in Fig. 4, P express a point in space. The projections on two adjacent key frames are p_1 and p_2, respectively. The camera centers corresponding to the two key frames are O_1 and O_2, respectively. O_1, O_2, and P define a plane, called the epipolar plane. The intersections e_1, e_2 of the O_1O_2 and the imaging planes are called epipoles, while l_1 and l_2 are called epipolar lines. Solving the motions R and t between I_1 and I_2 is the epipolar geometry problem.

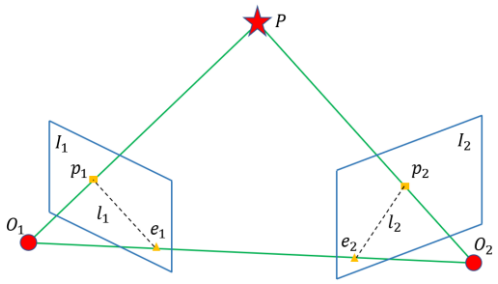

Figure 4. Epipolar geometry. The motion between adjacent key frames is restored according to the feature points matching different frames.

The epipolar constraint can be expressed as:

$$p_2^T K^{-T} t^{\wedge} R K^{-1} p_1 = 0, \tag{2}$$

where K represents the internal parameter matrix and $t^{\wedge}$ represents the antisymmetric matrix of the vector t. Here we take $E = t^{\wedge}R$, and E is called as essential matrix. The E can be solved by eight-point-algorithm. Consider eight pairs of matching points, their normalized coordinates are $x_1 = [u_1^i, v_1^i, 1]^T$ and $x_2 = [u_2^i, v_2^i, 1]^T$ $(i = 1, 2, ..., 8)$. Writing the matrix $E_{3\times3}$ as a vector $\boldsymbol{e}$, from the epipolar constraint, we can get a system of linear equations:

$$(\boldsymbol{N}_1, \boldsymbol{N}_2, \ldots, \boldsymbol{N}_8)^T \boldsymbol{e} = \boldsymbol{0}, \tag{3}$$

where

$$\boldsymbol{N}_i = [u_1^i u_2^i, u_1^i v_2^i, u_1^i, v_1^i u_2^i, v_1^i v_2^i, v_1^i, u_2^i, v_2^i, 1]^T \tag{4}$$

This linear equations' coefficient matrix is composed of the positions of the feature points. If the coefficient matrix is full rank, each element of E can be obtained by (4). When E is obtained, R and t can be solved by singular value decomposition (SVD). However, since the monocular camera cannot obtain the real scale information, the solved t is equivalent to the normalized coordinates.

When the key frames motion is obtained, the normalized space coordinates of the feature points can be conveniently calculated using the principle of triangulation. For example, assuming x_1 and x_2 are normalized coordinates of feature point at two key frames, then they satisfy:

$$s_1 x_1 = s_2 R x_2 + t \tag{5}$$

With the known parameters R and t, the distance between the space point to camera should be calculated. Multiplying x_1^T and $(Rx_2)^T$ on both sides of (5) respectively, we can get a system of equations:

$$\begin{bmatrix} x_x^T x_1 & -x_1^T R x_2 \\ (Rx_2)^T & -(Rx_2)^T (Rx_2) \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} x_1^T t \\ (Rx_2)^T t \end{bmatrix} \tag{6}$$

Bringing normalize the pixel coordinates, R and t into (6), then s_1 and s_2 can be calculated. Due to the existence of noise, R and t estimated through epipolar geometry cannot make (6) completely true. A general method is to find the least square solution.

After obtaining the normalized point cloud of all feature points, the point cloud needs to be clustered to detect different obstacles. This paper uses DBSCAN (density-based spatial clustering of applications with noise) algorithm [12]. The set of feature points in space is supposed as $D = (\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_m})$. Two parameters *Eps* and *MinPts* are used to describe the closeness of the sample distribution in the neighborhood.

Algorithm 1 is the pseudo code of the DBSCAN. For any sample $\boldsymbol{x}_j \in D$, its neighborhood subsample set $N_{Eps}(\boldsymbol{x}_j)$ can be obtained through Euler distance. Supposing sample $\boldsymbol{x}_i \in D$, if $\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2} \le Eps$, then $N_{Eps}(\boldsymbol{x}_j) = N_{Eps}(\boldsymbol{x}_j) \cup \{\boldsymbol{x}_i\}$. If the number of samples in the neighborhood subsample set is greater than *MinPts*, then $\boldsymbol{x}_j$

is the core object. If different core objects are included in their respective neighborhoods, they are connected, and their neighborhood subsample sets are considered to belong to the same class. DBSCAN algorithm divides samples into different clusters according to this idea. If a sample does not belong to the neighborhood subsample set of any core object, the sample is considered as noise.

Algorithm 1: DBSCAN clustering

Input:
$D=(\boldsymbol{x}_1,\boldsymbol{x}_2,...,\boldsymbol{x}_m)$: Sample set to be clustered
Eps : neighborhood radius
$MinPts$: minimum neighborhood samples
Output:
C : cluster division

```
Initialize core object set Ω = ∅
Initialize the number of clusters k = 0
Initialize unvisited sample set Γ = D
Initialize cluster division C = ∅
for j = 1,2,...,m do
    N_Eps(x_j) = Neighborhoodsubsample(D, x_j, Eps)
    if |N_Eps(x_j)| ≥ MinPts then
        Ω = Ω ∪ {x_j}
while Ω ≠ ∅ do
    o=random(Ω)
    Initialize current core object queue Ω_cur = {o}
    Initialize cluster number k = k + 1
    Initialize current cluster sample set C_k = {o}
    Update unvisited sample set Γ = Γ − {o}
    while Ω_cur ≠ ∅ do
        Update core object set Ω = Ω − C_k
        o'=random(Ω_cur)
        N_Eps(o') = Neighborhoodsubsample(D, o', Eps)
        Δ = N_Mps(o') ∩ Γ
        Update current cluster sample set C_k = C_k ∪ Δ
        Update unvisited sample set Γ = Γ − Δ
        Update Ω_cur = Ω_cur ∪ (Δ ∩ Ω) − o'
    Update cluster division C = {C_1, C_2, ..., C_k}
    Update core object set Ω = Ω − C_k
return C
```

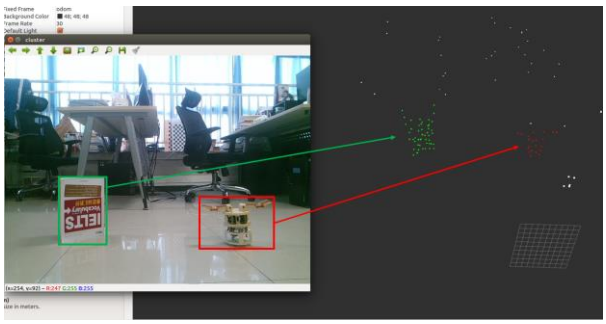

Figure 5. The result of point cloud clustering. A sparse point cloud of a book and a self-made mechanical claw was reconstructed and different objects can be distinguished.

The visual detection algorithm is verified with a simple experiment. A book and a self-made mechanical claw are prepared as obstacles, and experiment shows that the algorithm can successfully classify different obstacles. As shown in Figure 5, point clouds of different colors represent different obstacles, and white points represent noise.

C. EKF Data Fusion

The millimeter wave radar used in this paper is able to obtain the center points of obstacles after clustering. Through CAN protocol data analysis, the coordinates of the center point of the obstacle and the speed can be obtained. To fuse the radar measurement with the results of visual detection, the points detected by the radar must be projected into the pixel coordinate system of the camera. Thus, the first thing to do is coordinate transformation.

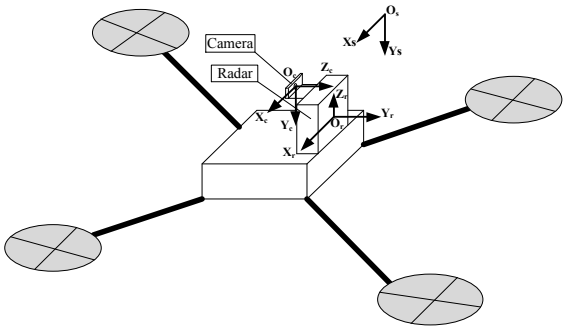

Figure 6. The coordinate systems of UAV autonomous obstacle avoidance scheme experimental verification platform.

Fig. 6 illustrates the relationships between radar, camera, and pixel coordinate system, where $O_rX_rY_rZ_r$, $O_cX_cY_cZ_c$ and $O_sX_sY_s$ represent the radar coordinate system, camera coordinate system and pixel coordinate system, respectively. Assuming the coordinate of a point in the radar coordinate system is $(x_{ri},y_{ri})^T$, the projection of the point on the camera coordinate system can be formulated as:

$$(x_{ci},y_{ci},z_{ci},1)^T = T(x_{ri},y_{ri},H,1), \tag{7}$$

where T is a 4×4 external parameter matrix of the camera and radar, consisting of a rotation matrix and a translation vector. H denotes the height difference between the camera and the radar. Both T and H can be calibrated by measurement. According to the conversion relationship between the camera coordinate system and the pixel coordinate system, we can also get the projection coordinates of the point in the pixel:

$$z_{ci}(x_{si},y_{si},1)^T = K(x_{ci},y_{ci},z_{ci}), \tag{8}$$

where R is the camera internal parameter matrix, which can be calibrated by Matlab or OpenCV, After the points in the radar coordinate system are projected onto the pixel plane, the measurements of the radar data and the output of the visual inspection can be fused.

In this paper, EKF is used for data fusion. The position and velocity $\boldsymbol{x}_f(p_x,p_y,p_z,v_x,v_y)$ of the obstacle in the radar coordinate system should be solved first. The measurement of

radar is $\boldsymbol{x}_r(x_r, y_r, v_{xr}, y_{xr})$, and the camera measurement can be represented as $\boldsymbol{x}_s(u, v)$. According to the EKF [13], the prediction of the state can be expressed as:

$$\begin{cases} \boldsymbol{x}_f^{'} = F\boldsymbol{x}_f \\ P^{'} = FPF^T + Q \end{cases} \tag{9}$$

In which, there are three variables to be solved, namely F , P and Q . F represents the changing of the system state. Here only linear system is considered. It is easy to know:

$$F\boldsymbol{x}_f = \begin{bmatrix} 1 & 0 & 0 & dt & 0 \\ 0 & 1 & 0 & 0 & dt \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ v_x \\ v_y \end{bmatrix} \tag{10}$$

In (9), P indicates the uncertainty of the system, which is expressed by the covariance of $\boldsymbol{x}_f$ and is an identity matrix here. Q indicates other external disturbances that $\boldsymbol{x}_f^{'} = F\boldsymbol{x}_f$ cannot include.

When the state prediction is completed, next step is to update the state quantity according to the radar and camera measurement data. The radar uses KF, and the update formula of the state quantity is:

$$\begin{cases} \boldsymbol{y}_r = \boldsymbol{x}_r - H\boldsymbol{x}_f , \quad S = HPH^T + R_{radar} , \\ K = PH^T S^{-1} , \quad \boldsymbol{x}_f^{'} = \boldsymbol{x}_f + K\boldsymbol{y}_r , \\ P^{'} = (I - KH)P, \end{cases} \tag{11}$$

, and camera uses EKF:

$$\begin{cases} \boldsymbol{y}_s = \boldsymbol{x}_s - f(\boldsymbol{x}_f), \quad S = H_j PH_j^T + R_{cam} , \\ K = PH_j^T S^{-1} , \quad \boldsymbol{x}_f^{'} = \boldsymbol{x}_f + K\boldsymbol{y}_s , \\ P^{'} = (I - KH_j)P \end{cases} \tag{12}$$

In equation (11), $H = [\boldsymbol{I}, \boldsymbol{0}]$ represents a linear transformation. R_{radar} represents the uncertainty of the measured value, which is usually provided by the sensor manufacturer. In equation (12), The nonlinear mapping $f(\boldsymbol{x}_f)$ from state space to camera measurement space is:

$$f(x_f) = [u, v]^T = KT[p_x, p_y, p_z, 1]^T \tag{13}$$

, and H_j represents the Jacobian matrix after linearization of the nonlinear mapping:

$$\begin{cases} H_j = -\begin{bmatrix} f_x / Z^{'} & 0 & -f_x X^{'} / Z^{'2} \\ 0 & f_y / Z^{'} & -f_y Y^{'} / Z^{'2} \end{bmatrix} R \\ [X^{'}, Y^{'}, Z^{'}]^T = R[p_x, p_y, p_z, 1], \end{cases} \tag{14}$$

where f_x, f_y represents the focal length of the camera and R is the rotation matrix from the radar coordinate system to the camera coordinate system. By now the state quantities according to radar and camera measurements have been updated.

Through the EKF data fusion, the relationship between radar data and camera measurements can be corresponded. Combining these two sensors, we can obtain not only the outline information of obstacles, but also the coordinates of obstacles in the real world.

D. Path Planning

RRT is an efficient planning method in multidimensional space. An initial point is used as the root node, and child nodes can be randomly added to generate a random expanded tree. When the child nodes in the random tree contain the target point or enter the target area, the path from the initial point to the target point can be found in the random tree.

Algorithm 2: Bi-RRT*

Input: the C-space χ ; start configuration x_{start} ; goal configuration x_{goal} , number of iterations K .

Output: Path connecting x_{start} to x_{goal} .

$V_1 \leftarrow \{x_{start}\}$, $E_1 \leftarrow \varnothing$, $T_1 \leftarrow (V_1, E_1)$
$V_2 \leftarrow \{x_{goal}\}$, $E_2 \leftarrow \varnothing$, $T_2 \leftarrow (V_2, E_2)$
for $k = 1$ to K **do**
  $x_{rand} \leftarrow RANDOM\_STATE(\chi)$
  $x_{near} \leftarrow NEARWEST\_NEIGHBOR(x_{rand}, T_1)$
  $x_{new} \leftarrow STEER(x_{rand}, x_{near})$
  if $obstaclefree(x_{new})$ **then**
    $V_1 \leftarrow V_1 \cup \{x_{new}\}$, $E_1 \leftarrow E_1 \cup \{(x_{near}, x_{new})\}$
    $rewire\_tree(T_1, x_{new}, x_{start}, r)$
    $x_{near}^{'} \leftarrow NEARWEST\_NEIGHBOR(x_{new}, T_2)$
    $x_{new}^{'} \leftarrow STEER(x_{new}, x_{near}^{'})$
    if $obstaclefree(x_{new}^{'})$ **then**
      $V_2 \leftarrow V_2 \cup \{x_{new}^{'}\}$, $E_2 \leftarrow E_2 \cup \{(x_{near}^{'}, x_{new}^{'})\}$
      $rewire\_tree(T_2, x_{new}^{'}, x_{goal}, r)$
    if $x_{new} = x_{new}^{'}$ then
      $T = T_1 \cup T_2$
      return T

Bidirectional RRT [14] was proposed to increase the search speed of RRT. Because two fast expanding random trees are grown from the initial state point and the target state point to search the state space, the efficiency will be higher. Based on the original RRT, RRT* improves the method of selecting parent nodes [11]. A cost function is used to select the node with the smallest cost in the expanded node field as the parent node. At the same time, it iteratively rewires nodes on the existing tree after each iteration, in order to ensure the computational complexity and progressive optimal solution. In this paper, Bi-RRT* algorithm combining of Bidirectional RRT and RRT* is used to improve the computing efficiency and ensure the optimality.

Algorithm 2 is pseudo code for Bi-RRT*, where χ represents the search space. With a starting point and a target point, a path can be obtained after multiple iterations. In the

Algorithm 2, *RANDOM _STATE*() indicates that random points are generated from the search space. *NEAREST _NEIGHBOR*() denotes a random tree to find the node closest to the random point. *STEER*() means the expanding of random tree according to the direction of the input value. *obstaclefree*() is used to determine when an obstacle is encountered. Algorithm 3 is mainly used to reselect the parent node as well as rewire the tree.

Algorithm 3: *rewire _tree*

Input: the tree T_i; newly added node x; initial node x_0; search radius r

Output: rewired tree T_i

$x_{near\_neighbor} \leftarrow findnear\_neighbor(T_i, x, r)$

if $obstaclefree(x, T, r)$ **then**

  for all $x_{near\_neighbor}$ **do**

    $cal(dist(x, x_{near\_neighbor}) + cost(x_{near\_neighbor}, x_0))$

    $x_{new\_parent} \leftarrow \min(dist(x, x_{near\_neighbor}) + cost(x_{near\_neighbor}, x_0))$

    $E_i \leftarrow E_i \cup \{(x_{near\_parent}, x)\}$

  for all $x_{near\_neighbor}$ **do**

    $cal(dist(x_{near\_neighbor}, x) + cost(x, x_0))$

    $x_{new\_neighbor_{newparent}} \leftarrow \min(dist(x_{near\_neighbor}, x) + cost(x, x_0))$

    $E_i \leftarrow E_i \{(x_{new\_neighbor_{newparent}}, x_{near\_neighbor})\}$

return T_i

The path obtained by the Bi-RRT* algorithm is in the form of tree nodes. In order to make the path more consistent with the flying characteristics of UAV, the path should be smoothed. In this paper, Catmull-Rom interpolation algorithm [15] is used to smooth the path.

IV. Experiments and Results

In this paper, all the algorithms have been verified on experimental verification platform, which together make up the UAV autonomous obstacle avoidance scheme. The three main schemes, including multiple obstacles detection, multi-sensor fusion localization and path planning algorithms are respectively tested on the experiment platform. In the end, the overall autonomous obstacle avoidance flight is completed.

A. visual detection experiments

In the visual detection experiments, various objects are used as targets for verification. As shown in Figure 7 the algorithm in this paper can achieve reliable results, even for totally different targets. The lighting conditions in Figure 7d and Figure 7a-c are quite different, but the algorithm performs well. Figure 8 shows the result of the visual detection algorithm outdoors. Although the outdoor environment is more complex, the algorithm described in this paper can still detect obstacles in the camera field of view. In general, for unknown obstacles in the complex and changeable environment, the visual detection algorithm has certain robustness. The algorithm is fast enough to reach a detection speed of 11 frames per second at an image acquisition frequency of 18Hz.

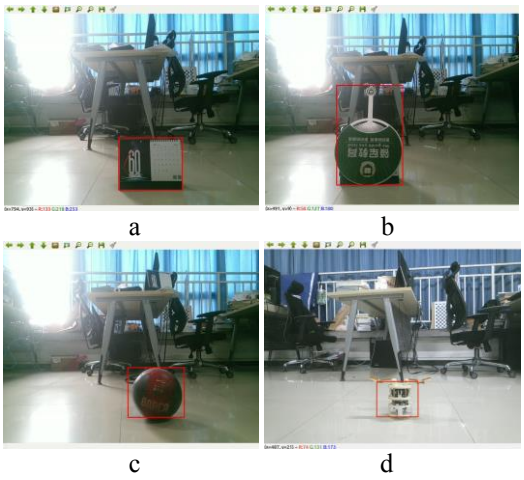

Figure 7. Results of visual detection algorithms for different objects.

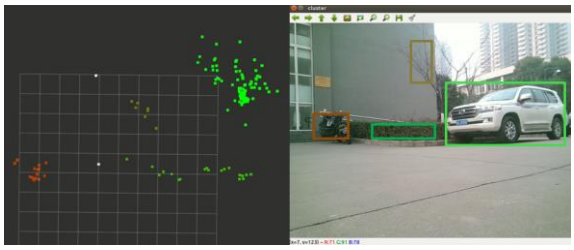

Figure 8. Outdoor large scene obstacles detection results and 3D point cloud information.

B. Obstacle Positioning based on EKF Data Fusion

With the collected measurement data of radar and monocular camera, the experiment of obstacle location based on EKF data fusion is carried out. The above vision detection algorithm can obtain the pixel coordinates (u, v) of obstacles, and the millimeter wave radar can detect the two-dimensional coordinates and speed (x, y, v_x, v_y) of obstacles in the radar plane. The raw radar data shown in Figure 9 has mutation caused by error detection or missing detection, while the fusion filtered data has no mutation. This also ensures that the UAV flight process is more stable and safer. In addition, after EKF data fusion, as shown in Figure 10, accurate 3D coordinate information of obstacles can be obtained, so the information after fusion is richer than that obtained by a single sensor.

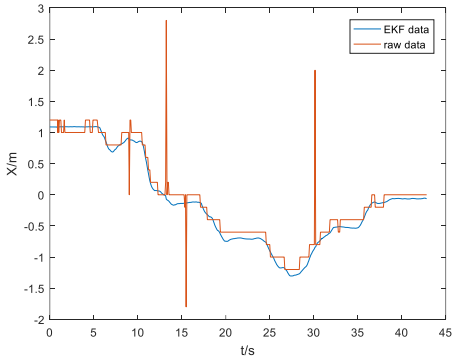

Figure 9. Comparison between the raw data obtained by radar and the data after EKF fusion. The blue line represents the data after fusion, and the red line represents the radar raw data.

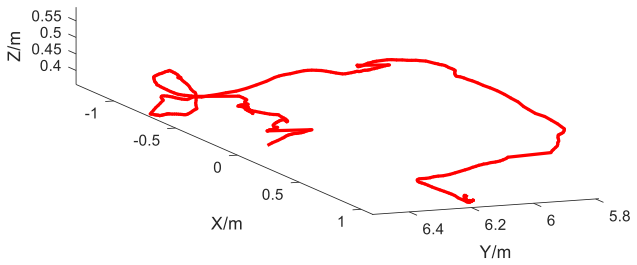

Figure 10. Trajectory of obstacle relative to UAV after EKF data fusion.

C. *Path Planning based on Bi-RRT**

Based on the outline and position information of obstacles, the OctoMap module is used in ROS to build a 3D map. As shown in Figure 11, the cuboids indicate obstacles, and different colors indicate changes in the height of obstacles. The green polyline represents the original path generated by the Bi-RRT* algorithm, and the red curve represents the path smoothed by the Cantmull-Rom algorithm. Considering the flight characteristics of the drone, the smoothed path is used in the flight experiment. Although the path generated by Bi-RRT* is random, because of its superiority, it can basically complete path planning in 3D space within 1s. We also finally verify the path planning algorithm in the outdoor scenario shown in Figure 12. The UAV can fly along the planned path and successfully avoided obstacles.

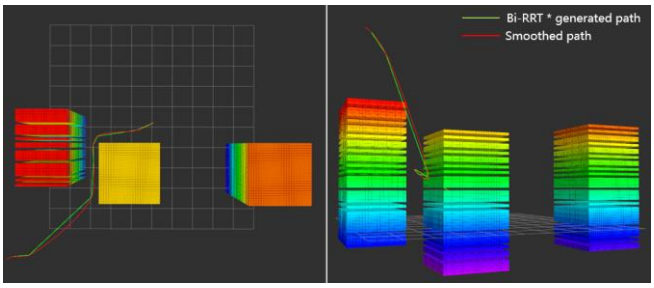

Figure 11. Path planning in 3D space based on Bi-RRT*. The left and right sides show different perspectives.

D. *UAV outdoor obstacle avoidance flight experiments*

In order to verify the feasibility of the whole algorithm, outdoor experimental environment is constructed. A special tested field is selected to test the proposed scheme and platform for safety. As shown in Figure 12, the black cubes are machined as obstacles. The red frame and blue dots indicate the results of visual detection, while the small green circles represent the projection of the radar data in the image. After the radar data corresponds to the results of visual detection, the coordinates in blue at the top of Figure 12 are obtained after the EKF data fusion. Under the Bi-RRT* path planning algorithm, the UAV successfully avoids obstacles and reaches the target point.

The test results show that for the monocular camera with 18Hz frame rate and the radar with 20 Hz frame rate, the data output speed of vision detection and EKF data fusion can reach 11 Hz. Bi-RRT * algorithm can plan path within 1s. It can be seen that the proposed algorithm can achieve real-time obstacle avoidance flight of UAV when it is only applicable to onboard computer of UAV.

Figure 12. UAV outdoor obstacle avoidance flight experiments.

V. Conclusion

To solve the problem of autonomous obstacle avoidance of UAV in large outdoor environment, this paper proposes an obstacle detection, location and avoidance scheme based on the fusion of radar and monocular camera. The visual detection algorithm, data fusion algorithm, and path planning algorithm were verified by the drone autonomous obstacle avoidance experimental verification platform. The experimental results show that the designed visual detection algorithm can effectively recognize various unknown obstacles and has strong robustness; the data fusion algorithm can combine radar and camera to obtain the true 3D coordinates of the obstacles stably; the path planning algorithm can efficiently plan paths in 3D space and avoid obstacles.

The obstacle avoidance schemes described in this paper have some shortcomings. For example, GPS still needs to be used for position during navigation. In the follow-up work, a visual inertial odometer (VIO) will be added. On one hand, it can get rid of the dependence of GPS, and on the other hand, it may achieve better results by combining the local map of VIO with millimeter wave radar.

References

[1] Mur-Artal, Raul, J. M. M. Montiel and J. D. Tardos, "ORB-SLAM: a Versatile and Accurate Monocular SLAM System." *IEEE Transactions on Robotics*, vol. 31, no. 5, pp.1147-1163, 2015.

[2] Mur-Artal, Raul and J. D. Tardos, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras." I*EEE Transactions on Robotics*, pp.1-8, 2017.

[3] Tong. Q, L. Peiliang and S. Shaojie, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator." *IEEE Transactions on Robotics*, pp.1-17, 2018.

[4] Girshick R, Donahue J, Darrell T. “Rich feature hierarchies for accurate object detection and semantic segmentation.” *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp.580-587, 2014.

[5] Redmon. J and A. Farhadi, "YOLO9000: Better, Faster, Stronger." 2016.

[6] Wang Fei, Cui Jin-Qiang and Chen Ben-Mei, “A Comprehensive UAV Indoor Navigation System Based on Vision Optical Flow and Laser FastSLAM”. *Acta Automatica Sinica*, pp.1889-1900, 2013.

[7] Bachrach. A, He. R and Roy. N, “Autonomous flight in unknown indoor environments”. *International Journal of Micro Air Vehicles*, vol. 1, no. 4, pp.217-228, 2009.

[8] Khatib and Oussama, "Real-Time Obstacle Avoidance System for Manipulators and Mobile Robots." *International Journal of Robotics Research*, vol. 5, no. 1, pp.90-98, 1986.

[9] Kavraki, L. E. , et al. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces." *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp.566-580, 2002.

[10] Cheng. P, Z. Shen and S. La Valle, "RRT-based trajectory design for autonomous automobiles and spacecraft." *Archives of Control Sciences*, vol. 11, no. 3, pp.167-194, 2001.

[11] Sertac, et al, "Anytime Motion Planning using the RRT*." *IEEE International Conference on Robotics and Automation*(*ICRA)*, 2011.

[12] A. Domenica and M. Coppola, "Experiments in Parallel Clustering with DBSCAN." 2001.

[13] Blanc. C, L. Trassoudaine, and J. Gallice, "EKF and particle filter track to track fusion: A quantitative comparison from Radar/Lidar obstacle tracks." *2005 8th International Conference on IEEE*, 2005.

[14] Kuffner and S. M. Lavalle, "RRT-connect: An efficient approach to single-query path planning." *IEEE International Conference on Robotics and Automation(ICRA)*, 2002.

[15] Twigg, C, "Catmull-rom splines." *Computer* vol. 41, no. 6, pp.4-6, 2003.