Proceedings of the 2006 IEEE International Conference on Robotics and Automation
Orlando, Florida - May 2006

Simultaneous Pose and Velocity Measurement by Vision for High-Speed Robots

Omar Ait-Aider, Nicolas Andreff, Philippe Martinet and Jean-Marc Lavest
LASMEA - CNRS - Université Blaise Pascal/IFMA, 63175 Aubière, France
Email: {firstname.lastname}@lasmea.univ-bpclermont.fr
http://www.lasmea.univ-bpclermont.fr/Control

Abstract— **This paper proposes an original and novel vision sensing method to be used in vision-based dynamic identification of parallel robots. Indeed, it is shown that in the latter problem one requires to estimate (to the least) or measure (to the best) the end-effector pose and its time derivatives. The sensor we propose, based on a clever modelling of CMOS rolling shutter camera, measures simultaneously the end-effector pose of a calibrated visual pattern and its Cartesian velocity using a single view. Although motivated by parallel robot identification, this low cost sensor does not make any assumption on the kinematics of the robot and can thus be used for other applications. Experimental results with real data confirm the relevance of the approach and show the sensor good practical measurement accuracy.**

I. Introduction

This work spins off from parallel robot dynamic control. Indeed, it has been reported that high-speed vision could be used for dynamic control of serial robots [1]. Hence, we initially wanted to study whether vision-based control can also be applied to parallel manipulators. To do so, one first requires to identify the parameters of the dynamical model used for control [2]. Since vision has proven [3] an accurate tool for identifying the kinematic parameters of parallel robots, it may also be useful for dynamic identification.

Indeed, the case of parallel robots is in some sense dual to the case of serial robots. Namely, while serial robot end-effector pose is analytically defined from the joint values and the forward kinematic model, parallel robots usually have instead a closed-form non-linear expression for the inverse kinematic problem. Thus, to estimate a parallel robot end-effector pose, one has to solve for a non-linear optimization problem (known as the forward kinematic problem) which may have several solutions [4], [5]. For instance, for the referent case of the Gough-Stewart platform [6], one may have up to 40 real solutions [7]. An alternative is to measure the pose by exteroceptive means, such as vision (which solves a non-linear minimisation problem [8] which is simpler than the forward kinematic problem) or laser tracker [9] (which is noticeably more expensive than vision solutions). Similarly, one can notice that the dynamical model of parallel robots generally depends on the full end-effector pose (position and orientation) and its time derivatives [10], [11], [12]. Consequently, vision seems perfectly relevant to dynamic identification, provided that the vision system can reach a sufficient frame rate (around 1 kHz). Vision-based dynamic identification may thus reach a higher accuracy than the classical proprioceptive sensor-based one [13], [14], [15].

However, using vision sensors in this framework, requires the ability to capture clear images of objects undergoing high velocity motion without any distortion, blur nor smear. To achieve this task, there is a need to image sensors which allow very short exposure time of all the matrix pixels simultaneously. This functionality requires a particular electronic design that is not included in all camera devices. Full Frame CCD sensors, without storage memory areas, require mechanical obturator or stroboscopic light source, introducing more complexity in the vision system. Frame Transfer CCD sensors may not reach the desired frame rate or may be costly because of additional sillicium in storage areas [16]. Standard CMOS Rolling Shutter sensors are considered as low cost and low power sensors. They are becoming more frequently used in cameras. They enable adequate exposure time without reducing frame rate thanks to overlapping exposure and readout. Their drawback is that they distort images of moving objects because the pixels are not all exposed simultaneously but row by row with a time delay defined by the sensor technology. This distortion may represents a major obstacle in tasks such as localization, reconstruction or default detection (the system may see an ellipse where in fact there is a circular hole). Therefore, CMOS Rolling Shutter cameras could offer a good compromise between cost and frame rate performances if the problem of deformations is taken into account.

Coming back to frame rate, the current affordable technology allows to reach very high frame rates by reducing the sensor resolution (about 250×250 pixels). The latter resolution reduces a lot the field of view of the camera for a given accuracy of the end-effector pose estimation [8], [17]. This is very restrictive since the workspace available for identification, which should be as large as possible to provide good excitation and high result accuracy [18], [19], is consequently much reduced. If one tries to enlarge the field of view, the frame rate falls down and the rolling shutter phenomenon becomes more perceptible. Faced with this huge drawback, we turned it into an advantage since this mere effect acts as an analog time derivator. With adequate modelling, we can thus propose a method where both the full end-effector pose and the full Cartesian end-effector velocity can be estimated simultaneously with a single image. Consequently, this not only suppresses the need for numerically inverting the inverse

0-7803-9505-0/06/$20.00 ©2006 IEEE

kinematic model to estimate the end-effector pose from the joint values, but also suppresses (or makes it more robust) the numerical time derivation of the end-effector pose in the dynamical model.

One may argue that doing so, we reduced the frame rate and thus impaired the dynamic identification which uses high-frequency signals. Nevertheless, let us recall that in these signals are used mainly to be filtered and sub-sampled (down to a 100Hz) in order to make the identification process robust to high-frequency noise. If we can reach a 100Hz frame rate with 1 Mpixel resolution and rolling shutter, the measured end-effector poses and velocities should have an equivalent signal-to-noise ratio.

This work, is related to our previous one presented in [20], which focused on the development of a method which maintains accuracy in pose recovery and structure from motion algorithms without sacrificing low cost and power characteristics of the sensor. This was achieved by integrating, in the perspective projection model, kinematic and technological parameters which are both causes of image deformations. In a parallel work by Meingast [21] (published after the submission of this paper), the projection in rolling shutter cameras is modelled in case of fronto-parallel motion obtaining equations which are similar to those of Crossed-Slits cameras [22]. To our knowledge, there is no work in the vision community literature on modelling effects of rolling shutter in pose recovery algorithms nor on computing velocity parameters using a single view. All pose recovery methods ([23], [24], [8], [25], [26]) make the assumption that all image sensor pixels are exposed simultaneously. The work done by Wilburn et al. [27] concerned the correction of image deformation by constructing a single image using several images from a dense camera array. Using the knowledge of the time delay due to rolling shutter and the chronograms of release of the cameras, one complete image is constructed by combining lines exposed at the same instant in each image from the different cameras.

The main contributions in this paper are first, the improvement of the perspective projection model of rolling shutter cameras presented in [20] by removing the assumption of small motion during image acquisition, and then the use of this new model to develop a simultaneous pose and velocity recovery algorithm using a single view. This new model is more accurate when very fast moving objects are observed. The obtained algorithm, not only enables accurate pose recovery, but also provides with the instantaneous angular and translational velocity of observed objects. Rolling shutter effects which are considered as drawbacks are transformed here into an advantage ! This approach may be considered as an alternative to methods which uses image sequences to estimate the kinematic between views since it reduces the amount of data and the computational cost (one image is processed rather than several ones).

Section II of this paper describes the process of image acquisition using a CMOS Rolling Shutter imager. In section III, a general geometric model for the perspective projection of 3D point on a solid moving object is presented. Image coordinates of the point projections are expressed with respect to object pose and velocity parameters and to the time delay of CMOS sensor image row scanning. Section IV deals with the problem of computing pose and velocity parameters of a moving object, imaged by a CMOS Rolling Shutter camera, using point correspondences. The proposed algorithm generalizes the bundle adjustment method to the case of moving points. It is based on non-linear least-square optimization of an error function defined in image metric and expressed with respect to both pose and velocity parameters (rather than to only pose parameters in classical approaches). Finally, experiments with real data are presented and analyzed in section V.

II. What is rolling shutter ?

In digital cameras, an image is captured by converting the light from an object into an electronic signal at the photosensitive area (photodiode) of a solid state CCD or CMOS image sensor. The amount of signal generated by the image sensor depends on the amount of light that falls on the imager, in terms of both intensity and duration. Therefore, an on-chip electronic shutter is required to control exposure. The pixels are allowed to accumulate charge during the integration time. With global shutter image sensors, the entire imager is reset before integration. The accumulated charge in each pixel is simultaneously transferred to storage area. Since all the pixels are reset at the same time and integrate over the same interval there is no motion artifacts in the resulting image. With a CMOS image sensor with rolling shutter, the rows of pixels in the image are reset in sequence starting at the top and proceeding row by row to the bottom. The readout process proceeds in exactly the same fashion and the same speed with a time delay after the reset. This time delay between a reset of a row and its reading out is the exposure time. Each line in the image has the same amount of integration, however the start and end time of integration is shifted in time as the image is scanned (rolled) out of the sensor array as shown in Fig.1. In this case, if the object is moving during the integration time, some artifacts may appear. The faster the object moves the larger is the distortion. An example of distortion due to rolling shutter on the appearance of a rotating ventilator is shown in Fig.2. The benefit of rolling shutter mode is that exposure and readout are overlapping, enabling full frame exposures without reducing frame rate.

III. Projecting a point with rolling shutter camera

Let us consider a classical camera with a pinhole projection model defined by its intrinsic parameter matrix [26]

$$\mathbf{K} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

Let $\mathbf{P} = [X, Y, Z]^{\mathrm{T}}$ be a 3D point defined in the object frame. Let $\mathbf{R}$ and $\mathbf{T}$ be the rotation matrix and the translation

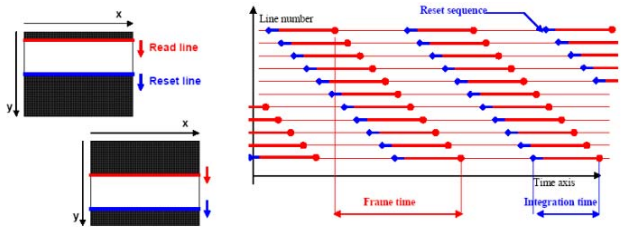

Fig. 1. Reset and reading chronograms in rolling shutter sensor (SILICON IMAGING documentation).

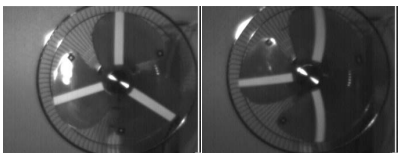

Fig. 2. An example of distortion of a rotating ventilator observed with a Rolling Shutter camera: static object (right image) and moving object (left image).

vector between the object frame and the camera frame. Let $\mathbf{m} = [u, v]^{\mathrm{T}}$ be the perspective projection of $\mathbf{P}$ on the image. Noting $\tilde{\mathbf{m}} = \left[\mathbf{m}^{\mathrm{T}}, 1\right]^{\mathrm{T}}$ and $\tilde{\mathbf{P}} = \left[\mathbf{P}^{\mathrm{T}}, 1\right]^{\mathrm{T}}$, the relationship between $\mathbf{P}$ and $\mathbf{m}$ is:

$$s\tilde{\mathbf{m}} = \mathbf{K}\left[\mathbf{R} \quad \mathbf{T}\right]\tilde{\mathbf{P}} \tag{1}$$

where s is an arbitrary scale factor. Note that the lens distortion parameters which do not appear here are obtained by calibrating [17] and are taken into account by correcting image data before using them in the algorithm.

Assume now that an object of a known geometry modelled by a set of n points $\mathbf{P_i} = [X_i, Y_i, Z_i]^{\mathrm{T}}$, undergoing a motion with instantaneous angular velocity Ω around an instantaneous axis of unit vector $\mathbf{a} = [a_x, a_y, a_z]^{\mathrm{T}}$, and instantaneous linear velocity $\mathbf{V} = [V_x, V_y, V_z]^{\mathrm{T}}$, is snapped with a rolling shutter camera at an instant t_0. In fact, t_0 corresponds to the instant when the top line of the sensor is exposed to light. Thus, the light from the point $\mathbf{P_i}$ will be collected with a delay τ_i proportional to the image line number on which $\mathbf{P_i}$ is projected. As illustrated in figure 3, τ_i is the time delay necessary to expose all the lines above the line which collects the light from $\mathbf{P_i}$. Therefore, to obtain the projection $\mathbf{m}_i = [u_i, v_i]^{\mathrm{T}}$ of $\mathbf{P_i}$, the pose parameters of the object must be corrected in equation 1 by integrating the motion during the time delay τ_i. Since all the lines have the same exposure and integration time, we have $\tau_i = \tau v_i$ where τ is the time delay between two successive image line exposure. Thus $\tau = \frac{fp}{v_{max}}$ where fp is the frame period and v_{max} is the image height. Assuming that τ_i is short enough to consider uniform motion during this interval, the object rotation during this interval is obtained thanks to the Rodrigues formula:

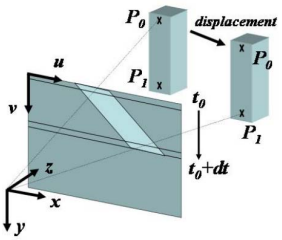

Fig. 3. Perspective projection of a moving 3D object: due to the time delay, points $\mathbf{P}_0$ and $\mathbf{P}_1$ are not projected from the same object pose

$$\delta\mathbf{R}_i = \mathbf{a}\mathbf{a}^T\left(1 - cos\left(\tau v_i \Omega\right)\right) + \mathbf{I}cos\left(\tau v_i \Omega\right) + \hat{\mathbf{a}}sin\left(\tau v_i \Omega\right)$$

where $\mathbf{I}$ is the 3×3 identity matrix and $\hat{\mathbf{a}}$ the antisymetric matrix of $\mathbf{a}$. The translation during the same interval is:

$$\delta\mathbf{T}_i = \tau v_i \mathbf{V}$$

Thus, equation 1 can be rewritten as follows:

$$s\tilde{\mathbf{m}}_i = \mathbf{K}\left[\mathbf{R}\delta\mathbf{R}_i \quad \mathbf{T} + \delta\mathbf{T}_i\right]\tilde{\mathbf{P}}_\mathbf{i} \tag{2}$$

where $\mathbf{R}$ and $\mathbf{T}$ represent now the instantaneous object pose at t_0. Equation 2 is the expression of the projection of a 3D point from a moving solid object using a rolling shutter camera with respect to object pose, object velocity and the parameter τ. One can note that it contains the unknown v_i in its two sides. This is due to the fact that coordinates of the projected point on the image depend on both the kinematics of the object and the imager sensor scanning velocity.

IV. Computing the Instantaneous Pose and Velocity of a Moving Object

In this section, we assume that a set of rigidly linked 3D points $\mathbf{P_i}$ on a moving object are matched with their respective projections $\mathbf{m_i}$ measured on an image taken with a rolling shutter camera. We want to use this list of 3D-2D correspondences to compute the instantaneous pose and velocity of the object at instant t_0. The scale factor of equation 2 can be removed as follows:

$$\begin{aligned} u_i &= \alpha_u \frac{\mathbf{R^{(1)}}_i\mathbf{P_i} + T^{(x)}}{\mathbf{R^{(3)}}_i\mathbf{P_i} + T^{(z)}} + u_0 = \xi_i^{(u)}\left(\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V}\right) \\ v_i &= \alpha_v \frac{\mathbf{R^{(2)}}_i\mathbf{P_i} + T^{(y)}}{\mathbf{R^{(3)}}_i\mathbf{P_i} + T^{(z)}} + v_0 = \xi_i^{(v)}\left(\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V}\right) \end{aligned} \tag{3}$$

where $\mathbf{R^{(j)}}$ and $T^{(x,y,z)}$ are respectively the j^{th} row of $\mathbf{R}_i = \delta\mathbf{R}_i\mathbf{R}$ and the components of $\mathbf{T}_i = \mathbf{T} + \delta\mathbf{T}_i$. Subsiding the right term from the left term and substituting u_i and v_i by image measurements, equation 3 can be seen as an error function with respect to pose and velocity (and possibly τ) parameters:

$$\begin{aligned} u_i - \xi_i^{(u)}\left(\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V}\right) &= \epsilon_i^{(u)} \\ v_i - \xi_i^{(v)}\left(\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V}\right) &= \epsilon_i^{(v)} \end{aligned}$$

We want to find $\mathbf{R}$, $\mathbf{T}$, Ω, $\mathbf{a}$ and $\mathbf{V}$ that minimize the following error function:

$$\epsilon = \sum_{i=1}^{n} \left[u_i - \xi_i^{(u)} \left(\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V} \right) \right]^2 + \left[v_i - \xi_i^{(v)} \left(\mathbf{R}, \mathbf{T}, \Omega, \mathbf{a}, \mathbf{V} \right) \right]^2 \quad (4)$$

This problem with 12 independent unknowns can be solved using a non-linear least square optimization if at least 6 correspondences are available. This can be seen as a bundle adjustment with a calibrated camera. Note that, in our algorithm, the rotation matrix $\mathbf{R}$ is expressed by a unit quaternion representation $q(\mathbf{R})$. Thus, an additional equation, which forces the norm of $q(\mathbf{R})$ to 1, is added. It is obvious that this non-linear algorithm requires an initial guess to converge towards an accurate solution.

V. Experiments

The aim of this experimental evaluation is first to illustrate our pose recovery algorithm accuracy in comparison with classical algorithms under the same acquisition conditions, and second, to show its performances as a velocity sensor. The algorithm was tested on real image data. A reference 3D object with white spots was used. Sequences of the moving object at high velocity were captured with the Silicon Imaging CMOS Rolling Shutter camera SI1280M-CL, calibrated using the method described in [17]. Acquisition was done with a 1280×1024 resolution and at a rate of 30 frames per second so that $\tau = 7.15 \times 10^{-5}$ s. Image point coordinates were accurately obtained by a sub-pixel accuracy computation of the white spot centers and corrected according to the lens distortion parameters. Correspondences with model points were established with a supervised method. The pose and velocity parameters were computed for each image using first our algorithm, and compared with results obtained using the classical pose recovery algorithm described in [17] where an initial guess is first computed by the algorithm of Dementhon [8] and then the pose parameters are accurately estimated using a bundle adjustment technique.

Figure 4 shows image samples from a sequence where the reference object was moved following a straight rail forcing its motion to be a pure translation. In the first and last images of the sequence the object was static. Pose parameters corresponding to these two views were computed accurately using the classical algorithm. The reference object trajectory was then assumed to be the 3D straight line relating the two extremities. Table I shows the RMS pixel re-projection error obtained using the pose computed with the classical algorithm and a classical projection model from the one hand-side, and the pose computed with our algorithm and the rolling shutter projection model from the other hand-side. Results show that errors obtained with static object views are similar. However, as the velocity increases, the error obtained with the classical algorithm becomes too important while the error obtained with our algorithm remains small.

Fig. 4. Image samples of pure translational motion.

TABLE I
RMS re-projection error (pixel).

Image number	Classical algorithm RMS-u	Classical algorithm RMS-v	Our algorithm RMS-u	Our algorithm RMS-v
1	0.14	0.12	0.15	0.13
2	1.71	1.99	0.10	0.09
3	3.95	4.18	0.11	0.09
4	7.09	7.31	0.09	0.07
5	5.56	6.73	0.13	0.12
6	1.87	3.02	0.18	0.11
7	0.25	0.12	0.25	0.17

Let us now analyze pose recovery results shown in figure 5. The left-hand side of this figure shows 3D translational pose parameters obtained by our algorithm and by the classical algorithm (respectively represented by square and *-symbols). Results show that the two algorithms give appreciably the same results with static object views (first and last measurements). When the velocity increases, a drift appears in the classical algorithm results while our algorithm remains accurate (the 3D straight line is accurately reconstructed by pose samples) as it is illustrated on Table II where are represented distances between computed poses with each algorithm and the reference trajectory. Table III presents computed rotational pose parameters. Results show the deviation of computed rotational pose parameters from the reference orientation. Since the motion was a pure translation, orientation is expected to remain constant. As one can see, a drift appears on classical algorithm results while our algorithm results show a very small deviation due only to noise on data.

Another result analysis concerns the velocity parameters. Figure 5 shows that the translational velocity vector is clearly parallel to translational axis (up to noise influence). Table IV represents magnitude of computed velocity vectors in com-

TABLE II
Distances from computed poses to reference trajectory (cm).

Image number	1	2	3	4	5	6	7
Classical algorithm	0.00	0.19	0.15	1.38	3.00	4.54	0.00
Our algorithm	0.18	0.24	0.26	0.22	0.32	0.11	0.07

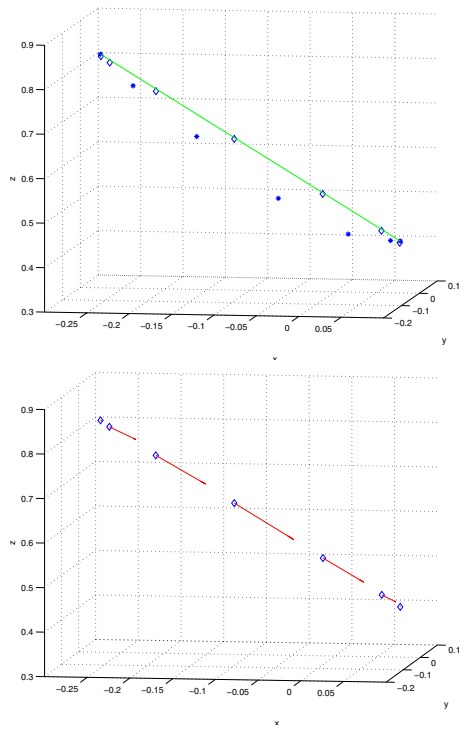

Fig. 5. Pose and velocity results: reconstructed trajectory (top image), translational velocity vectors (bottom image).

TABLE III

ANGULAR DEVIATION OF COMPUTED POSES FROM REFERENCE ORIENTATION (DEG.).

Image number	1	2	3	4	5	6	7
Dementhon's algo.	0.00	2.05	4.52	6.93	6.69	3.39	0.30
our algorithm	0.07	0.13	0.15	0.24	0.90	0.91	0.40

parison with measured values. These reference values were obtained by dividing the distance covered between each two successive images by the frame period. This gives estimates of the translational velocity magnitudes. Results show that the algorithm recovers correctly acceleration, deceleration and static phases. Table V represents computed rotational velocity parameters. As expected, the velocity parameter values are small and only due to noise.

In the second experiment, the algorithm was tested on coupled rotational and translational motions. The previously described reference object was mounted on a rotating mechanism. Its circular trajectory was first reconstructed from a set of static images. This reference circle belongs to a plan whose

TABLE IV

COMPUTED TRANSLATIONAL VELOCITY MAGNITUDE IN COMPARISON WITH MEASURED VELOCITY VALUES (M/S)

Image number	1	2	3	4	5	6	7
Measured values	0.00	1.22	2.02	2.32	1.55	0.49	0.00
Computed values	0.06	1.10	1.92	2.23	1.54	0.50	0.02

TABLE V

COMPUTED ROTATIONAL VELOCITIES (RAD/S).

Image number	1	2	3	4	5	6	7
our algorithm	0.04	0.07	0.05	0.01	0.15	0.11	0.12

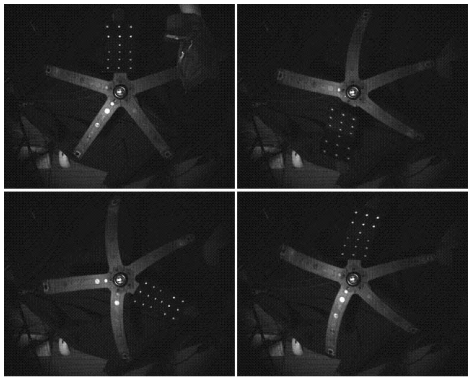

Fig. 6. Image samples of coupled rotational and translational motions.

measured normal vector is $\mathbf{N} = [0.05, 0.01, -0.98]^{\mathrm{T}}$. Thus, $\mathbf{N}$ represents the reference rotation axis. An image sequence of the moving object was then captured. Figure 6 shows samples of images taken during the rotation.

The left part of figure 7 represents the trajectory reconstructed with a classical algorithm (*-symbol) and with our algorithm (square symbol). As for the pure translation, results show that the circular trajectory was correctly reconstructed by the poses computed with our algorithm, while a drift is observed on the results of the classical algorithm as the object accelerates. The right part of the figure shows that translational velocity vectors were correctly oriented (tangent to the circle) and that the manifold of instantaneous rotation axis vectors was also correctly oriented. The mean value of the angles between the computed rotation axis and $\mathbf{N}$ is 0.50 degrees. Results in table VI shows a comparison of the computed rotational velocity magnitudes and the measured values.

VI. CONCLUSION AND PERSPECTIVES

An original method for computing pose and instantaneous velocity of rigid objects using a single view from a rolling shutter camera was presented. The perspective projection

TABLE VI

COMPUTED AND MEASURED ROTATIONAL VELOCITY MAGNITUDES (RAD/S)

Image number	1	2	3	4	5
Measured values	0.00	1.50	9.00	11.20	10.50
Computed values	0.06	1.20	8.55	10.38	10.32

Image number	6	7	8	9	10
Measured values	10.20	10.10	10.00	10.00	7.50
Computed values	10.30	9.80	9.90	9.73	8.01

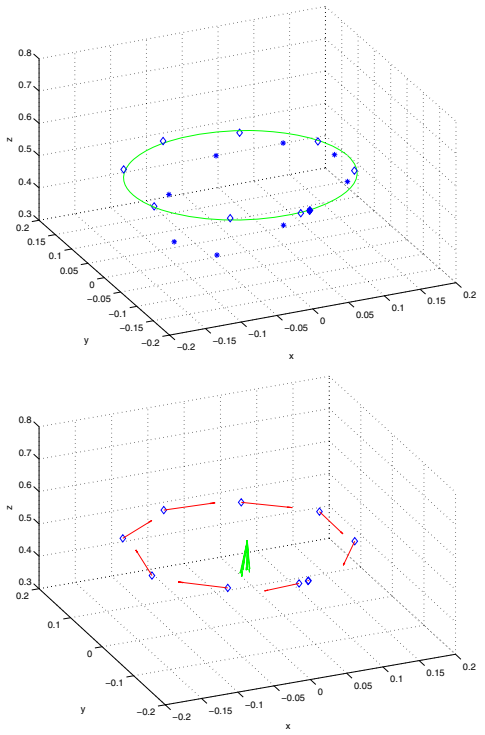

Fig. 7. Pose and velocity results for coupled rotational and translational motion: reconstructed trajectory (top image), rotational and translational velocities (bottom image).

equations of a moving 3D point was first established. Then, an error function equivalent to collinearity equations in camera calibration was defined and minimized numerically to obtain the object pose and velocity parameters. The approach was evaluated on real data showing its feasibility. The method is not less accurate than similar classical algorithms in case of static objects, and improves pose accuracy in case of fast moving objects. In addition, the method gives the instantaneous velocity parameters using a single view. This last property makes it an original tool for many fields of research. For example, it may avoid numerical derivation to estimate the velocity from several poses. Instantaneous velocity information may also be used as evolution models in motion tracking, as well as in visual servoing, to predict the state of moving patterns.

Our future works, will focus on showing the feasibility of using the presented algorithm for dynamic identification of parallel robots by developing an evaluation framework with a real robot. The other interesting focus will concern the development of linear solutions for pose and velocity computation. This will provide to the non-linear optimization an initial estimate of the solution.

REFERENCES

[1] J. Gangloff and M. de Mathelin. High-speed visual servoing of a 6 DOF manipulator using multivariable predictive control. *Advanced Robotics. Special issue: advanced 3D vision and its application to robotics*, 17(10):993–1021, December 2003.

[2] W. Khalil and E. Dombre. *Modeling, Identification and Control of Robots*. Taylor and Francis, 2002.

[3] P. Renaud, N. Andreff, J.-M. Lavest, and M. Dhome. Simplifying the kinematic calibration of parallel mechanisms using vision-based metrology. *IEEE Transactions on Robotics*, 2005.

[4] J-P. Merlet. An algorithm for the forward kinematics of general 6 d.o.f. parallel manipulators. Technical Report 1331, INRIA, November 1990.

[5] M. Husty. An algorithm for solving the direct kinematics of general Gough-Stewart platforms. *Mech. Mach. Theory*, 31(4):365–380, 1996.

[6] V.E. Gough and S.G. Whitehall. Universal tyre test machine. In *Proc. FISITA 9th Int. Technical Congress*, pages 117–137, May 1962.

[7] P. Dietmaier. The Stewart-Gough platform of general geometry can have 40 real postures. In J. Lenarčič and M. L. Husty, editors, *Advances in Robot Kinematics: Analysis and Control*, pages 1–10. Kluwer, 1998.

[8] D. Dementhon and L.S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1/2):123–141, June 1995.

[9] M. Vincze, J.P. Prenninger, and H. Gander. A laser tracking system to measure position and orientation of robot end-effectors under motion. *Int. J. Robotics Research*, 13(4):305–314, 1994.

[10] W. Khalil and S. S. Guégan. Inverse and direct dynamic modeling of gough-stewart robot. *IEEE Transactions on Robotics and Automation*, 20(4):754–762, August 2004.

[11] S. Guégan and W. Khalil. Dynamic modeling of the orthoglide. In *Proc. 8th International Symposium on Advances in Robot Kinematics (ARK'2002)*, pages 387–396, Caldes de Malavella, Spain, June 2002.

[12] O. Ibrahim, W. Khalil, and S. Guégan. Dynamic modeling of some parallel robots. In *Proc. 35th International Symposium on Robotics*, Villepinte, March 23-26 2004.

[13] P. Renaud, A. Vivas, N. Andreff, P. Poignet, P. Martinet, F. Pierrot, and O. Company. Kinematic and dynamic identification of parallel mechanisms. *Control Engineering Practice*, 2005.

[14] M. Gautier and Ph. Poignet. Extended kalman filtering and weighted least squares dynamic identification of robot. *Control Engineering Practice*, 9(12):1361–1372, 2001.

[15] M.M. Olsen and H.G. Petersen. A new method for estimating parameters of a dynamic robot model. *Trans. on Robotics and Automation*, 17(1):95–100, 2001.

[16] A. J. P. Theuwissen. *Solid-state Imaging with Chargecoupled Devices*. Kluwer Academic Publishers, 1995.

[17] J. M. Lavest, M. Viala, and M. Dhome. Do we really need an accurate calibration pattern to achieve a reliable camera calibration. In *Proceedings of ECCV98*, pages 158–174, Freiburg, Germany, June 1998.

[18] M. Gautier and W. Khalil. Exciting trajectories for the identification of base inertial parameters of robots. *Int. J. Robotics Research*, 11(4):362–375, 1992.

[19] J. Swevers, C. Ganseman, B.D. Tükel, J. De Schutter, and H. Van Brussel. Optimal robot excitation and identification. *Trans. on Robotics and Automation*, 13(5):730–740, 1997.

[20] O. Ait-Aider, N. Andreff, J. M. Lavest, and P. Martinet. Exploiting rolling shutter distortions for simultaneous object pose and velocity computation using a single view. In *Proc. IEEE International Conference on Computer Vision Systems*, New York, USA, January 2006.

[21] M. Meingast, C. Geyer, and S. Sastry. Geometric models of rolling-shutter cameras. In *Proc. of the 6th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras*, Beijing, China, October 2005.

[22] A. Zomet, D. Feldman, S. Peleg, and D. Weinshall. Mosaicing new views: The crossed-slits projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(6):741–754, 2003.

[23] D. G. Lowe. Fitting parameterized three-dimensional models to image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(5):441–450, May 1991.

[24] T. Q. Phong, R. Horaud, and P. D. Tao. Object pose from 2-d to 3-d point and line correspondences. *International Journal of Computer Vision*, pages 225–243, 1995.

[25] M. Dhome, M. Richetin, J. T. Lapreste, and G. Rives. Determination of the attitude of 3-d objects from a single perspective view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(12):1265–1278, December 1989.

[26] R. Y. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 364–374, Miami Beach, 1986.

[27] B. Wilburn, N. Joshi, V. Vaish, M. Levoy, and M. Horowitz. High-speed videography using a dense camera array. In *IEEE Computer Society Conference on Pattern Recognition*, 2004.