2014 IEEE International Conference on Systems, Man, and Cybernetics
October 5-8, 2014, San Diego, CA, USA

6-DoF Velocity Estimation Using RGB-D Camera Based on Optical Flow

Pyojin Kim, Hyon Lim, and H. Jin Kim
School of Mechanical and Aerospace Engineering
Seoul National University, Seoul, South Korea
Email: rlavywls@snu.ac.kr, hyonlim@snu.ac.kr, and hjinkim@snu.ac.kr

Abstract—**In this paper, we suggest a new 6-DoF velocity estimation algorithm using RGB and depth images. Autonomous control of mobile robots requires their velocity information. There exist numerous researches on estimating and measuring the velocity. However, more investigations are needed related to vision sensors and depth image. In this work, we propose an algorithm for velocity estimation with an RGB-D sensor based on image jacobian matrix usually used in image-based visual servoing. We validate the performance of the proposed estimation algorithm in various environments with the RGB-D benchmark dataset. The velocity estimation results show the high quality of estimated 6-DoF velocity compared to the ground truth velocity.**

I. Introduction

In recent years, UAVs (Unmanned Aerial Vehicles) receive much attention from companies, researchers, and the public. However, for fully autonomous flight and execution of various control, there remain many crucial technical factors to be resolved. Among them, one of the most significant factors is accurate state estimation. For example, many UAVs obtain their location and 6-DoF velocity information based on GPS in outdoors [1]. While in indoors, artificial markers are attached to the UAVs and get accurate states from motion capture systems, e.g. Vicon system. GPS, however, cannot always be the solution because of its sensitivity and GPS-denied environments. Also motion capture systems in practical life may not be suitable due to their high cost and the need for artificial markers. These limitations caused much attention on state estimation using only onboard and exteroceptive sensors.

In [2], [3], [4], the authors investigate the problem of estimating the linear velocity of a quadrotor using inertial measurement unit (IMU) including accelerometer and gyroscope sensor. Velocity estimation is performed in these researches based on the principle that accelerometer values along x and y axes are proportional to the linear velocity of quadrotor with respect to the quadrotor frame. They utilize the quadrotor dynamic model with the drag force term which reflects the blade flapping effect, detailed in [5]. But this estimation technique can be applied only to the UAVs that produce lift force using rotor blades.

There have been numerous studies on velocity estimation using video inputs. Estimation of velocity using visual information is performed in [6] based on optical flow. This method, however, can estimate not the true quantity of velocity but only the ratio of velocity. Those velocity estimations are ambiguous in scale because of a single camera problem. So, additional estimators are required for true metric scale. To know this absolute scale, an air pressure sensor is employed in [7] and a range sensor is used in [8]. Stereo vision also resolves this scale problem as it provides disparity (depth) map information at each frame using triangulation [9], [10]. However, this depth image computation requires heavy image processing at every step, which is not available on micro aerial vehicles.

Recently, RGB-D sensors such as Microsoft Kinect have become popular. In [11], RGB-D data are combined with IMU measurements to estimate the odometry using extended Kalman filter (EKF). It is used to estimate velocity based on rotation matrix R and translation vector T calculated using the singular value decomposition (SVD) [12]. However, the accuracy of the velocity estimation and computation time need to be further discussed. In [13] and [14], visual odometry is performed with an RGB-D sensor through a standard stereo visual odometry method. The motion of the stereo camera, R and T, can be firstly estimated by Horn's absolute orientation method [15]. Nonlinear least-squares solver is then utilized to minimize feature reprojection error [16] and the estimated results are combined with simultaneous localization and mapping (SLAM) for autonomous indoor flight. Many studies deal with SLAM which estimates the position of the vehicle for navigation purposes [14], [17]. However, SLAM requires much computation time which is not feasible for autonomous MAVs. Furthermore, to make MAVs stable in terms of their horizontal position, linear velocity estimation is essential [18].

In order to address the problems mentioned above, we propose new, simple but accurate method for velocity estimation with an RGB-D sensors based on image jacobian matrix. Our proposed algorithm does not need any additional sensors to know the absolute scale and sophisticated estimators to estimate the linear and angular velocity. Calculation process of our velocity estimation method is simple and easy, which allows the implementation using much less computation power than a standard stereo visual odometry method and SLAM framework.

This paper is organized as follows: our proposed velocity estimation algorithm is described in Section 2. Section 3 gives an overview of the velocity estimation system and its data flow scheme. After the experimental results are presented in Section 4, the paper remarks conclusion and future works in Section 5.

II. Vision-Based Velocity Estimation Algorithm

We explain how the proposed method works to estimate velocity in detail. In Section A, we explain the methods used

978-1-4799-3840-7/14/$31.00 ©2014 IEEE

to detect and extract the feature points in RGB images. We cannot utilize these detected feature points as they are, because there could be some mis-matched results between feature points. So, Section B describes how we remove the outliers of matched results. This procedure is essential to achieve robust data association between images. After that, we discuss the velocity estimation algorithm for the RGB-D camera based on these matched results (i.e. inliers) and depth values from depth images in Section C. Finally in Section D, post-filtering processing we used is described, which can reduce noisy values and abrupt changes in the estimated camera velocity.

A. *Feature point detecting and tracking*

First, Shi & Tomasi detector [19] is employed for detecting N feature points initially. Then, to track these detected feature points constantly, pyramidal optical flow with Lucas-Kanade (LK) feature tracker [20] is used. As a result of processing optical flow with LK method, we can obtain the tracked feature points from the previous detected feature points. The number of well-tracked feature points, however, decreases rapidly while the camera is moving continuously. We design the algorithm so that if the number of the tracked feature points is less than η, we add newly detected feature points found by Shi & Tomasi detector to have total N tracked and detected feature points. η should be determined less than N, the number of initially detected feature points. This is described in Algorithm 1. Through this technique, at least η tracked feature points can survive all the time.

B. *Robust data association process*

We already have putative matched results from the previous step of computing optical flow with LK feature tracker. But these matched results, of course, are not accurate and inconsistent because of some image noise, blurry images, etc. We solve this problem by employing the adaptive RANSAC procedure with an assumption that there must be a mathematical projective transformation model between each image. First, we randomly pick four matched sample pairs in both image frames. Based on these points, we can calculate the homography matrix using the normalized direct linear transform (DLT) algorithm described in [21]. In the next step, we fit the homography matrix in eight dimensions with an adaptive version [22] of RANSAC [23]. The adaptive term in RANSAC is the number of repetition of RANSAC process. As a consequence, the projective transformation matrix obtained from the above processes effectively eliminates the outliers of putatively matched results. Matched results of the authentic feature points can exist only through this type of correspondence process.

C. *Algorithm of velocity estimation*

There exists a kinematic relationship [24] between the velocity of feature points in the normalized image plane of the RGB-D camera and the velocity of the camera itself. That relationship and overall problem situations are depicted in Fig. 1.

$$V_c = J^{\dagger} V_{fp} \tag{1}$$

$$V_c = \begin{bmatrix} u & v & w & p & q & r \end{bmatrix}^T \in \mathbb{R}^6$$

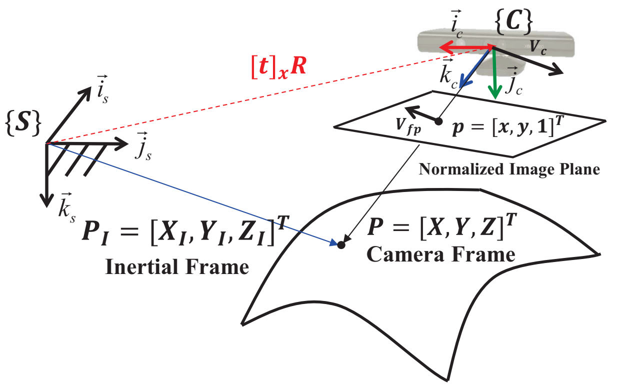

Fig. 1. **The kinematic relationship between velocities.** Eq. (1) is easily understood with the above configuration. Velocity of a feature point (V_{fp}) in the normalized image plane and velocity of the camera (V_c) with respect to the camera frame are closely connected through the image jacobian matrix ($J^{\dagger}$)

$$J \in \mathbb{R}^{2n \times 6}, V_{fp} \in \mathbb{R}^{2n}$$

$$J = \begin{bmatrix} \frac{-1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & \frac{-1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{bmatrix} \tag{2}$$

In Eq. (1), V_c denotes the spatial velocity of the RGB-D camera with respect to the camera frame. It consists of instantaneous linear and angular velocity of the camera frame where u, v, w and p, q, r represent the components of instantaneous linear and angular velocity in the $\vec{i}_c$, $\vec{j}_c$ and $\vec{k}_c$ axes, respectively. J is the matrix called *Image Jacobian Matrix* or *Interaction Matrix*, which is the mapping from the velocity of feature points to the velocity of the camera frame. Image Jacobian matrix is composed of x, y and Z where x, y are the location of the feature point and Z is depth value of that feature point extracted by the depth image. $J^{\dagger}$ denotes Moore-Penrose pseudo inverse of image Jacobian matrix J and n means the number of well-tracked feature points. V_{fp} denotes the velocity of the feature points in the normalized image plane of the camera. It should be noted that the matrix J is constructed from the values which we already know because the location of feature points and the depth values can be extracted by the RGB-D sensor. V_{fp} also can be calculated using the location of the feature points with the difference equation of Euler method. The time difference is chosen as the time gap between each image sequence. Therefore, as the right hand side of Eq. (1) is all known and if there exist at least three tracked feature points, the spatial velocity of the camera frame can be deduced through this simple algebraic calculation.

D. *Post-filtering processing*

Due to vibration of camera, existence of image noise, etc, the estimated spatial velocity of camera frame directly calculated from Eq. (1) is not accurate enough to be used as the authentic spatial velocity value. Deduced spatial velocity exists together with some noise. So, we have to put the raw estimated spatial velocity of the camera through post-filtering processing rather than just use as it is. Here, we use a linear Kalman filter method [25].

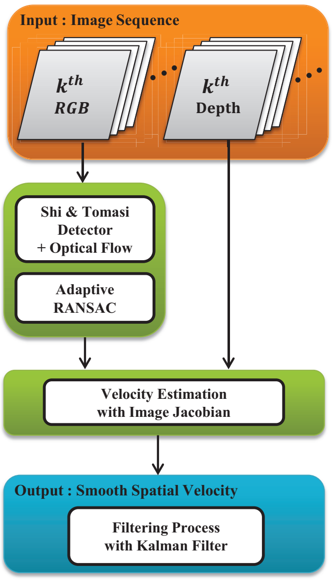
Input : Image Sequence
k^{th} RGB
k^{th} Depth
Shi & Tomasi Detector + Optical Flow
Adaptive RANSAC
Velocity Estimation with Image Jacobian
Output : Smooth Spatial Velocity
Filtering Process with Kalman Filter

Fig. 2. **System structure.** The overall structure of the velocity estimation method and flow of data are drawn. Initially, keypoint detector is run and the detected feature points are tracked with optical flow. Outliers of tracked feature points are removed by adaptive RANSAC. Based on well-tracked feature points and depth information, raw estimated velocity of camera is obtained. Lastly, smooth velocity of camera can be deduced from the Kalman filtering process.

$$\hat{X}_k^- = F\hat{X}_{k-1}^+ + w \tag{3}$$

$$z_k = HX_k + v \tag{4}$$

$$w \sim N(0, Q) \quad v \sim N(0, R)$$

$$\hat{X}_k^+ = \hat{X}_k^- + K_k(z_k - H\hat{X}_k^-) \tag{5}$$

In Eq. (3), state variable X is composed of linear and angular velocity of the camera with respect to the camera frame, which means $X = [u, v, w, p, q, r]^T$. We choose as the process model the constant velocity model, which means $F = I_{6\times6}$. w denotes the process noise used to consider the unmodeling part in the process model. In Eq. (4), the observation model matrix is determined as $H = I_{6\times6}$. v means the observation noise that we want to remove. w and v are white, zero-mean, uncorrelated, and have known covariance matrix Q and R, respectively. z is the measurement value; the raw estimated spatial velocity directly calculated from Eq. (1) in this case. K_k is the optimal Kalman gain. States are predicted and updated constantly by the above equations. As a result, the spatial velocity of the camera frame with contamination and some noise can be converted into the accurate spatial velocity of the camera frame through this filtering process.

III. System Overview

The whole system structure and data flow scheme are represented in Fig. 2 and Algorithm 1. The input of this system is video and depth images from the RGB-D camera in Fig. 3. First, we detect the feature points with Shi & Tomasi detector and the optical flow with Lucas-Kanade method is used to track them constantly in sequence of images. After that, to eliminate the outliers of tracking results, adaptive RANSAC method is employed. The tracked feature points and depth values of that points are utilized simultaneously to construct image Jacobian matrix J and the raw estimated camera velocity is calculated. Kalman filter process is performed to get clean and smooth spatial velocity of the camera frame. In the end, this system generates the current estimated 6-DoF spatial velocity of the camera with respect to the camera frame.

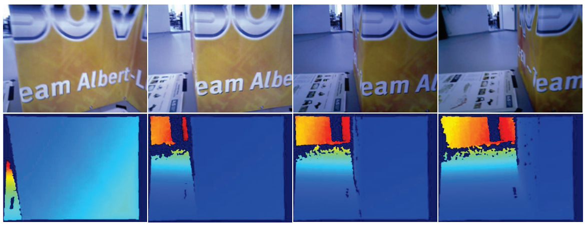

Fig. 3. **RGB (top row) and depth (bottom row) image from an RGB-D sensor.** The input RGB-D data to the proposed velocity estimation algorithm are depicted the above.

Algorithm 1 Velocity Estimation Algorithm

Input: Sequence of images & depth images
Output: Velocity of camera

```
run Shi & Tomasi detector
to obtain $N$ feature points ← initialization
for  Video input exists  do
    track feature points ← Optical flow with LK
    if  # of tracked feature points $< \eta$  then
        add detected feature points generated
        by Shi & Tomasi detector
    end if
    outliers are removed ← Adaptive RANSAC
    estimate camera velocity ← by Eq. (1)
    pass Kalman filter ← by Eq. (3),(4),(5)
end for
Velocity estimation process terminates.
```

IV. Results

In this section, we show experimental results of the proposed velocity estimation method. The experiments are performed based on the TUM RGB-D benchmark dataset [26] which consists of RGB and depth images at full frame rate (30 Hz) and sensor resolution (640×480). They also provide the ground truth of position, posture of camera frame. We present the comparison of our estimated spatial velocity with the ground truth spatial velocity using root mean square error (RMSE).

A. Setup

There exist many dataset scenarios and categories in the TUM RGB-D dataset. To test and prove performance of our proposed velocity estimation method, we experiment in many different kinds of environments of the TUM dataset. Several scenarios of each category are adopted to check the effect of the presence or absence of structure, object, and texture in

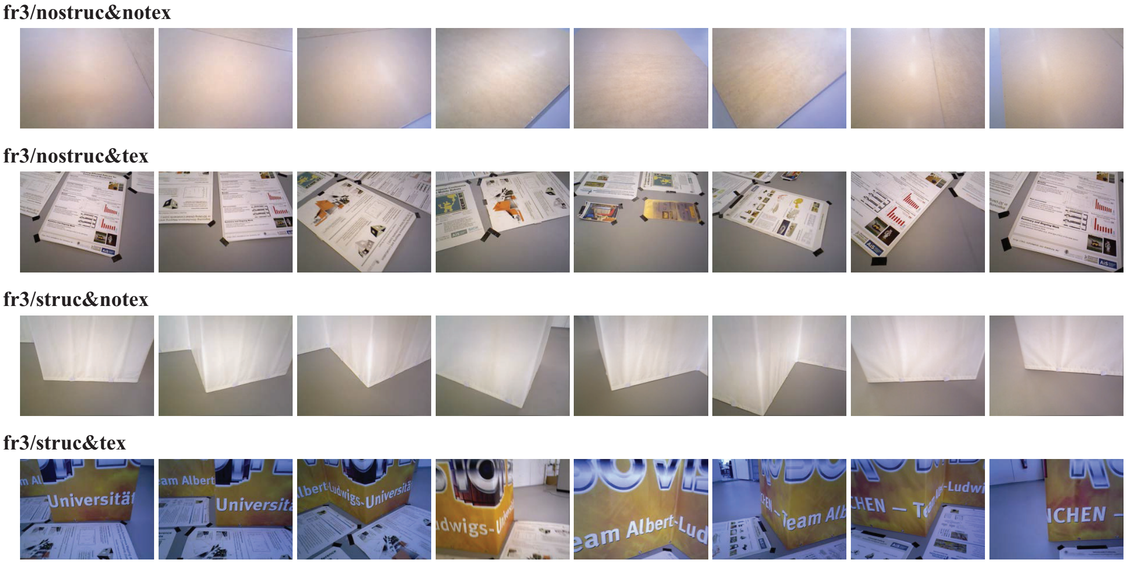

Fig. 4. **Image streams used from the TUM RGB-D dataset.** A series of images of the top four datasets in TABLE I is illustrated to show how different the presence and absence of structure and texture are in images. The best two experimental results are obtained using fr3/nostruc&tex and fr3/struc&tex as emphasized in TABLE I. On the other hand, the two worst results occur in fr3/struc&notex and fr3/nostruc&notex. The difference of existence of texture causes these experimental results.

TABLE I. PERFORMANCE OF EXPERIMENTAL RESULTS 1

Name of Dataset	Root Mean Square Error [m/s, rad/s]						Average Error		# of Images	Length [m]	Computation time [s]	Duration [s]
	v_x	v_y	v_z	w_x	w_y	w_z	v [m/s]	w [rad/s]				
fr3/nostruc&notex	0.370	0.356	0.248	0.305	0.369	0.227	0.325	0.300	1093	11.739	109.45	37.88
fr3/nostruc&tex	**0.043**	**0.030**	**0.044**	**0.056**	**0.049**	**0.057**	**0.039**	**0.054**	**1639**	**13.456**	**52.42**	**56.57**
fr3/struc&notex	0.453	0.417	0.497	0.362	0.461	0.141	0.456	0.322	1054	3.872	44.03	36.50
fr3/struc&tex	**0.028**	**0.021**	**0.025**	**0.045**	**0.039**	**0.037**	**0.025**	**0.040**	**1057**	**5.05**	**32.38**	**36.99**
fr1/floor	0.054	0.061	0.063	0.062	0.100	0.078	0.059	0.080	1227	12.569	75.63	46.34
fr3/longoffice	**0.067**	**0.047**	**0.053**	**0.049**	**0.053**	**0.048**	**0.055**	**0.050**	**1900**	**21.455**	**56.33**	**66.69**
fr1/plant	0.080	0.157	0.097	0.161	0.111	0.157	0.111	0.143	1126	14.795	42.63	37.97
fr1/teddy	0.128	0.164	0.111	0.198	0.195	0.167	0.134	0.186	1401	15.709	55.66	47.26

TABLE II. PERFORMANCE OF EXPERIMENTAL RESULTS 2

Name of Dataset	Root Mean Square Error [m/s, rad/s]						Average Error		# of Images	Length [m]	Computation time [s]	Duration [s]
	v_x	v_y	v_z	w_x	w_y	w_z	v [m/s]	w [rad/s]				
fr3/nostruc&notex	0.450	0.351	0.501	0.262	0.426	0.236	0.434	0.309	1093	11.739	69.67	37.88
fr3/nostruc&tex	**0.035**	**0.029**	**0.031**	**0.055**	**0.044**	**0.057**	**0.032**	**0.052**	**1639**	**13.456**	**46.76**	**56.57**
fr3/struc&notex	0.539	0.499	0.492	0.485	0.546	0.191	0.510	0.408	1054	3.872	44.35	36.50
fr3/struc&tex	**0.040**	**0.032**	**0.073**	**0.073**	**0.051**	**0.039**	**0.049**	**0.054**	**1057**	**5.05**	**31.07**	**36.99**
fr1/floor	0.051	0.064	0.068	0.053	0.092	0.074	0.061	0.074	1227	12.569	39.29	46.34
fr3/longoffice	**0.066**	**0.046**	**0.057**	**0.047**	**0.052**	**0.048**	**0.057**	**0.050**	**1900**	**21.455**	**55.83**	**66.69**
fr1/plant	0.080	0.157	0.096	0.162	0.111	0.157	0.111	0.143	1126	14.795	42.15	37.97
fr1/teddy	0.132	0.146	0.112	0.193	0.197	0.166	0.130	0.185	1401	15.709	54.47	47.26

image stream. Image sequences illustrated in Fig. 4 help understand the terms of structure and texture. Their information and specifications are listed in TABLE I.

To compare and analyze a set of experimental results quantitatively, we choose the error metric as root mean square error (RMSE) defined as

$$\xi_{error} = \sqrt{\frac{\sum_{i=1}^{N_d} (\xi_{estimation.i} - \xi_{groundtruth.i})^2}{N_d}} \qquad (6)$$

Here, ξ denotes each element of the camera state vector $X = [u, v, w, p, q, r]^T$ (e.g. linear and angular velocity components) and N_d is the total number of data points.

We choose the initial number of feature points N as 300, and the minimum threshold of the number of tracked feature points η as 100. The process and observation noise covariance, Q and R, are assigned as diagonal matrices with the following elements, $0.001^2 \times I_{6\times6}$ and $[0.02^2, 0.017^2, 0.017^2, 0.015^2, 0.015^2, 0.015^2]$, respectively.

All calculations and processes are conducted on a desktop computer with Intel Core i5 with 3.2 Ghz. To detect and track feature points, we make a program utilizing Microsoft Visual Studio 2010 Release Win32 mode with OpenCV 2.4.4 version for Windows. Other processes including the adaptive RANSAC, velocity estimation, generation of the ground truth trajectory, and calculation of error are done on the MATLAB R2013b.

B. *Discussion*

As mentioned before, the ground truth values provided by the TUM RGB-D benchmark dataset are not the spatial velocity of the camera itself but the location and pose of the camera. So, by assuming that the ground truth value follows the constant acceleration model like [27], we can generate the reliable ground truth spatial velocity of the camera based on a standard Kalman filter Eq. (3)$\sim$(5). With this ground truth, we can evaluate the performance of the estimated camera velocity and compare our proposed method with the ground truth velocity quantitatively.

The total results of experiments including RMSE, computation time of each scenario are arranged in TABLE I. In particular, let us discuss the fr3/struc&tex experimental scenario to see the outcome of 6-DoF velocity estimation. In Fig. 5, intermediate steps of deducing velocity such as detecting, tracking, and robust data association are presented. The estimation results composed of the ground truth and the estimated spatial velocity are shown in Fig. 6. The estimated spatial velocity follows the ground truth stably without any divergence. The computation time of this scenario is about four seconds less than the duration time. But, as shown in Table I, the computation time for other scenarios exceeds the duration time except fr3/struc&tex, fr3/nostruc&tex, and fr3/longoffice. The main reason for this is the existence of enough texture in images in these three scenarios, which allows to easily detect and track the feature points. Limiting the iteration number of the adaptation slightly improves the computational time as shown in TABLE II. However, it sacrifices the accuracy of estimation as suggested by the increased RMS error. So, we can infer that the presence of sufficient texture is one of the most significant factors determining the accuracy of estimation and computation load of our proposed method.

V. Conclusion and Future Work

In this paper, we proposed a simple and accurate velocity estimation method using RGB and depth image stream. The Shi & Tomasi detector and optical flow with LK could find the feature points and track them constantly. To remove the outliers of the tracked feature points, adaptive RANSAC was used. Using image Jacobian matrix, finally, we could estimate the spatial velocity of the camera with respect to the camera frame. We showed the proposed approach was performed satisfactorily compared to the deduced velocity and the ground truth given by the benchmark RGB-D dataset. Our algorithm had many strong points in view of the accuracy of estimation and required computation time if there exists abundant textures in every image.

The disadvantages in our proposed algorithm may be the real-time problem and dependency of enough textures in images. The heaviness of computation load is quite obvious since the proposed method is designed to use and match every single image without taking into account the quality or condition of each image. To solve these problems, idea of representing the quality of image, keyframe selection mechanism and other dimensional reduction techniques will be necessary.

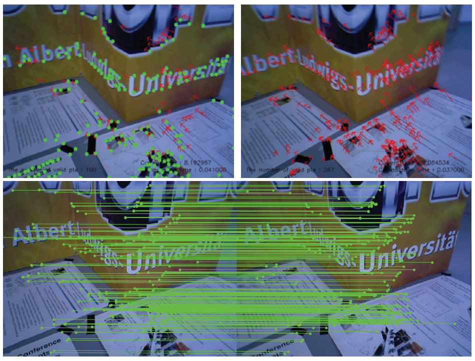

Fig. 5. **Image processing of fr3/struc&tex.** The top-left image demonstrates an example of how Shi & Tomasi detector works. The detected keypoints are displayed as green circles. The top-right image shows the keypoints tracking process using optical flow. Movements of the keypoints are drawn as red arrow. The bottom image is the result of robust data association process based on adaptive RANSAC.

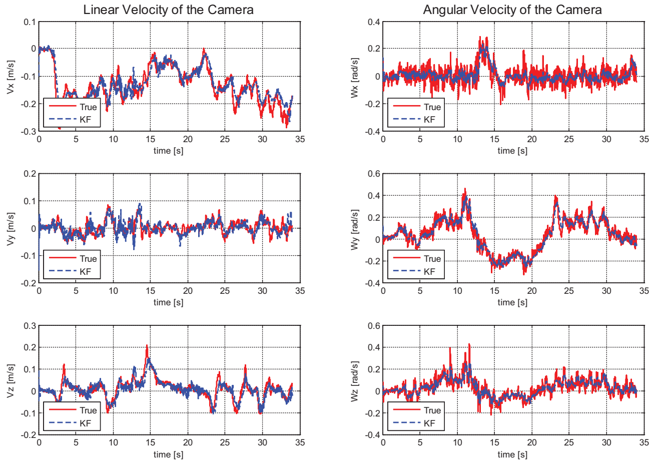

Fig. 6. **The velocity estimation result of experiment: fr3/struc&tex.** The estimated and truth velocity are plotted. The red lines denoted as 'True' mean the ground truth and the blue lines denoted 'KF' mean the estimated spatial velocity after passing Kalman filter.

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Ministry of Science, ICT & Future Planning (MSIP) (No. 2009-0083495) and (No. 2014034854).

References

[1] H. B. Christophersen, R. W. Pickell, J. C. Neidhoefer, A. A. Koller, S. K. Kannan, and E. N. Johnson, "A compact guidance, navigation, and control system for unmanned aerial vehicles," *Journal of aerospace computing, information, and communication*, vol. 3, no. 5, pp. 187–213, 2006.

[2] P. Martin and E. Salaun, “The true role of accelerometer feedback in quadrotor control,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 1623–1629.

[3] R. Mahony, V. Kumar, and P. Corke, “Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor,” *Robotics & Automation Magazine, IEEE*, vol. 19, no. 3, pp. 20–32, 2012.

[4] R. Leishman, J. Macdonald, R. Beard, and T. McLain, “Quadrotors and accelerometers: State estimation with an improved dynamic model,” *Control Systems, IEEE*, vol. 34, no. 1, pp. 28–41, 2014.

[5] P.-J. Bristeau, P. Martin, E. Salaün, N. Petit *et al.*, “The role of propeller aerodynamics in the model of a quadrotor uav,” in *European control conference*, vol. 2009, 2009.

[6] V. Grabe, H. Bulthoff, and P. R. Giordano, “On-board velocity estimation and closed-loop control of a quadrotor uav based on optical flow,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 491–497.

[7] M. Achtelik, S. Weiss, and R. Siegwart, “Onboard imu and monocular vision based control for mavs in unknown in-and outdoor environments,” in *Robotics and automation (ICRA), 2011 IEEE international conference on*. IEEE, 2011, pp. 3056–3063.

[8] F. Caballero, L. Merino, J. Ferruz, and A. Ollero, “Vision-based odometry and slam for medium and high altitude flying uavs,” in *Unmanned Aircraft Systems*. Springer, 2009, pp. 137–161.

[9] D. Honegger, P. Greisen, L. Meier, P. Tanskanen, and M. Pollefeys, “Real-time velocity estimation based on optical flow and disparity matching,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 5177–5182.

[10] K. Konolige, M. Agrawal, R. C. Bolles, C. Cowan, M. Fischler, and B. Gerkey, “Outdoor mapping and navigation using stereo vision,” in *Experimental Robotics*. Springer, 2008, pp. 179–190.

[11] D. Li, Q. Li, N. Cheng, J. Song, Q. Wu, and L. Tang, “Invariant observer design of a rgb-d aided inertial system for mav in gps-denied environments,” in *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*. IEEE, 2013, pp. 2593–2599.

[12] L. R. G. Carrillo, A. E. D. López, R. Lozano, and C. Pégard, “Combining stereo vision and inertial navigation system for a quad-rotor uav,” *Journal of Intelligent & Robotic Systems*, vol. 65, no. 1-4, pp. 373–387, 2012.

[13] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, “Visual odometry and mapping for autonomous flight using an rgb-d camera,” in *International Symposium on Robotics Research (ISRR)*, 2011, pp. 1–16.

[14] A. Bachrach, S. Prentice, R. He, P. Henry, A. S. Huang, M. Krainin, D. Maturana, D. Fox, and N. Roy, “Estimation, planning, and mapping for autonomous flight using an rgb-d camera in gps-denied environments,” *The International Journal of Robotics Research*, vol. 31, no. 11, pp. 1320–1343, 2012.

[15] B. K. Horn, “Closed-form solution of absolute orientation using unit quaternions,” *JOSA A*, vol. 4, no. 4, pp. 629–642, 1987.

[16] E. Malis, “Improving vision-based control using efficient second-order minimization techniques,” in *Robotics and Automation, 2004. Proceedings. ICRA’04. 2004 IEEE International Conference on*, vol. 2. IEEE, 2004, pp. 1843–1848.

[17] S. A. Scherer and A. Zell, “Efficient onbard rgbd-slam for autonomous mavs,” in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 1062–1068.

[18] H. Lim, H. Lee, and H. J. Kim, “Onboard flight control of a micro quadrotor using single strapdown optical flow sensor,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 495–500.

[19] J. Shi and C. Tomasi, “Good features to track,” in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR’94., 1994 IEEE Computer Society Conference on*. IEEE, 1994, pp. 593–600.

[20] J.-Y. Bouguet, “Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm,” *Intel Corporation*, vol. 2, p. 3, 2001.

[21] E. Dubrofsky, “Homography estimation,” Ph.D. dissertation, UNIVERSITY OF BRITISH COLUMBIA, 2009.

[22] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[23] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[24] F. Chaumette and S. Hutchinson, “Visual servo control. i. basic approaches,” *Robotics & Automation Magazine, IEEE*, vol. 13, no. 4, pp. 82–90, 2006.

[25] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[26] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A benchmark for the evaluation of rgb-d slam systems,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 573–580.

[27] A. Geiger, J. Ziegler, and C. Stiller, “Stereoscan: Dense 3d reconstruction in real-time,” in *Intelligent Vehicles Symposium (IV), 2011 IEEE*. IEEE, 2011, pp. 963–968.