Velocity Prediction for Multi-rotor UAVs Based on Machine Learning

Rongzhi Wang, Danping Zou, Ling Pei, Peilin Liu and Changqing Xu

Abstract Currently, multi-rotor UAV's navigation mainly depends on satellite navigation systems. In many environments (e.g., indoor, urban, or canyon), lack of satellite signal will lead UAV navigation to failure. In that case, one backup solution is to use the inertial navigation method. Inertial navigation method integrates measurements from gyroscope and accelerometer to obtain the orientation, speed, and position. Due to the instability of sensor bias, a slight orientation error caused by gyroscopic bias change will lead to enormous position error. Noting that there is a strong correlation between the pose and velocity for a flying multi-rotor UAV, we can search for a solution to calculate velocity directly from UAV's pose. In this work, we propose to use support vector machine (SVM)-based machine learning technique to predict the moving speed of the aircraft. This approach builds a relationship directly between the orientation data and velocity by training. The experiment has two stages. In early stage, we have tested our method in simulation environment; then at later stage, we have tested our method in real-world cases. Experimental results our method can predict the UAV's velocity within the error of 0.3 m/s and the squared correlation coefficient between the predicted velocity and the ground truth is about 0.8. The method can be used as a complementary navigation source to achieve higher localization accuracy and stability.

Keywords Multi-rotor UAV · Machine learning based INS (inertial navigation system) integration · SVM (supported vector machine)

R. Wang · D. Zou (✉) · L. Pei · P. Liu · C. Xu
Shanghai Key Laboratory of Navigation and Location-Based Services,
School of Electronic Information and Electrical Engineering,
Shanghai Jiao Tong University, Shanghai 200240, China
e-mail: dpzou@sjtu.edu.cn

R. Wang
e-mail: wangrongzhi0225@163.com

J. Sun et al. (eds.), *China Satellite Navigation Conference (CSNC) 2016 Proceedings: Volume II*, Lecture Notes in Electrical Engineering 389,
DOI 10.1007/978-981-10-0937-2_41

1 Introduction

Existing multi-rotor UAV's navigation mainly depends on satellite navigation systems, such as GPS, to obtain the position information. But in many scenes, there may be a satellite signal failure, such as around tall buildings, in a deep canyon, and indoor space. Because of loss of satellite signal, GNSS receiver cannot provide the position of UAV, which in consequence cause autonomous navigation to fail.

The classic way remedy of this issue is to use inertial navigation as a backup. An inertial navigation system, INS, consists of gyroscope and accelerometer, which measure the linear acceleration and angular velocity of UAV. The instantaneous velocity and altitude are obtained by integration operation [1]. However, the measurement errors of inertial sensors can be amplified by integration and accumulated over time, and finally lead to a significant error in a short time [2].

Another solution to lack of GNSS signals is to use optical flow sensor [3]. There are many micro-UAVs equipped with optical flow sensors that are used to achieve like obstacle avoidance [4], automatic landing [5], and stable hovering [6]. However, optical flow is influenced by the surface texture and illumination conditions, especially for indoor navigation in the night. Productions from the most advanced UAV companies such as Parrot and DJI have no good solution to the problem of optical flow.

Notice that the special dynamic model of the multi-rotor aircraft, we can build a mathematical relationship between UAV's flying velocity and its attitude and acceleration. But in practice, it is very hard to calibrate all the parameters of dynamic model. So we view this model as a black-box and adopt machine learning technique to train the model, and finally accomplished use the trained parameters to estimate the UAV's velocity.

The proposed method relies on only the IMU readings, without using integration that usually produces a large position error as a typical inertial navigation system does. It also works in any light conditions, not limited only in the daytime, which can be a complementary navigation approach when satellite signal losses and optical flow fails.

2 The Dynamic Model

In this paper, we consider only the altitude-hold control mode that exists in most multi-rotor UAV platforms for clarity. Out method, however, is not limited to this case, and can be also adopted in other modes.

Let $\{A\}$ denote a right-hand inertial frame with unit vectors along the axes denoted by $\{\overrightarrow{\alpha_1}, \overrightarrow{\alpha_2}, \overrightarrow{\alpha_3}\}$; Let $\{B\}$ be a right-hand body fixed frame for the airframe with unit vectors $\{\overrightarrow{b_1}, \overrightarrow{b_2}, \overrightarrow{b_3}\}$. According to Newton's law, we have

$$\mathrm{m}\dot{v} = mg\overrightarrow{a_3} + RF_B \tag{1.1}$$

where F_B combines the principle non-conservative forces applied to the quadrotor's body frame; R is rotation matrix from body frame $\{B\}$ to inertial frame $\{A\}$ transform,

$$R = \begin{bmatrix} c\psi c\theta - s\phi s\psi s\theta & -c\phi s\psi & c\psi s\theta + c\theta s\phi s\psi \\ c\theta s\psi + c\psi s\phi s\theta & c\phi c\psi & s\psi s\theta - c\psi c\theta s\phi \\ -c\phi s\theta & s\phi & c\phi c\theta \end{bmatrix} \tag{2}$$

where ϕ, θ, ψ are the roll, pitch, yaw angle respectively; c and s are abbreviations for cosine and sine.

F_B consists of propeller lift force and induced drag force which can be modeled by

$$F_B = -T_\Sigma \overrightarrow{b_3} - T_\Sigma D v_B \tag{3}$$

where T_Σ, v_B are the propeller lift force and velocity in body frame $\{B\}$; $D \in R^{3\times3}$ is the coefficient matrix [7] for induced drag.

The reading of accelerometer and the acceleration in inertial frame has the following relationship

$$Ra_{\mathrm{IMU}} + g\overrightarrow{a_3} = \dot{v} \Leftrightarrow a_{\mathrm{IMU}} = R^{\mathrm{T}}\left(\dot{v} - g\overrightarrow{a_3}\right). \tag{4}$$

By formula (1) and (4) we can write

$$ma_{\mathrm{IMU}} = F_B. \tag{5}$$

Recalling (3) we get

$$a_{IMU} = -\frac{T_\Sigma}{m}\overrightarrow{b_3} - \frac{T_\Sigma}{m} DR^{\mathrm{T}} v. \tag{6}$$

Then we gain the velocity which is expresses as

$$v = -\left(DR^{\mathrm{T}}\right)^{-1}\left(\frac{m}{T_\Sigma} a_{\mathrm{IMU}} + \overrightarrow{b_3}\right). \tag{7}$$

When in attitude-hold mode, the vertical direction is in a state of force balance as Eq. (8) expresses.

$$P_v\left(T_\Sigma R\overrightarrow{b_3}\right) = mg \Rightarrow \frac{m}{T_\Sigma} = \frac{P_v R\overrightarrow{b_3}}{g} \tag{8}$$

where P_{h} and P_{v} are defined as the projection matrix onto the x-y plane and z direction.

$$P_h = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad P_v = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}. \tag{9}$$

Substituting $\frac{m}{T_\Sigma}$ in (7) using Eq. (8), finally we obtain the horizontal translational velocity

$$v_h = -P_h \left(D R^{\mathrm{T}} \right)^{-1} \left(\frac{P_v R \overrightarrow{b_3}}{g} a_{\mathrm{IMU}} + \overrightarrow{b_3} \right) \tag{10}$$

where $v_{\mathrm{h}} = (v_x, v_y)$ is the horizontal velocity in inertial frame $\{\mathrm{A}\}$.

Equation (10) tells us that if we get an accurate attitude estimation of R and acceleration estimation of α_{IMU}, the velocity in altitude-hold mode can be theoretically solved without GPS or optical flow. But in practice, parameters in (10) are very hard to be obtained. In this paper, we view dynamic model (10) as a black box, and use machine learning method support vector machine (SVM) to train the model (10). Finally, use the trained model to predict velocity of the flying UAV in altitude-hold mode.

3 SVM Regression

The regression problem is to fit a straight line, a plane, or a super plane to approximate the distribution of given sample points.

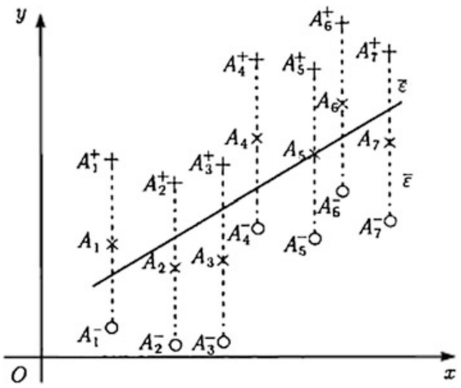
For given training dataset

$$T = \{(x_1, y_1), \ldots, (x_l, y_l)\} \in (R^n \times R)^l \tag{11}$$

where $x_i \in R^n, y_i \in R = \{1, -1\},\ i = 1, \ldots, l,$ the regression problem is to find function $g(x)$ which can predict y very well at any input x. A linear regression problem is to find linear function $g(x) = (w \cdot x) + b$ that can fit the samples best. Before detailed introduction of regression, we take a look at SVM linear classification.

SVM linear classification is to deal with following optimization [8]:

$$\begin{aligned} &\min_a \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \left(x_i \cdot x_j \right) a_i a_j - \sum_{j=1}^{l} a_j \\ &\text{s.t} \sum_{i=1}^{l} y_i a_i = 0, \quad a_i \geq 0, \quad i = 1, \ldots, l \end{aligned} \tag{12}$$



y
x
O
A_1^+ A_2^+ A_3^+ A_4^+ A_5^+ A_6^+ A_7^+
A_1 A_2 A_3 A_4 A_5 A_6 A_7
A_1^- A_2^- A_3^- A_4^- A_5^- A_6^- A_7^-
$\bar{\varepsilon}$
$\bar{\varepsilon}$

Fig. 1 The method to create two separate parts using training dataset

It can be proved that Eq. (12) has a solution $\alpha^* = \left(a_1^*, \ldots, a_l^*\right)^{\mathrm{T}}$, so the SVM classification problem has solution [9]:

$$w^* = \sum_{i=1}^{l} a_i^* y_i x_i \quad \text{and} \quad b^* = y_i - \sum_{i=1}^{l} a_i^* y_i (x_i x_j) \tag{13}$$

SVM tries to transform linear regression problem to linear classification problem. The basic idea is to create two separate parts by moving each sample in the training set a distance ε along the positive and negative y direction, as shown in Fig. 1.

We denote the two separate parts by $D^+ = \left\{\left(x_i^{\mathrm{T}}, y_i + \varepsilon\right)^{\mathrm{T}}, i = 1, \ldots, l\right.$ and $D^- = \left\{\left(x_i^{\mathrm{T}}, y_i - \varepsilon\right)^{\mathrm{T}}, i = 1, \ldots, l\right.$. Now our goal has been changed to be searching a function to separate the two parts which is exactly the classification problem described as following:

For given classification training set: $\left\{\left(\left(x_1^{\mathrm{T}}, y_1 + \varepsilon\right)^{\mathrm{T}}, 1\right), \ldots, \left(\left(x_l^{\mathrm{T}}, y_l + \varepsilon\right)^{\mathrm{T}}, 1\right), \left(\left(x_1^{\mathrm{T}}, y_1 - \varepsilon\right)^{\mathrm{T}}, -1\right), \ldots, \left(\left(x_l^{\mathrm{T}}, y_l - \varepsilon\right)^{\mathrm{T}}, -1\right)\right\}$ }, the aim is to solve the following optimization problem [10]:

$$\begin{aligned} \min_{a^* \in R^{2l}} \frac{1}{2} \sum_{i,j=1}^{l} \left(a_i^* - a_i\right)\left(a_j^* - a_j\right)\left(x_i x_j\right) \\ + \varepsilon \sum_{i=1}^{l} \left(a_i^* + a_i\right) - \sum_{i=1}^{l} y_i \left(a_i^* - a_i\right), \\ \text{s.t} \quad \sum_{i=1}^{l} \left(a_i^* - a_i\right) = 0, \quad a_i^* \geq 0, \quad i = 1, \ldots l \end{aligned} \tag{14}$$

which leads to regression problem solution as shown below:

$$w = \sum_{i=1}^{l} \left(a_i^* - a_i\right) x_i, \quad b = y_i - \left(w \cdot x_j\right) + \varepsilon \tag{15}$$

where $\alpha^* = \left(a_1, a_1^*, \ldots, a_l, a_l^*\right)^{\mathrm{T}}$ is the solution of problem (14). Finally, the estimate function $g(x)$ has the form

$$y = (w \cdot x) + b = \sum_{i=1}^{l} (a_i^* - a_i)(x_i \cdot x) + b \tag{16}$$

4 Method

This section describes the method of motion prediction based on machine learning.

4.1 Data Preprocessing

The experiment data includes the gyroscope data, accelerometer data, optical flow data and GPS data. Before training, preprocessing includes flow value filtering, transformation from quaternions to Tait-Bryan angle and rotation matrix, flow velocity conversion between inertial and airframe coordinate need to be done first.

The jittering of UAV when it is flying causes glitch in flow values. We can use a low-pass filter to eliminate it, namely:

$$y(n+1) = w * x(n) + (1 - w) * y(n) \tag{17}$$

where x is the input of filter, y is the output, w is a coefficient to adjust the smoothness of filtering.

As attitude is represented by quaternions in most flight control units, we need to convert it to Tait-Bryan angle roll pitch yaw. The transformation formula shows as below.

$$\begin{aligned} \text{Roll} &= a\tan 2(2 * w * x + 2 * y * z, 1 - 2 * x^2 - 2 * y^2) \\ \text{Pitch} &= a\sin(2 * w * y - 2 * z * x) \\ \text{Yaw} &= a\tan 2(2 * w * z + 2 * x * y, 1 - 2 * y^2 - 2 * z^2) \end{aligned} \tag{18}$$

The conversion between quaternions and rotation matrix is

$$R = \begin{bmatrix} 1 - 2*(y^2 + z^2) & 2*(x*y - w*z) & 2*(x*z + w*y) \\ 2*(x*y + w*z) & 1 - 2*(x^2 + z^2) & 2*(y*z - w*x) \\ 2*(x*z - w*y) & 2*(y*z + w*x) & 1 - 2*(x^2 + y^2) \end{bmatrix}. \tag{19}$$

Now we can use R to get flow velocity in inertial frame $\{A\}$:

$$\mathrm{vel}_A = R*(\mathrm{flow\_}x, \mathrm{flow\_}y, 0)^{\mathrm{T}} \tag{20}$$

4.2 Training Method

The training features are chosen as the sum of roll, pitch, yaw and acceleration from all frames during a short time span t. And the output value is the flow velocity in inertial coordinate. In our experiment, *t* is 1 s.

In this paper, the SVM regression is implemented by Libsvm tool [11]. The training process includes the following steps: training vector formatting, selection of kernel function, parameter tuning and training dataset choosing.

For training vector formatting, due to the format requirement of Libsvm, the input training vector must meet the format

$$\langle \mathrm{label}\rangle\langle \mathrm{index1}\rangle\langle \mathrm{value1}\rangle\langle \mathrm{index2}\rangle\langle \mathrm{value2}\rangle\ldots$$

where ⟨label⟩ is the output value, ⟨value⟩ is the feature and ⟨index⟩ must start from 1.

Before selection of kernel function, we take a look at formula (10). We know that trigonometric function can be written as Euler formula as shown below.

$$\cos(\theta) = \frac{\mathrm{e}^{j\theta} + \mathrm{e}^{-j\theta}}{2} \tag{21}$$

Due to the trigonometric function of attitude angel in rotation matrix *R*, an intuition is that the kernel function may have form like exponential function which leads us to choosing the radial basis function as our kernel.

$$\mathrm{Kernel} = \mathrm{e}^{-\mathrm{gamma}*|u-v|^2} \tag{22}$$

Experimental results show that radial basis function has higher estimate accuracy than other kernels, which confirms our previous conjecture. Table 1 lists the four kinds of kernel functions provided by Libsvm tool. We used four kernels alternately

Table 1 The kernel function lists

Kernel 0	Linear: $\mathbf{u}'*\boldsymbol{v}$
Kernel 1	Polynomial: $(\text{gamma} * u' * v + \text{coef0})^{\text{degree}}$
Kernel 2	Radial basis: $\exp(-\text{gamma}*\lvert \text{u} - \text{v}\rvert^{2})$
Kernel 3	Sigmoid: $\tanh(\text{gamma}*\text{u}' * v + \text{coef0})$

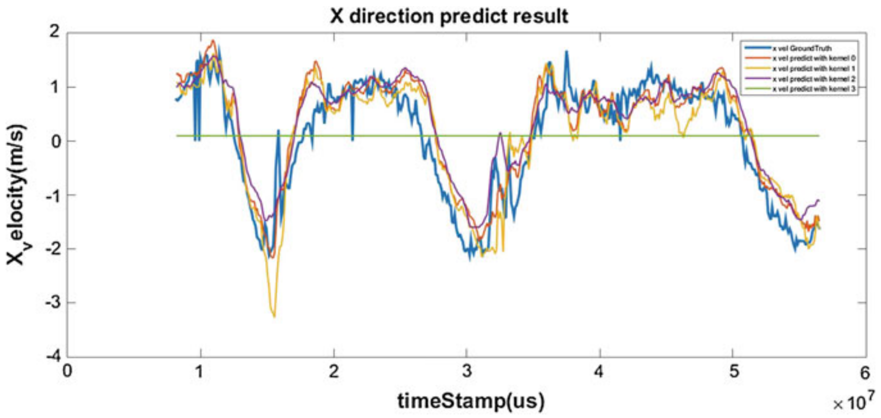


Fig. 2 The comparison of estimation result of four kernels

to train with the same training dataset and obtain corresponding estimation models and use them for prediction with the same test dataset. The result is shown in Fig. 2.

Table 2 also displays the iteration number, STD (standard deviation), SCC (square correlation coefficient) of training result using each kind of kernel.

Considering overall performance of iteration, STD and SCC, we chose the radial basis function as the final kernel.

Parameter tuning is also a very important issue in training process. Table 3 lists the main parameter used in training.

Table 2 The comparison of four kernels predict result

Kernel functions	Iteration	STD (m/s)	SCC
Kernel 0	41,730	0.4811	0.7884
Kernel 1	380,851	1.0782	0.2472
Kernel 2	831	0.4781	0.8211
Kernel 3	424	1.0356	1.1566×10^{-9}

Table 3 The list of main parameters

-*c* cost	Set cost value, default: 1
-*p* epsilon	Set ε in loss function, default: 0.1
−*e* tolerance	Set the termination condition, default: 0.001

Table 4 The process of parameters selecting

参数值	SCC	参数值	SCC	参数值	SCC
-c 0.1	0.77450	*-p* 0.050	0.81640	*-e* 0.0005	0.82103
-c 0.5	0.80054	*-p* 0.100	0.82018	*-e* 0.0010	0.821051
-c 1	–	*-p* 0.150	0.82074	*-e* 0.0015	0.821028
-c 1.5	0.81842	*-p* 0.175	0.821051		
-c 2	0.81962	*-p* 0.2	0.82972		
-c 2.5	0.82018				
-c 3	0.81974				

Tuning parameter is like this. We keep other parameters constant and change the target parameter a little. Step by step, we compare estimation results. Finally, we choose the value that produces the most accurate estimation. Table 4 shows the process of parameter tuning.

Notice that we only use training dataset in parameter tuning.

To choose training dataset, in the experiment, we train the regression models based on each training dataset and use each model to estimate the same test dataset. Finally compare corresponding estimation result and choose the training dataset with the best estimation result.

5 Experiment and Result Analyze

5.1 Hardware Platform

Our experiment is based on a multi-rotor aircraft platform. Figure 3 shows the hardware structure diagram. It mainly consists of pixhawk, odroid, and some other

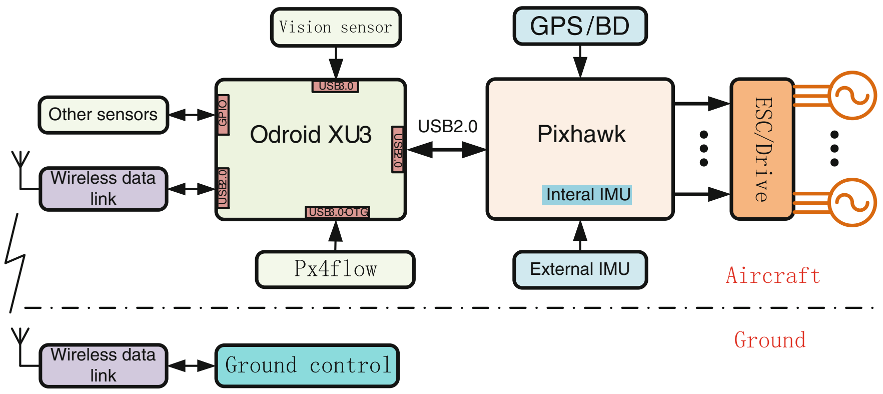


Fig. 3 Hardware structure diagram

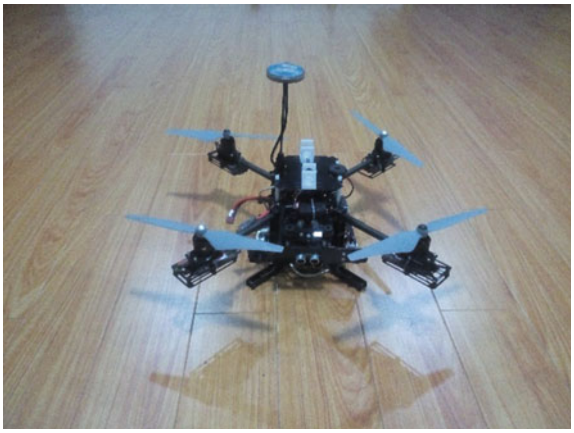


Fig. 4 The multi-rotor aircraft used in experiment

sensors. Pixhawk based on Cortex-M4 processor is a high-performance autopilot-on-module for multi-rotors and any other robotic platform. Odroid XU3 is the world's first single-board computer with a heterogeneous multi-Processing (HMP). In our experiment, the odroid is used to collect data from all kinds of sensors including optical flow sensor, GPS, accelerometer, gyroscope, and so on. The p×4flow is an optical flow smart camera which has a native resolution of 752 × 480 pixels and calculates optical flow on a 4× binned and cropped area at 400 Hz. All sensors data can be read into odroid and sent by Wi-Fi to ground control station where the training datasets come from. Figure 4 shows the multi-rotor aircraft used for collecting data.

5.2 Data Collection

Data we need to collect include gyroscope, accelerometer, optical flow, and GPS data. When collecting data, it is required to keep UAV at a constant height, and ensure that the flight action is rich enough. For instance, you can flight along a circle path at a fixed height. We also need to collect datasets in various environments including GPS and optical flow are both available, GPS is available but optical flow is not, optical flow is available while GPS is not.

The first kind of environment needs bright light, rich ground textures and available GPS signal. The dataset collecting in this environment is used to train the estimation model, in which the feature vector contains accelerometer and gyroscope values and the output vector is velocity calculated from GPS or optical flow. The second environment contains dull light that will cause optical flow sensor producing significant error. The data from second environment is used to test the estimation result when optical flow is unavailable based on some specified trained model. The third environment has rich ground texture and bright light while GPS is unavailable.

The data is used to test the estimation result when GPS is unavailable. In order to ensure the sufficient of training and testing dataset, the acquisition time is set to be larger than 3 min in our experiment.

5.3 Training Data Checking

In our training process, the output vector is the optical flow velocity in inertial frame. Because of the sensitivity to environment of optical flow value, there is a high probability that optical flow value has significant error. On the other hand, the attitude angle yaw is influenced by magnetometer fixed on UAV body. If there is a certain intensity of magnetic field in environment which causes a significant effect on magnetometer fixed on UAV, the yaw value will have significant error too.

The way to check training data is to plot the optical flow integration path using flow value and attitude angel yaw, and compare it with path plotted from GPS value, as shown in Fig. 7. If the two paths are consistent, it means optical flow and yaw value have small error, otherwise, it means there is significant error in flow and yaw value and they cannot be used as training dataset. Because the value from optical flow is a two-dimensional vector in body frame, we need to transform it to that in inertial frame using a two-dimensional rotation matrix calculated from angle yaw. Then we integrate the flow value in inertial frame to get the flight path. Finally, we plot the GPS path as the ground-truth path and compare it with optical flow path.

5.4 Result and Analyse

Finally, we use svm-predict function provided by Libsvm to estimate velocity based on estimation model. In our experiment, the ground-truth data is optical flow velocity. We compared it with the estimated velocity to evaluate the performance of our method. In addition, we plotted the ground-truth velocity integration path and estimated velocity integration path, and compared them for more intuitive result.

Figures 5 and 6 show the comparison of ground-truth velocity and estimated velocity using our method.

Table 5 lists the STD and SCC between predicted velocity and ground-truth.

Figure 7 shows the optical flow integration path and GPS path for checking whether optical flow value is available.

Figure 8 shows the flow path and predicted velocity integration path from which we can see the obvious path drift due to the velocity estimation error. But it should be noticed that ground-truth and estimated path have consistent path shape change which tell us our method is reasonable and feasible.

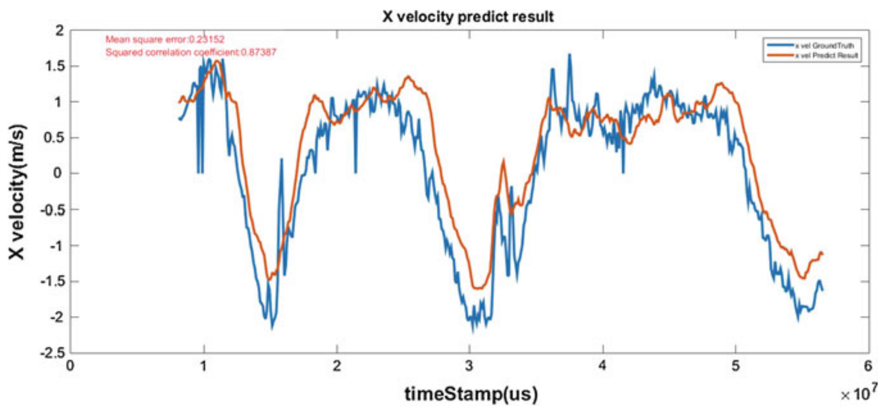


Fig. 5 The *x* direction predicts result (*yellow*) and ground-truth velocity (*blue*)

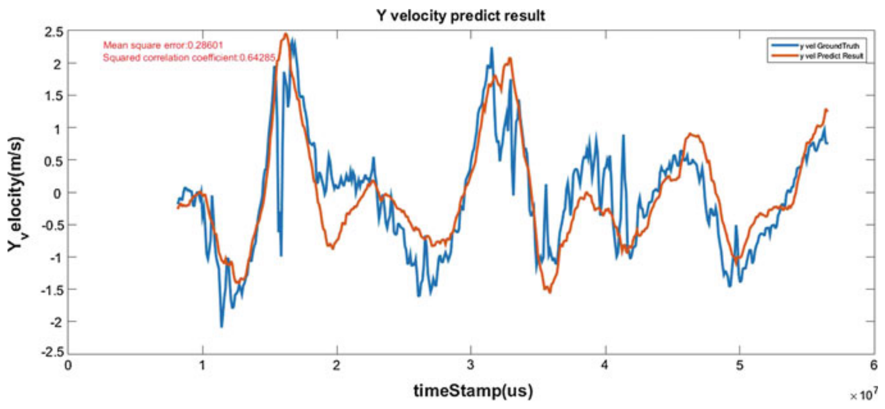


Fig. 6 The *y* direction predicts result (*yellow*) and ground-truth velocity (*blue*)

Table 5 The STD and SCC of X/Y velocity

	STD (standard deviation)	SCC (squared correlation coefficient)
X方向	0.4811 m/s	0.8739
Y方向	0.5348 m/s	0.6429

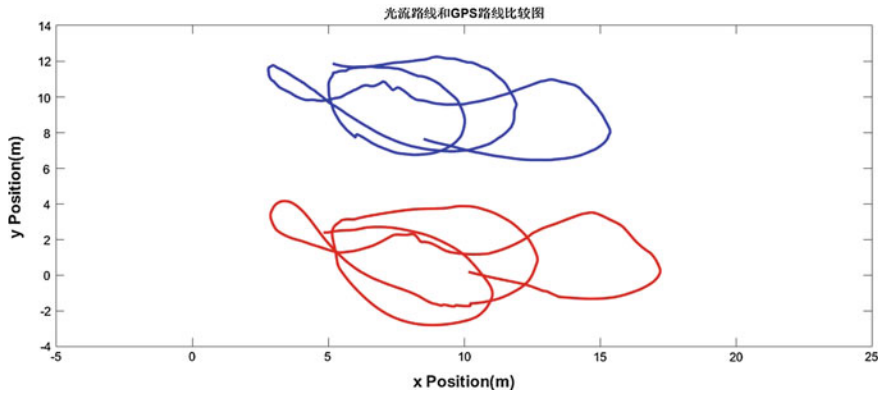


Fig. 7 The p×4flow position (*blue*) and GPS position (*yellow*)

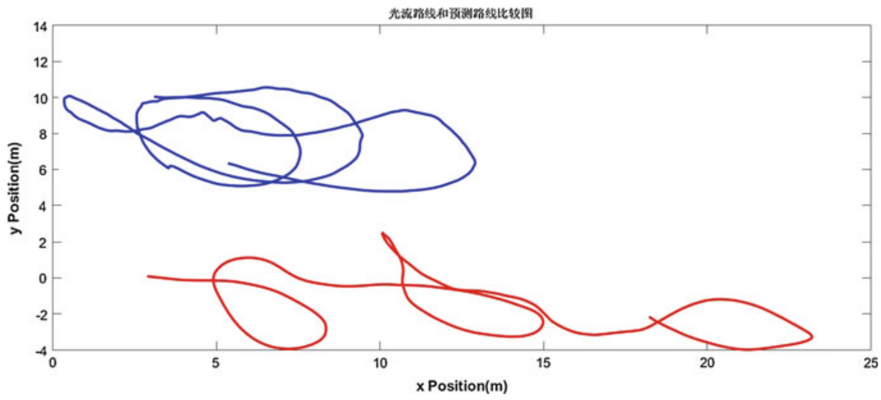


Fig. 8 The p×4flow position (*blue*) and predict position (*yellow*)

6 Conclusion

In this paper, we propose a method based on machine learning to estimate velocity of multi-rotor UAV from inertial measurements without using integration method as in a traditional inertial navigation system. Due to the input feature vector contains only accelerometer and gyroscope readings, our estimation method can work even when GPS and optical flow are both unavailable. This method can be used to assist navigation when the UAV flies into bad environment.

We carried out the experiment in attitude-hold flight mode. We choose optical flow velocity as ground-truth. The result shows that the correlation degree between estimation velocity and Ground-Truth is 0.8739 at X direction and 0.6429 at Y direction. In addition, we plot the estimation path and ground-truth path for comparing intuitively.

In the future research, we will extend our work to free flight mode and wind resisting flight mode, to make our method more flexible to different applications.

Acknowledgment This work is supported by the National Natural Science Foundation of China under Grant 61573242, in partly by the Shanghai Science and Technology Committee under Grant 14511100300, 15511105100 and partly sponsored by Shanghai Pujiang Program (No. 14PJ1405000) and Qingpu Industry-University-Research Project (2015-4).

References

1. Yang M (2010) Research on small strapdown inertial navigation system. Central South University
2. Titterton D, Weston J (2005) Strapdown inertial navigation technology—2nd edition (book review). IEEE Aerosp Electron Syst Mag 20(7):33–34
3. Honegger D, Meier L, Tanskanen P et al (2013) An open source and open hardware embedded metric optical flow CMOS camera for indoor and outdoor applications. In: 2013 IEEE international conference on robotics and automation (ICRA). IEEE, pp 1736–1741
4. Hrabar S, Sukhatme GS, Corke P et al (2005) Combined optic-flow and stereo-based navigation of urban canyons for a UAV. In: 2005 IEEE/RSJ international conference on intelligent robots and systems, (IROS 2005). IEEE, pp 3309–3316
5. Barber DB, Griffiths SR, Mclain TW et al (2007) Autonomous landing of miniature aerial vehicles. J Aerosp Comput Inf Commun 4(5):770–784
6. Romero H, Salazar S, Lozano R (2009) Real-time stabilization of an eight-rotor UAV using optical flow. IEEE Trans Rob 25(4):809–817
7. Mahony R, Kumar V, Corke P (2012) Multirotor aerial vehicles: modeling, estimation, and control of quadrotor. IEEE Robot Amp Amp Autom Mag 19(3):20–32
8. Platt J (1999) Fast training of support vector machines using sequential minimal optimization. Adv Kernel Methods Support Vector Learn 3
9. Cheng X (2012) Study on geometric algorithm of linear optimization problems with equality constraints. Fujian Normal University
10. Deng N, Tian Y (2009) The theory, algorithm and expansion of support vector machine. Science Press, Beijing
11. Chang CC, Lin CJ (2015) LIBSVM—a library for support vector machines. http://www.csie.ntu.edu.tw/∼cjlin/libsvm/. Accessed 8 Dec 2015