Research Article

Design of Jitter Compensation Algorithm for Robot Vision Based on Optical Flow and Kalman Filter

B. R. Wang, 1,2 Y. L. Jin, 1 D. L. Shao, 1 and Y. Xu 1

1 College of Mechanical and Electrical Engineering, China Jiliang University, Hangzhou, Zhejiang 310018, China
2 Department of Control Science and Engineering, Zhejiang University, Hangzhou, Zhejiang 310007, China

Correspondence should be addressed to B. R. Wang; wangbinrui@163.com

Received 1 August 2013; Accepted 14 November 2013; Published 29 January 2014

I n t r o d u c t i o n

Computer vision is the most important sensor of intelligent moving robots. In real environment, the surface evenness always causes the camera jitters to affect the precision of operation. Electronic vision stabilization has been widely used in the autonomous robot vision [1–4], which includes motion estimation, smoothing and compensation processes [5]. The method of BMA (Blocks Matching Algorithm) was provided with the exhaustive search method in [6]. BMA can get high precision with large amount of calculation. However, its real time capability is bad and just fit for the simple motion vision [7]. Reference [8] used the circular block matching to improve the estimation for the rotational motion vision. Reference [9] researched a matching algorithm based on splitting and merging of block and representative points to improve the calculation speed. Projection algorithm (PA) was put forward for the vision motion estimation to gain the displacement vector based on the grayscale change of images. However, PA is valid for these images with the obvious grayscale change. Reference [10] researched the feature-tracking algorithm (FTA). Reference [11] used the optical flow constraints equations to solve the motion parameters of images. Optical flow is the 2D instantaneous velocity field of moving points in focal plane array. The aim of motion error compensation is to reconstruct images based on the smoothed parameters. The mean filter was used for smoothing motion vector in [12]. Reference [13] put motion parameters into the finite impulse response (FIR) filter and filtered them with product of input sequences and interruptive function. To distinguish the independent movement from the jitters, Kalman filter was used in image stabilization [14]. Kalman algorithm can predict the image motion and adjust observation data based on error covariance. Moreover, authors had compared the translation jitters to rotation jitters, FIR filter to Kalman filter, and relative parameters filter to absolute parameters filter, and the advantages and disadvantages of various algorithms were given out clearly [15]. From the above literatures, we found that the analysis of images motion equations solving error is less, especially in view of feature point pairs number and matching error, and jitters compensation test is absent on the autonomous robot moving on bricks road in outroom.

The rest of this paper is organized as follows. Firstly, an image kinematics model is established and the feature points
detecting and matching methods are designed based on the gradient matrix Eigen-value and the optical flow, and image motion parameters solving method is given in Section 2. A jitters compensation process based on filter is described in Section 3. Then the condition number of the equation coefficient is used to analyse the parameters error. A compensation software is developed and experiments are implemented based on the platform of two autonomous robots moving in outdoor in Section 4, and curves are compared between before and after Kalman filter. Finally, results are given in Section 5.

2. Kinematics Modelling and Solving

2.1. Kinematics Modelling of Images. Coordination of the pixel P is defined as \( P(u, v, t) \) at time \( t \) in the image coordinate system. \( (u_1, v_1) \) denotes the coordination of \( P \) in the given frame and \( (u_2, v_2) \) in the adjacent frame. According to the imaging producing principle, \( P \) coordination moving equations are described as

\[
\begin{align*}
    u_2 &= \frac{z_1}{z_1 + \Delta z} (u_1 \cos \theta + v_1 \cos \theta) \\
    &+ \frac{f}{z_1 + \Delta z} \left( \Delta x \cos \theta - \Delta y \sin \theta \right), \\
    v_2 &= \frac{z_1}{z_1 + \Delta z} (-u_1 \sin \theta + v_1 \cos \theta) \\
    &+ \frac{f}{z_1 + \Delta z} \left( \Delta x \sin \theta - \Delta y \cos \theta \right),
\end{align*}
\]

where \( f \) is the focal length of camera and \( z_1 \) is the position of \( P \) in direction of optical axis of camera coordinate system. \( \Delta x \), \( \Delta y \), and \( \Delta z \) are the amount of coordination increment, and \( \theta \) is the rotation amount. According to (1), images motion is relative to 6 parameters. So the image motion kinematics can be established as

\[
\begin{bmatrix}
    u_2 \\
    v_2
\end{bmatrix} = \begin{bmatrix}
    u_1 \\
    v_1
\end{bmatrix} \begin{bmatrix}
    a_0 & a_1 & a_2 \\
    a_3 & a_4 & a_5
\end{bmatrix} = A \begin{bmatrix}
    u_1 \\
    v_1
\end{bmatrix},
\]

where \( a_0, a_1, a_2, a_3, a_4 \) and \( a_5 \) indicate the scale and rotation amount, and \( a_2 \) and \( a_5 \) the translation amount.

2.2. Feature Point Detecting. The feature window is defined as a \( j \times j \) (\( j \) is odd number) square, and the center point of this feature window is, namely, the feature point. Gradient matrix of the feature window is

\[
G = \begin{bmatrix}
    \sum_{w} g_x^2 & \sum_{w} g_x g_y \\
    \sum_{w} g_x g_y & \sum_{w} g_y^2
\end{bmatrix},
\]

where \( w \) represents the feature window scope and \( g_x \) and \( g_y \) represent the gradients in horizontal and vertical direction, respectively, which can be got by numerical difference. Using \( k \times k \) (\( k < j \)) subwindow to scan the feature window along horizontal and vertical direction, we can get \( (j - k)^2 \) scanning windows. Each gradient matrix of the scanning window has two real Eigen-values, and the lesser Eigen-value is denoted by \( EV_{\text{min}} \). The maximum value of the lesser Eigen-value of all scanning windows can be expressed as

\[
EV = \max \{ EV_{1\text{min}}, EV_{2\text{min}}, \ldots, EV_{i\text{min}} \}
\]

\[
i = 1, 2, \ldots, (j - k)^2.
\]

This paper adopts \( EV \) to describe the center point characteristics quantity. If \( EV \) is larger than the given threshold, this point will be selected as the useful feature point.

2.3. Feature Points Matching Based on Optical Flow. When robot is moving in the continuous surface, the adjacent points have homothetic motions, constant brightness, and a tiny small motion in continuous time.

The frames constraint equation can be transformed using Taylor formula \[11\]; the following equation can be got

\[
I_x u' + I_y v' + I_z = 0,
\]

where \( I_x = \partial I / \partial x \) and \( I_y = \partial I / \partial y \) are the derivatives in horizontal and vertical direction, respectively, \( I_z = \partial I / \partial t \) is the derivative of time, and \( u' \) and \( v' \) are the coordinate difference of the feature point, namely, the gradient of optical flow.

We establish (6) of all points in \( 5 \times 5 \) square and use the least square method to solve the \( u' \) and \( v' \). Based on the \( u' \) and \( v' \), the feature point’s corresponding pixels coordinate in adjacent frame can be got.

2.4. Kinematics Parameters Solving. The aim of kinematics parameters solving is to get the image motion parameters \( a_0, a_1, a_2, a_3, a_4 \), and \( a_5 \). According to (3), there are 6 parameters needing to be solved, so at least 3 pairs of matching points are necessary. To ensure precise solving and stability, the number of pairs, denoted by \( K \), is always more than 3. The solving equation of the kinematics parameters is established as

\[
\begin{bmatrix}
    a_0 \\
    a_1 \\
    a_2 \\
    a_3 \\
    a_4 \\
    a_5
\end{bmatrix} = \begin{bmatrix}
    u_1 & v_1 & 1 & 0 & 0 & 0 \\
    0 & 0 & 0 & u_1 & v_1 & 1 \\
    0 & 0 & 0 & u_2 & v_2 & 1 \\
    \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
    u_K & v_K & 1 & 0 & 0 & 0
\end{bmatrix}^{-1} \begin{bmatrix}
    u'_1 \\
    v'_1 \\
    u'_2 \\
    v'_2 \\
    \vdots \\
    u'_K \\
    v'_K
\end{bmatrix} = B \begin{bmatrix}
    u'_1 \\
    v'_1 \\
    u'_2 \\
    v'_2 \\
    \vdots \\
    u'_K \\
    v'_K
\end{bmatrix},
\]

where \( (u_1, v_1) \) and \( (u'_1, v'_1) \) are the matching point coordinate position of the feature point, respectively.

When \( K > 3 \), (7) is an overdetermined equation and can also be solved using the least square method.

3. Jitter Compensation and Errors Analysis

Using \( q_{m-n} \) and \( q_n \) to denote the corresponding pixels coordinates in the frames, of \( I_{m-n} \) and \( I_n \) separated by \( m \) frames respectively, we can transform (3) as the affine transformation kinematics model through the recursive method

\[
q_n = F_{n}^{r-m} q_{m-n} + D_{n}^{r-m},
\]

\[\text{The Scientific World Journal}\]
the 3rd feature matching points, such as

\[ m = 1, 2, \ldots, n - 1 \]

\[ B = \prod_{n-m}^{m-1} F_{n-m}^{n-k} D_{n-m}^{n-k} \]

\[ \begin{bmatrix} a_0 & a_1 & a_2 & a_3 & a_4 \end{bmatrix}^T \]

Similarly, 30 pairs of the feature point, including the above 3 pairs, were used for solving (7), and also designedly add 1 pixel error to the 3rd point. Kinematics parameters solving results are shown in Table 1. The bold numbers in Table 1 refer to the parameters with errors. Table 1 shows that solving with 30 pairs matching points is robust to the matching error, and the individual point matching error would not produce a large change on results. The reason is that the condition number of \( B \) with 30 pairs is much lesser than with 3 pairs. So the solving method just using 3 pairs matching points is more sensitive to the matching error.

In view of (3), using parameters got by (7) to compute the condition number of \( A \), solve the corresponding point of the compensation processing. Results are shown in Table 2. The bold numbers in Table 2 refer to the parameters with errors. Table 2 also shows that solving (3) with parameters got by 30 pairs matching points is robust and can get more precise results.

### 4. Vision Stabilization Software Testing

#### 4.1. Test Setup

Experiments were conducted on two autonomous moving robots. Robot.1 (large) is the Voyager-IIA autonomous robot made in China. Robot.2 (small) is the X80-H robot made in Canada. Robot.1 has many sensors such as vision camera, ultrasonic, infrared ray, and gyroscope. Robot.2 is equipped with wireless communication equipment. The physical experiment scene is shown in Figure 1. In experiment processing of Robot.1 following Robot.2, the moving speed of Robot.1 is set to 0.11 m/s, while of Robot.2 to 0.07 m/s and linear forward.

The two autonomous moving robots are controlled by the personal computer (PC) through wireless network. Autonomous navigation software on PC controls motion of the autonomous mobile robot, such as move forward, turn back, speed up, and slow down. The CMOS camera is fixed on Robot.1 and connected with PC by USB line, and it transfers the real-time images to PC.

#### 4.2. Jitter Compensation Software Design

Software was developed using the Visual C++6.0 programming language on the Windows XP operating system. And the central processing unit (CPU) is an Intel Core2Duo 2 GHz system with 1 GB.

| Table 1: Kinematics parameters errors analysis of 3 to 30 pairs feature points. |
|---------------------|-----------------|-----------------|-----------------|
| B condition number  | Kinematics parameters | Max. errors | Ideal kinematics parameters |
| 3 pairs | 1563.2 | (1, 0, 0, 0.0095, 1.0571, 0.9714) | 2.0286 | (1, 0, 0, 0, 1, 3) |
| 30 pairs | 347.5 | (1, 0, 0, 0.0054, 0.9981, 3.2713) | 0.2713 | |

The bold numbers refer to the parameters with errors.

| Table 2: Corresponding point errors analysis of 3 to 30 pairs points. |
|---------------------|-----------------|-----------------|-----------------|
| A condition number  | Corresponding point | Max. errors | Given point | Ideal corresponding point |
| 3 pairs | 13.5 | (102, 319.01) | 16.01 | (102, 300) |
| 30 pairs | 12.6 | (102, 303.25) | 0.25 | (102, 303) |

The bold numbers refer to the parameters with errors.
of RAM. The whole software is composed of three parts, the control software of Robot.1, the control software of Robot.2, and the jitters compensation software. Video was captured based on DirectShow. After the image stabilization, the smooth video is displayed on the screen of PC. The compensation software procedure is illustrated in Figure 2(a) and software visual interface in Figure 2(b).

The video sampling frequency in the mobile robot moving is 20 Hz; namely, the time interval of the adjacent frame is 50 ms. All jitters compensation time was tested through GetTickCount() and cvGetTickFrequency() functions provided by LIB files. And test result is about 24 ms, greatly less than 50 ms. So the proposed jitters compensation algorithm is real-time.

4.3. Subwindow Feature Point Detecting Experiment. The feature point detecting algorithm based on the gradient matrix Eigen-value always gets the feature points collected on the same objects. In order to uniformly distribute the feature points and accelerate the detecting speed, we divided whole image into many nonoverlapping domains with \( s \times s \) square size. Then we scanned \( s \times s \) subwindow to get feature points. Figure 3 shows the feature points detecting of one frame in the video sequence based on the conventional feature extraction and the improved feature extraction.

Feature points may concentrate on some objects in Figure 3(a), such as Robot.2 and the tree in background, which will easily make wrong matching and is disadvantageous to the parameters solving. Figure 3(b) shows that scan in subwindow makes the feature points equally distributed in the whole image. And the dispersed feature points are beneficial for the kinematics parameters solving using the least square method.

4.4. The Parameters Smooth Results Using Kalman Filter. Two robots moved linearly forward, respectively, apart by about 1.5 m. There is the same size blocks paved on robot moving road. The length and width of blocks are 19 cm and 9.4 cm, respectively, and slot between blocks is of 0.7 cm width and depth 0.3 cm. Robot.2 is forward, while Robot.1 is behind and its motion velocity is more than that of Robot.2. So Robot.1 is continuously getting closer to Robot.2. The test time of jitters compensation is 16 s.
Figure 4: The filter results of matrix $A$ parameters.
Table 3: The comparisons of matrix $A$ parameters of before and after filter.

|      | $a_0$  | $a_1$  | $a_2$  | $a_3$  | $a_4$  | $a_5$  |
|------|--------|--------|--------|--------|--------|--------|
| Before | 0.001434 | 0.004646 | 0.003686 | 0.010688 | 1.525818 | 0.348722 |
| After  | 0.000711 | 0.000649 | 0.000874 | 0.001435 | 0.348722 | 0.267184 |

Mean square errors (MSE) comparisons between before and after filter are shown in Table 3.

4.5. The Effect of Video Stabilization. The series frames of before and after the images stabilization are as shown in Figure 5.

Figure 5 shows that the curves after filter are smoother than before filter. The $a_5$ variation is more complicated than other parameters. The reason is that the video is obtained during the mobile robot is moving on the road paved with bricks, and the interval slots between bricks mainly produce vibration of robot wheel in vertical direction. Table 3 shows the mean square error after filter is less than before evidently.

5. Conclusions

Based on comparative analysis, the following can be got.

(1) The number of feature point pairs has great effect on the parameters solving precision, and this effect can be quantificationally analyzed by the condition number of matrices $A$ and $B$. The condition number of $B$ is far larger than $A$. Equation (7) is very sensitive to errors; kinematics parameters must be solved using the feature point pairs as many as possible to reduce the solving errors.

(2) Subwindow feature point detecting can avoid the feature points gathering on some objects.

(3) The visual jitters compensation algorithm based on optical flow and Kalman filter, developed based on PC, USB camera, Microsoft Windows operating system, and VC++, meets the requirements of precision and real-time demand of robot vision.

But the proposed method cannot compensate the migration jitters caused during the exposure time of the camera. Further study will focus on how to make the parameters of Kalman filter adaptively change with the different jitters amplitude and frequency.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.
Acknowledgments

This research was funded by a Grant (no. LQ13E050004) from the Natural Science Foundation of Zhejiang province and a Grant (no. 201210076) from the Research Project of General Administration of Quality Supervision, Inspection and Quarantine of China.

References

[1] T. Proscevicius, A. Bukis, V. Raudonis, and M. Eidukeviciute, "Hierarchical control approach for autonomous mobile robots," *Elektronika ir Elektrotechnika*, no. 4, pp. 101–104, 2011.

[2] L. Lingqiao, F. Zhizhong, X. Jingjing, and Q. Wei, "Edge mapping: a new motion estimation method for video stabilization," in *Proceedings of the International Symposium on Computer Science and Computational Technology (ICCSCT ’08)*, pp. 440–444, Shanghai, China, December 2008.

[3] S. Battiato, G. Puglisi, and A. R. Bruna, "A robust video stabilization system by adaptive motion vectors filtering," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME ’08)*, pp. 373–376, Hannover, Germany, June 2008.

[4] G. Corsini, M. Diani, and A. Masini, "Video sequence stabilization for real-time remote sensing applications," in *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS ’06)*, pp. 988–991, Denver, Colo, USA, August 2006.

[5] T. Marius, A. Sakari, and V. Markku, "Method of motion estimation for image stabilization," *Acoustics, Speech and Signal Processing*, vol. 2, no. 2, pp. 277–280, 2006.

[6] F. Vella, A. Castorina, M. Mancuso, and G. Messina, "Digital image stabilization by adaptive block motion vectors filtering," *IEEE Transactions on Consumer Electronics*, vol. 48, no. 3, pp. 796–801, 2002.

[7] S. Niauronis, V. Laurutis, and R. Zemblys, "Eye-hand coordination during self-moved target guiding along labyrinth path," *Elektronika ir Elektrotechnika*, no. 10, pp. 71–74, 2011.

[8] L. D. Xu, F. W. Fu, and X. G. Lin, "Circular block matching based video stabilization," in *Proceedings of the Visual Communications and Image Processing*, pp. 1307–1314, July 2005.

[9] J. L. Shi, W. J. Zhang, and S. Y. Yu, "Motion estimation based on block-splitting-and-merging and its criteria," *Journal of Shanghai Jiaotong University*, vol. 32, no. 4, pp. 49–53, 1998.

[10] J. C. Huang and W. S. Hsieh, "Automatic feature-based global motion estimation in video sequences," *IEEE Transactions on Consumer Electronics*, vol. 50, no. 3, pp. 911–915, 2004.

[11] B. Feng, C. H. Zhao, T. Yang, H. C. Zhang, and Y. M. Chen, "Real-time human action recognition based on optical-flow feature and sequence alignment," *Application Research of Computers*, vol. 24, no. 3, pp. 194–198, 2007.

[12] S. I. Ko, S. H. Lee, and S. W. Jeon, "Digital image stabilizing algorithm based on bit-plane matching," *IEEE Transactions on Consumer Electronics*, vol. 44, no. 3, pp. 32–39, 1998.

[13] W. H. Zhao, T. X. Yao, X. Q. Ye, and W. K. Gu, "RANSAC algorithm in video stabilization," *Journal of Circuits and Systems*, vol. 10, no. 4, pp. 91–94, 2005.

[14] K. K. Lee, K. H. Wong, M. M. Y. Chang, Y. K. Yu, and M. K. Leung, "Extended Kalman filtering approach to stereo video stabilization," in *Proceedings of the 19th International Conference on Pattern Recognition (ICPR ’08)*, pp. 1–4, Tampa, Fla, USA, December 2008.