2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

High-Speed Pose and Velocity Measurement from Vision

Redwan Dahmouche, Omar Ait-Aider, Nicolas Andreff and Youcef Mezouar
LASMEA - CNRS - Université Blaise Pascal
63175 Aubière, France
{firstname.lastname}@lasmea.univ-bpclermont.fr

Abstract— **This paper presents a novel method for high speed pose and velocity computation from visual sensor. The main problem in high speed vision is the bottleneck phenomenon which limits the video rate transmission. The proposed approach circles the problem out by increasing the information density instead of the data rate transmission. This strategy is based on a rotary sequential acquisition of selected regions of interest (ROI) which provides space-time data. This acquisition mode induces an image projection deformation of dynamic objects. This paper shows how to use this artifact for the simultaneous measure of both pose and velocity, at the same frequency as the ROI's acquisition one.**

I. INTRODUCTION

Vision is used at several levels in robotics, particularly in localization, identification [1] and control [2], [3]. However the slow rate of video sensors is an evident drawback in high sampling frequency applications. Indeed, standard high-speed cameras video rate is about 120Hz while high speed dynamic control application runs typically at 1kHz. Nevertheless, it has been reported that high-speed vision could be used in dynamic control of serial robots [4], where a General Predictive Control (GPC) scheme was associated to a visual loop linearisation to adapt the video rate (120 Hz) to the control sampling frequency (500 Hz). However, this solution increases control complexity. An alternative solution is to increase the video rate to reach the system sampling frequency. To do so, different approaches have been presented in the literature.

Usually, camera video rate is limited by the transmission interface bandwidth. Reducing the image resolution to decrease the video flow tightens a lot the field of view of the camera for a given accuracy of the end-effector pose estimation. To solve this problem, different approaches are possible such as video rate increasing by developing a more efficient video compression [5], creating faster transmission interfaces (for instance, CamLink) or embedding the signal processing close to the acquisition system [6], [7]. Nevertheless, we believe that the optimal solution is to increase the video flow information density. Indeed, the current approach in vision based applications is to grab and transmit the whole image, to extract interesting features to process and to throw away the rest of the image. For instance, to provide vision based pose estimation of a moving object from a single image, four non degenerate point projections are enough [8]. The ratio between the amount of data needed to perform the pose estimation and the transmitted flow of acquired image of size S is given by $\frac{4 \times 2 \times precision\ size}{S \times unsigned\ char\ size}$. For a mega-pixel image size the ratio is $6.4\ 10^{-5}$. The transmitted data is bigger than $1.5\ 10^{4}$ times the needed amount.

Instead of transmitting the whole image and then selecting a regions of interest (ROI), it is more interesting, from the data flow and the 'silicium cost' points of view, to inverse the process by first selecting the ROI position, and then to transmit it. Note that in the two cases, the ROI positions are predicted, so there is no difference between the two approaches if the rest of the image is not used. This acquisition mode was proposed in [9] where a new CMOS camera was designed to grab a simultaneously multiple ROIs.

The same approach can be performed using an "off-the-shelf" CMOS fast reconfigurable camera which uses the CamLink interface. A single rectangular area can be selected for shuttering and transmitting. Its parameters can be changed dynamically at each acquisition. By grabbing only areas in the scene that contain information, such as interest points or blobs (Figure 1), the information density in the video flow is increased. The direct effect of this is that the ROI acquisition frequency can be multiplied by the ratio of the full image size on the size of the grabbed area. For instance, transmitting ten regions of interest of 10×10 pixels that contain the desired information, instead of the 1024 $\times$ 1024 pixels image size, reduces the data flow from 1M pixels to 1K pixels, and theoretically multiplies the acquisition frequency by 1000. In practice, transmission control bits, parameters setting and exposure time limits the video rate. Note that the exposure time and the acquisition frequency can also be controlled.

Unfortunately, sequential acquisition of partial areas on the retina introduces time delay between acquisitions and affects image projection of moving objects. Thus, classical pose estimation algorithm can not be used in this case. In addition, these methods enable only to estimate successive poses. The velocity information is generally retrieved by numerically differentiating the pose measurements. This introduces additional noise.

To compute pose and velocity at each sample, one approach consists in using data fusion methods from a set of partial time varying information (eg. Kalman filter [6]). However this approach assumes a Gaussian noise which is not guaranteed in pose measurement applications. To

This work was supported by Région d'Auvergne through the Innov@pôle project and by the European Union through the Integrated Project NEXT no. 0011815.

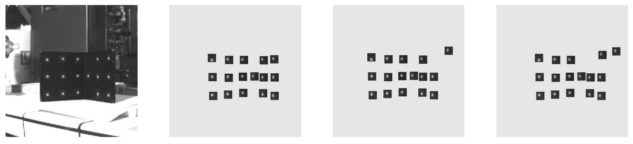
(a) (b) (c) (d)

Fig. 1. The resulting motion artifacts from ROI grabbing method. (a) The whole image taken by a standard camera. (b) ROI grabbing in static. (c) and (d) Two successive states of ROI grabbing when the object is moving.

overcome this constraint, an AEKF [10] could be used but the optimality of the method is not verified either. Another approach consists in considering the problem as an optimization process.

The idea of this paper is therefore to use the image artifacts for real-time high speed object pose and velocity computation based on non-linear least squares minimization. Previous works exploited the non-simultaneity of pixel exposure in CMOS Rolling Shutter camera. In this acquisition mode, the pixels are exposed row by row with a time delay introducing image artifacts when capturing moving objects [11]. The image projection depends on both pose and velocity. A method of simultaneous pose and velocity measurement using such artifacts from a single view was proposed in [12]. This is particularly adequate for high speed moving objects where high speed video rate is required.

The contribution of the paper is to generalize, to ROI random access, the theoretical results from [12]. Indeed, the latter were dedicated to rolling shutter cameras (*i.e.* assuming sequential grabbing of time coherent image rows). Moreover, in that work, the whole image was needed, which did not allow for fast estimation. On the opposite, the proposed extension is validated by an experimental 333Hz estimation frequency.

Next section presents the projection model of a rigid set of points acquired with a time delay. It will be shown that the time varying projection model depends on both pose and velocity, given a time acquisition period. In Section III, the proposed projection model is used to develop a method for pose and the velocity measurement for high speed moving object, using a single set of space-time feature matches. The optimization algorithm used in the estimation process and its particularities will also be discussed. Section IV deals with ROI tracking using the 3D pose and velocity prediction, which is the necessary complement to the method. Section V deals with implementation details, namely how to increase the pose and velocity estimation frequency and to reduce the latency. Finally, experimental results are presented and analyzed in Section VI.

II. Time varying camera projection model

A. Single point projection

First, let us consider the point projection pinhole model for classical cameras.

Let $\mathbf{P}$ be a 3D point lying on a rigid object, ${}^{o}\mathbf{P} = (x,\ y,\ z)^{\mathrm{T}}$ its coordinates in the object frame and $\mathbf{m} = (u, v)^{\mathrm{T}}$ its image projection. Let $\tilde{\mathbf{m}} = \left(\mathbf{m}^{\mathrm{T}}, 1\right)^{\mathrm{T}}$ and $\tilde{\mathbf{P}} = \left(\mathbf{P}^{\mathrm{T}}, 1\right)^{\mathrm{T}}$, be their homogeneous representations. Let ${}^{c}\mathbf{R}_o$, ${}^{c}\mathbf{t}_o$ and ${}^{c}\mathbf{T}_o$ be respectively the rotation matrix, the translation vector and the rigid transformation made between the camera frame and the object frame. Then, the perspective projection model is given by:

$$s\tilde{\mathbf{m}} = \mathbf{K}\begin{pmatrix}{}^{c}\mathbf{R}_o & {}^{c}\mathbf{t}_o\end{pmatrix}{}^{o}\tilde{\mathbf{P}} \tag{1}$$

where s is the unknown scale factor, and $\mathbf{K}$ the calibration matrix defined by the well known intrinsic camera parameters u_0, v_0, α_u and α_v which are respectively the horizontal and vertical optical center pixel coordinates and the scale factors along u and v axis. In the sequel, the intrinsic parameters are assumed to be known by calibration, as well as the distorsion parameters.

B. Projection of a rigid set of points

Historically, computer vision has always considered the use of the above point projection for CCD cameras, that grab all the pixels simultaneously. Thus, when considering a rigid object, viewed as a set of points, the resulting projection model becomes:

$$\forall i = 1..n,\ s\tilde{\mathbf{m}}_i = \mathbf{K}\begin{pmatrix}{}^{c}\mathbf{R}_o & {}^{c}\mathbf{t}_o\end{pmatrix}{}^{o}\tilde{\mathbf{P}}_i \tag{2}$$

where all 2D-3D point correspondences share the same pose $({}^{c}\mathbf{R}_o, {}^{c}\mathbf{t}_o)$.

In [12], a first attempt was proposed to release this constraint when considering CMOS rolling shutter cameras. However, the projection model proposed is dedicated to this specific kind of sensor and can be generalized as:

$$\forall i = 1..n,\ s\tilde{\mathbf{m}}_i(t_i) = \mathbf{K}\begin{pmatrix}{}^{c}\mathbf{R}_o(t_i) & {}^{c}\mathbf{t}_o(t_i)\end{pmatrix}{}^{o}\tilde{\mathbf{P}}_i \tag{3}$$

where now, each image point $\mathbf{m}_i$ may be grabbed at separate time instants t_i and thus, is associated to the current pose $({}^{c}\mathbf{R}_o(t_i), {}^{c}\mathbf{t}_o(t_i))$ at the time where it is grabbed.

Notice that the model proposed in [12] is included in this generic model by the definition of t_i as a function of the image line number and the image line exposure time.

With such a generic model, we can now even imagine a random access to the image pixels. However, in order to detect image points with sub-pixel accuracy, it is necessary to perform a local image treatment (such as blob or Harris point extraction). This means that to each point is associated a region of interest (ROI) whose size S can be tailored: not too small to get enough accuracy, not too large to keep a high grabbing frequency. Using this strategy, only $n{*}S$ pixels need to be grabbed instead of the whole image.

C. First order approximation of the rigid object projection

Consider now the high speed successive acquisition of n ROIs. The i^{th} ROI is grabbed at instant t_i, $\forall i = 1..n$. Assume that all the pixels in a given ROI can be grabbed simultaneously. This assumption is not very restrictive. Indeed, it only means that each ROI has to be grabbed as a whole, which is what is naturally implemented on the sensor electronics. Moreover, even if one can not ensure the exact simultaneity of each pixel grabbing, the total delay between the first and last pixel of the ROI will at most be S times the single pixel exposure and reading time. In such a short time (around 1 μs for a 10×10 ROI with current devices), the displacement of the object is small enough to generate movement artifacts lower than the extraction noise.

Under the first order approximation of the rigid object projection, the object velocity is assumed piecewise constant between the first and last point acquisition. This yields the following time-varying rigid object projection model:

$$\forall i = 1..n,$$
$$s\tilde{\mathbf{m}}_i(t_i) = \mathbf{K}\left({}^c\mathbf{R}_o(t_0)\delta\mathbf{R}_i \quad {}^c\mathbf{R}_o(t_0)\delta\mathbf{t}_i + {}^c\mathbf{t}_o(t_0)\right) {}^o\tilde{\mathbf{P}}_i \quad (4)$$

where $({}^c\mathbf{R}_o(t_0), {}^c\mathbf{t}_o(t_0))$ is the pose at a reference time t_0 and $(\delta\mathbf{R}_i, \delta\mathbf{t}_i)$ is the displacement of the object between t_0 and t_i. Introducing $dt_i = t_i - t_0$ as well as ${}^o\Omega = \omega\, {}^o\underline{\mathbf{a}}$ and ${}^o\mathbf{v}_o$, the constant rotational and translational velocities of the object, the following expression is obtained:

$$\begin{bmatrix} \delta\mathbf{R}_i & \delta\mathbf{t}_i \\ 0 & 1 \end{bmatrix} = e^{\begin{bmatrix} [{}^o\Omega]_\times & {}^o\mathbf{V} \\ 0 & 0 \end{bmatrix} dt_i} \quad (5)$$

where $[\cdot]_\times$ denotes the skew-symmetric matrix associated to the vector cross-product.

Causality suggests to choose the reference time t_0 as the time when the first ROI is grabbed (*i.e.* $t_0 = t_1$). This takes the opposite of [12] where the reference time was taken when grabbing the first image row. However, the goal of this work being the real-time estimation of the instantaneous pose and velocity, this choice is troublesome. Indeed, such a choice means that the estimated pose is the pose of the object when grabbing the *first* ROI, but can not be estimated until the *last* ROI is grabbed. This yields a pure time delay of $n\ \tau$ in the estimation of the pose. Another consequence of this choice is that the velocity is estimated in the object frame at t_0. It is not time-delayed (since it is assumed constant over the time interval) but it is not expressed in the current object frame.

To remove the time delay, an intuitive approach consists in computing the integral over the image capture time of the velocity as obtained above and update the pose and velocity. However, this introduces superfluous computation and amplifies noise. A better way is to determine directly the pose and velocity at the current time. Therefore, the reference time t_0 must thus correspond to the last point acquisition: $t_0 = t_n$. The consequence of this choice is that the i^{th} point is captured at a negative time $dt_i = (i - n)\tau$ with respect to t_0.

III. Pose and Velocity Computation

Let us assume, for the moment, that a set of 2D-3D correspondences is available. Given the camera parameters, the scale factor is classically removed from (4) to form the image error:

$$\epsilon_i^{(u)}(X) = \alpha_u \frac{\mathbf{r}_{x,i}^{\mathrm{T}}\mathbf{P}_i + \mathbf{t}_{x,i}}{\mathbf{r}_{z,i}^{\mathrm{T}}\mathbf{P}_i + \mathbf{t}_{z,i}} + u_0 - u_i \quad (6)$$

$$\epsilon_i^{(v)}(X) = \alpha_v \frac{\mathbf{r}_{y,i}^{\mathrm{T}}\mathbf{P}_i + \mathbf{t}_{y,i}}{\mathbf{r}_{z,i}^{\mathrm{T}}\mathbf{P}_i + \mathbf{t}_{z,i}} + v_0 - vi \quad (7)$$

where $\mathbf{t}_{\bullet,i}$ are the components of ${}^c\mathbf{R}_o(t_0)\delta\mathbf{t}_i + {}^c\mathbf{t}_o(t_0)$ and $\mathbf{r}_{\bullet,i}^{\mathrm{T}}$ are the rows of ${}^c\mathbf{R}_o(t_0)\delta\mathbf{R}_i$, whereas X contains the variables that represent the unknown pose and velocity.

This set is chosen in a standard manner, except for one numerical simplification. Indeed, the product ${}^c\mathbf{R}_o(t_0)\delta\mathbf{t}_i$ in (4) couples several unknowns. However, noticing that this product is simply a change of coordinates in the expression of the translational velocity:

$$ {}^c\mathbf{R}_o(t_0)\delta\mathbf{t}_i = {}^c\mathbf{R}_o(t_0)\,{}^o\mathbf{v}_o dt_i = {}^c\mathbf{v}_o dt_i \quad (8)$$

this product can simply be replaced by the latter expression, which is independent from the other unknowns. Therefore, the set of unknown is:

$$X = \{x, y, z, u_\theta x, u_\theta y, u_\theta z, v_x, v_y, v_z, \Omega_x, \Omega_x, \Omega_x\} \quad (9)$$

where $(x, y, z)^{\mathrm{T}} = {}^c\mathbf{t}_o(t_0)$ is the current position of the object frame, $(u_\theta x, u_\theta y, u_\theta z)^{\mathrm{T}} = {}^c\underline{\mathbf{u}}\theta$ is a minimal axis-angle representation of the current orientation of the object frame, $(v_x, v_y, v_z)^{\mathrm{T}} = {}^c\mathbf{v}_o$ is the current translational velocity of the object, expressed in the camera frame, and $(\Omega_x, \Omega_x, \Omega_x)^{\mathrm{T}} = {}^o\Omega$ is the current rotational velocity of the object, expressed in the current object frame. Finally, the pose and velocity are estimated through the solution of the following minimization problem:

$$\min_X \frac{1}{2}\sum_{i=1}^{n} \epsilon_i^{(u)}(X)^2 + \epsilon_i^{(v)}(X)^2 \quad (10)$$

A necessary condition for the 12 unknown values to be estimated is that, at least, 6 correspondences are available, each point providing two equations. This condition is only necessary but not sufficient because the equation system may loose rank. For instance, for degenerate object structure or specific pose and velocity configurations that still have to be studied. Moreover, to handle the noise coming from the point extraction from the ROIs, one should over-constrain this minimization problem.

IV. ROI tracking

In order to solve the above problem, one needs, on the one hand, an initial pose and velocity estimate and, on the other hand, 2D-3D correspondences. The latter can be ensured if a point tracking algorithm is implemented. Tracking is easy if a given ROI contains a single point, which additionally simplifies the point extraction. However, grabbing a ROI has now a strict physical meaning: one has to grab it at the correct place on the CMOS array, otherwise the expected point will not appear in the ROI. Tracking is not anymore just a local search in a whole image (containing all the information) but a global search on a local subpart of the image (which might contain no information at all). Therefore, the prediction stage of the tracker is crucial. Nevertheless, it is here trivial since, if a valid pose and velocity estimation is available at a given time, then it is very easy to estimate the image projection of the next point, simply by using the projection model (4) where t_0 is the current time and dt_i is the time to the next ROI capture.

```
// Init blob projection array
PrjBuff = InitProjections(Image);
// Parameters initialisation
Pose  = InitPose; // e.g Dementhon
Velocity = 0; // Null vector
X = [Pose; Velocity]
TDelay = $-\tau\ [n-1, n-2, ..., -1, 0]$;
do
{
  // Get new ROIs
  [ROIs, ROINbr] = GetLastROI();
  // Indices of new points
  Index = RefreshIndices(ROINbr);
  // Refresh the acquired blob coordinates
  PrjBuff(Index) = ImgPositions(ROIs);
  // New time delay vector
  TDelay = UpdateTime(ROINbr);
  // Update Pose and Velocity
  X = LSQ(Model,X,PrjBuff,TDelay);
  // Pose prediction by velocity integration
  XPred = ParamPredict(X);
  // Points projection prediction
  PrjPred = ImProject(Model,XPred,Index);
  // ROI Tracking
  ROIPos = SetRoiPositions(PrjPred);
}
```

Algorithm 1: Process algorithm

Consequently, the proposed pose and velocity estimation scheme fuses, in a single framework, 2D image tracking, pose (or 3D) tracking and, of course, velocity tracking.

Moreover, the proposed pose and velocity estimation scheme only requires an adequate initialization. This is easy in the case where a robot is observed, since one only has to grab the whole image when the robot is at rest. Indeed, if the observed object is still, motion artifacts vanish, $\delta\mathbf{R}$ becomes the identity, and $\delta\mathbf{t}_i$ turns to a null vector. The projection model corresponds to the classical one, and the object pose can be computed using the existing methods [13], [8]. If the need arises, the proposed algorithm can be initialized for a moving pattern, provided that an *a priori* pose and velocity estimate is available.

V. Implementation

A. The optimization Process

The minimization (10) of the image reprojection error provides pose and velocity parameters fitted to the given data. Different minimization methods exist. One of the main drawback of iterative methods is the possible existence of local minima if the initial parameters are far from the desired solution. Yet, this problem should not appear in the proposed algorithm, since the algorithm serves to update the solution, the new estimation is never being far from the last one. This particularity reduces (and hopefully eliminates) the risk of falling into a local minimum. The damped Gauss-Newton method [14] is thus well appropriate for this task, thanks to the good local convergence of the algorithm. This convergence can be controlled by a damping matrix, whose tuning allows for noise filtering.

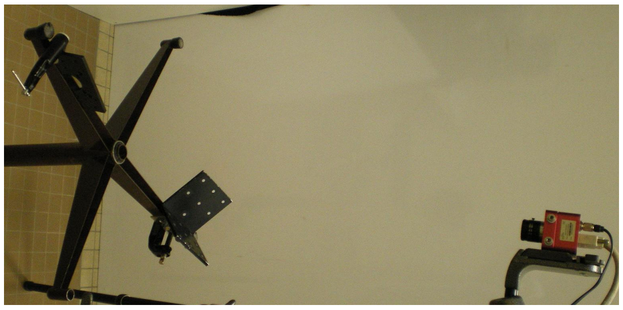

Fig. 2. Experimental device for the first experiment.

B. Computation cost

We know that the iterative methods are usually time consuming. Therefore, to reduce the cycle time, a closed-form expression of the Jacobian matrix $\mathbf{J}$ associated to the projection function (3) and the parameter set was used.

In the iterative process, given the last pose and velocity estimation, a pose prediction can easily be calculated under constant velocity assumption. The predicted pose can then be used to initialize the optimization process. The closeness of the initial prediction and the solution allows to compute the Jacobian matrix only once at the first iteration. Indeed, the stability condition of the optimization process is

$$\mathbf{J}\hat{\mathbf{J}}^{+} > 0 \tag{11}$$

where $\mathbf{J}$ is the actual Jacobian matrix and $\hat{\mathbf{J}}$ is an approximation of it.

Finally, the algorithm was implemented in C++ language using the *Numerical Template Toolbox* [15] based on Boost, BLAS and LAPACK libraries, which optimize performances.

VI. Experimental Results

For the practical validation of the proposed method, a 1024 $\times$ 1024 pixel high speed camera ”Photon focus Track Cam” enables ROI grabbing at different frequencies and allows for real-time ($9\mu s$) reconfiguration of the ROI position and size. The camera and the object, composed of 16 blobs, have been simultaneously calibrated using the method presented in [16]. The initialization of the algorithm is done from the whole still image. The blobs on the visual pattern are detected and the static pose is computed offline. After that, the high frequency sequential blobs acquisition starts. The ROIs size is selected 24×24 to be quite large to contain the blob with a definite tolerance in prediction and quite small to not increase the acquisition delay.

At each ROI acquisition, pose and velocity are updated using the proposed algorithm 1, and the next pose is predicted. The predicted pose projection is then used in tracking. The detection method in the ROI is based on a simple first moment computation of the thresholded image. The maximum speed of the vision system (acquisition and pose+velocity measurement) is 333 Hz. The first experiment is qualitative and deals with a rotational trajectory. Indeed, a visual pattern is fixed on a rotating mechanism (Figure 2)

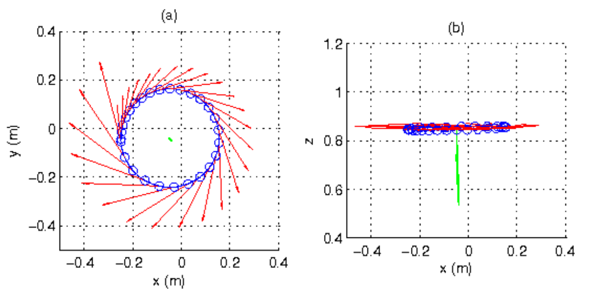

Fig. 3. Position and Velocity measurement of Rotating movement

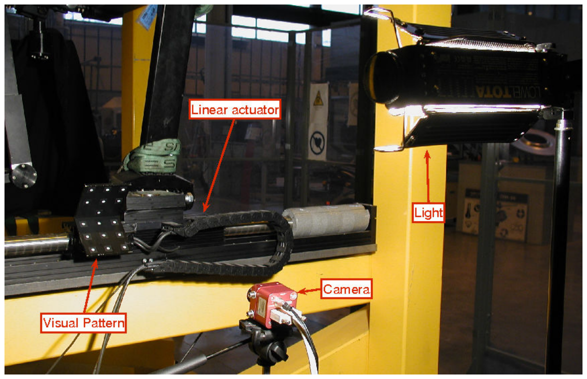

Fig. 4. Experimental device for the second experiment.

wherein rotation velocity accelerates progressively, generating a circular motion of the object in space. The pose and velocity are measured during the acceleration phase.

Figure 3 presents the reconstructed trajectory, projected on and orthogonally to the displacement plane. On this trajectory are superimposed the reconstructed translational and rotational velocity vectors. One verifies easily that the reconstructed trajectory is actually a circle in a plane. Moreover, the translational velocity stays tangent to the circle and its norm increases as expected. As for the rotation velocity vector, it stays orthogonal to the plane. The tracking is lost at about $11rad/s$ which corresponds to $2.4m/s$ tangent velocity and a normal acceleration of $22m/s^2$.

Consequently, the proposed method can correctly reconstruct a complex trajectory in space of a rapidly moving object.

The second experiment tries now to quantify the effective accuracy of such a reconstruction. It consists in measuring the pose and velocity of an object mounted on a fast linear actuator (Figure 4). The position of the latter is measured with a $1\mu m$ optical encoder at $1kHz$ sampling frequency. The velocity of the actuator is numerically derived from the position signal.

The desired trajectory is generated using a fifth order polynomial time interpolation, which corresponds to a maximum acceleration of 1G.

Figure 5 displays the joint trajectory, the estimated one and their difference. It can be noticed that the estimated trajectory

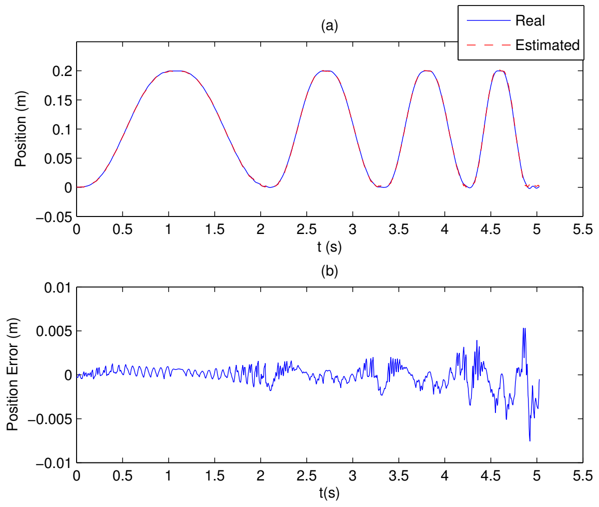

Fig. 5. (a) Real and estimated positions of the object translating on a linear actuator. (b) Position errors.

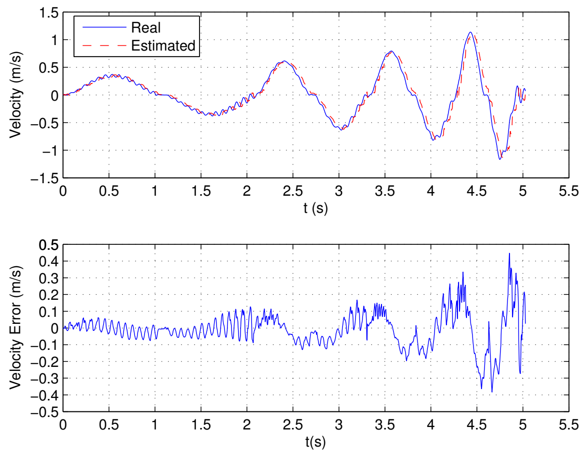

Fig. 6. (a) Real and estimated velocities of the object translating on a linear actuator. (b) Velocity errors.

correctly fits the reference one. More precisely, the position error has a mean value equal to 0.832 mm. This position error grows progressively with the velocities, and reaches its maximum values when the direction changes, corresponding to the highest values of acceleration. This might be explained by the fact that the projection model (4) assumes a constant velocity during the acquisition of all the points and that high values of acceleration do not cope with this assumption. The maximal position error value ($E_{max} = 7.5mm$) is reached at the last deceleration when object vibrates.

In static, the feature projection error has a mean of 0.32 pixel and a standard deviation of 0.11 pixel. In dynamic, the tracking error is added to the static one. The error mean in this case is 0.54 pixel and a standard deviation of 0.62 pixel.

It can be noted that the orientation estimate remained constant during the trajectory as expected.

Figure 6 (a) represents the real and the estimated veloc-

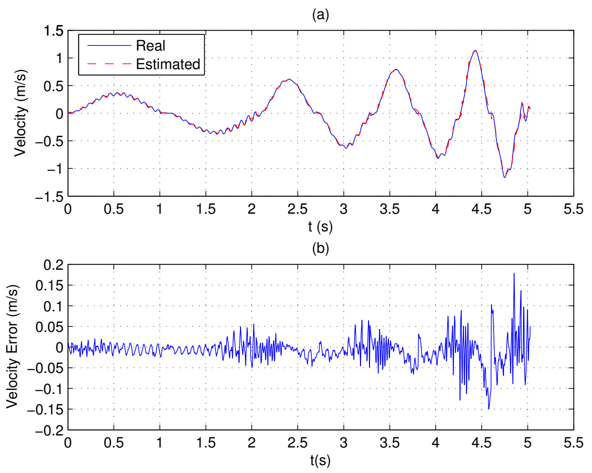

Fig. 7. (a) Real and estimated velocities of the object translating on a linear actuator, with compensation of the time delay. (b) Velocity errors, with compensation of the time delay.

ities. At first glance, one notices a time delay between the two. This delay is equal to $\Delta t = 0.024s$ equivalent to 8 samples. It represents exactly half the number of points in the object. The explication of that can be that under the constant velocity assumption, the optimization process converges to the mean of the object velocity between the 1^{st} and the 16^{th} point, which, due to the regularity of the trajectory, is close to the median velocity over the 8 samples. Consequently, the velocity error in Figure 6 presents a rather high deviation (about 0.1 m/s).

Nevertheless, to support the above analysis, Figure 7 shows that when shifting the estimated velocities by the time delay, the estimated velocities fit much better the real ones. Indeed, the variance is one order of magnitude lower than previously (now, about 0.01 m/s).

Hence, the velocities are correctly estimated but are subject to a constant time delay, which will have to be accounted for at control time.

VII. Conclusion and Perspectives

This paper has presented a novel concept of high speed pose and velocity measurement, based on visual ROI grabbing method. Indeed, the artifacts introduced by the sequential grabbing model which depends on both pose and velocity, are exploited, thanks to an extended projection model, for the simultaneous computation of 6 dof pose and velocity. This extends previous results and opened the possibility to reach very high estimation frequencies. Currently, an estimation frequency of 333 Hz was reached with a fairly good accuracy, which makes the method available for high-speed robot control.

Yet, the method can be improved from a technical point of view. Indeed, a multi-thread approach could allow for faster ROI acquisition with the same estimation frequency. This would hence make the constant velocity assumption valid for motions with higher dynamics, and thus reduce the delay in the velocity estimation. Another way to increase the accuracy of the method is to replace the constant velocity assumption by a constant acceleration, or even to include additional knowledge on the object motion.

From a methodological point of view, it also opens the way to random access retinas, where one would grab exclusively the information needed.

Acknowledgements

The authors would like to warmly thank Joël Falcou and Flavien Paccot for their valuable help in the implementation and experimental validation of this work.

References

[1] 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*, 22(1):12–22, February 2006.

[2] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, 8(3):313–326, jun 1992.

[3] S. A. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE Trans. Robotics and Automation*, 12(5):651–670, October 1996.

[4] 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.

[5] I. Richardson. *H.264 and MPEG-4 Video Compression : Video Coding for Next-generation Multimedia*. J. Wiley, 2003.

[6] Wilson L.W, Hulls C.W, and Bell G.S. Relative end-effector control using cartesian position based visual servoing. *IEEE Transactions on Robotics and Automation*, 12(5):684–696, October 1996.

[7] Y. Nakabo, M. Ishikawa, H. Toyoda, and S. Mizuno. 1ms column parallel vision system and its application of high speed target tracking. In *IEEE Int. Conf. Robotics and Automation*, pages 650–655, San Francisco, USA, April 2000.

[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, 1995.

[9] M. Ulrich, M. Ribi, P. Lang, and A. Pinz. A new high speed CMOS camera for real-time tracking application. In *IEEE Int. Conference on Robotics and Automation*, pages 5195 – 5200, New Orleans, april 2004.

[10] Lippiello. Vincenzo, Siciliano. Bruno, and Villani. Luigi. Adaptive extended Kalman filtering for visual motion estimation of 3D objects. *Control Engineering Practice*, 15(1):123–134, May 2006.

[11] 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.

[12] O. Ait-Aider, N. Andreff, J.M. Lavest, and P. Martinet. Simultaneous object pose and velocity computation using a single view from a rolling shutter camera. In *Proceedings of the 9th European Conference on Computer Vision, ECCV'06*, volume 2, pages 56–68, Graz, Austria, May 7-13 2006.

[13] 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.

[14] H. Jiang and D. Ralph. Global and local superlinear convergence analysis of newton-type methods for semismooth equations with smooth least squares. In *Reformulation - Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods, M. Fukushima and L. Qi, eds. Kluwer Academic Publisher*, pages 181–209, Boston, 1999.

[15] J. Falcou and J. Serot. E.V.E., an object oriented SIMD library. *Scalable Computing: Practice and Experience*, 6(4):31–41, December 2005.

[16] 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.