This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE Transactions on Pattern Analysis and Machine Intelligence
The final version of record is available at http://dx.doi.org/10.1109/TPAMI.2019.2894395

Rolling Shutter Camera Absolute Pose

Cenek Albl, Zuzana Kukelova, Viktor Larsson and Tomas Pajdla, *Member, IEEE,*

***Abstract*—We present a minimal, non-iterative solutions to the absolute pose problem for images from rolling shutter cameras. The absolute pose problem is a key problem in computer vision and rolling shutter is present in a vast majority of today's digital cameras. We discuss several camera motion models and propose two feasible rolling shutter camera models for a polynomial solver. In previous work a linearized camera model was used that required an initial estimate of the camera orientation. We show how to simplify the system of equations and make this solver faster. Furthermore, we present a first solution of the non-linearized camera orientation model using the cayley parameterization. The new solver does not require an initial camera orientation estimate and therefore serves as a standalone solution to the rolling shutter camera pose problem from six 2D-to-3D correspondences. We show that our algorithms outperform P3P followed by non-linear refinement using rolling shutter model.**

***Index Terms*—Computer vision, camera absolute pose, rolling shutter, minimal problems**

I. Introduction

THE PERSPECTIVE-N-POINT problem (PnP) for calibrated cameras is the task of finding a camera orientation and translation from n 2D-to-3D correspondences. It is a key problem in many computer vision applications such as structure from motion, camera localization, object localization and visual odometry. PnP has been thoroughly studied in the past with first solution being published in 1841 by Grunert and later revisited in [10]. The PnP problem for calibrated cameras can be formulated as a system of simple polynomial equations and solved from three correspondences. Many authors focused on different formulations of the problem, comparing numerical stability, speed or methods how to calculate the camera pose from more than three correspondences, see e.g. [4], [9], [22], [26], [27], [33], [35].

In general, existing methods for calculating PnP can be divided based on two criteria. They use either (1) a minimal number of 2D-to-3D correspondences, usually within a RANSAC paradigm to improve robustness, or (2) use more than the minimal number of measurement to simplify the equations. Another division can be made between iterative algorithms and non-iterative algorithms, where the former usually requires some approximate solution to begin with.

Previously mentioned methods assume a perspective camera model which is a model physically valid for cameras with a global shutter. However, CMOS sensors that are used in vast majority of today's consumer cameras, smartphones etc. use the rolling shutter (RS) mechanism [24] to capture images. The key difference is that with the global shutter, the entire image is exposed to the light at once, whereas when using the RS the individual image rows (or columns) are captured at different times. When a RS camera moves while capturing the image, several types of distortion such as smear, skew or wobble appear. A perspective camera model is no longer valid in this case and methods which do not model the rolling shutter effects will give poor estimates.

C. Albl (cenek.albl@gmail.com) and T. Pajdla (pajdla@cvut.cz) are with the Czech Institute of Informatics, Robotics and Cybernetics of the Czech Technical University in Prague

Z. Kukelova (kukelova@cmp.felk.cvut.cz) is with the Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague.

V. Larsson is with the Department of Mathematics, Faculty of Engineering, Lund University

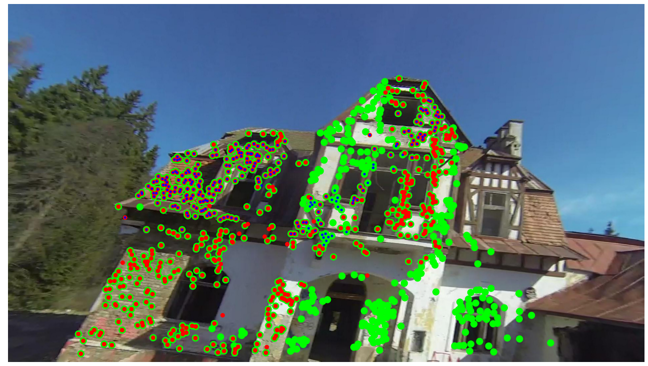


Fig. 1: Result of a standard P3P, P3P with local optimization using RS model and our R6P algorithm applied on image with high rolling shutter distortion. Inliers found among tentative 2D-3D correspondences for different algorithms are shown. Inliers found by P3P are blue, inliers found by P3P with local optimization are red and inliers found by R6P are green. Notice that R6P found many more matches than P3P and also than the locally optimized solution initialized by P3P.

Recent works have shown that RS is an important effect that should be considered in image rectification [17], [28], structure from motion [2], [13], [12] and multiple view stereo [29]. Those works have shown that existing methods can perform poorly on RS data or even fail completely and that incorporating some sort of RS camera model can solve these issues.

In [2] authors tackled the problem of RS absolute pose using a non-linear optimization with the initial guess obtained by a linear method using 8½ points and assuming a planar scene. We present the first minimal solution to the RS absolute pose problem from 6 non-planar points.

Another non-linear optimization method is presented in [12] which is well-suited for video sequences, where camera poses are computed sequentially taking the previous camera pose as an initial guess. In [18], the authors compensated for the RS effect prior to the optimization using estimated camera motion parameters from subsequent video frames. In contrast, our solvers work for single images exactly as P3P for perspective cameras.

A globally optimal solution using polynomial equations

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.
Copyright (c) 2019 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE Transactions on Pattern Analysis and Machine Intelligence
The final version of record is available at http://dx.doi.org/10.1109/TPAMI.2019.2894395

and Gloptipoly [14] solver to solve rolling shutter PnP is shown in [23]. Authors show that the method is capable of providing better results than [2] with the use of seven or more point correspondences. However, the runtime of this approach, which uses Gloptipoly, is at least 500 times longer than our solvers (0.3ms) and therefore [23] is not practical for using in a RANSAC loop. Moreover, when using more points, this approach is more sensitive to mismatches.

A first minimal solution to RS absolute pose was presented in [3]. It uses the Gröbner basis method to produce a solver which computes the camera pose and velocities from six correspondences (R6P). Due to the linearized model used for camera rotation matrix it requires an initial estimate of the camera orientation.

This paper is an extended version of [3] that presents a new solution to the R6P problem which does not require any initial estimate of camera orientation. Furthermore, we present a more efficient version of the original R6P double-linearized solver [3] which reduces computation time significantly. Lastly, this work provides a thorough comparison of the minimal solutions to non-linear optimization alternatives and gives a more thorough insight on the capabilities of R6P minimal solvers in various real-world experiments.

A. *Motivation*

It is important to note that rolling shutter effects are present even in still images and not only video sequences. Therefore it is desirable to have methods which only require a single image, which can be used for structure from motion, camera or object localization in cases where we don't have a video sequence or are too limited in computing resources to process every frame of a video. An example could be an unmanned aerial vehicle equipped with a camera for on-board localization or doing a large scale structure from motion reconstruction from a camera mounted on a car.

Contribution. This paper contains two non-iterative minimal solutions to the rolling shutter absolute pose (RnP) problem. These solutions provide more accurate camera pose estimates than standard P3P solution when images are affected by RS distortion and they work for still images as well as video sequences. Feasibility of several different RS camera models is analyzed and interesting experimental observations are made showing the benefits and limits of a RS camera model and a standard P3P model on a rolling shutter data.

We present a new solution for RnP from six-correspondences based on the cayley transform model (R6P-1lin). Unlike with the double-linearized model used in [3] the new solution does not require an initial estimate of camera orientation. Furthermore, using the hidden variable trick [7] to simplify the system of polynomial equations, we produced a more efficient solution to the original R6P [3] (R6P-2lin) reducing the computation time significantly. Instead of solving six equations in six unknowns we solve a system of equations in only three unknowns.

In addition to comparing the results of R6P-1lin, R6P-2lin and P3P minimal solvers we investigate the influence of subsequent local optimization. We compare the results obtained by RS solvers to using non-linear optimization with P3P result as an initial estimate. Local optimization is done using the full RS rotation model in LO-RANSAC and subsequent bundle adjustment. Extensive experiments are conducted in order to show what kind of improvement can be made and which approach provides the best results.

We investigate several rolling shutter camera models in section III and we discuss and verify their feasibility for a minimal solution to the RS absolute pose problem. In section IV we describe how to prepare the equations of the two models to be solved by a polynomial solver. Section V presents a method how to keep the data close to the linearization point where the double-linearized model works well. The resulting R6P solvers and subsequent local optimization approaches are thoroughly tested in section VII.

II. Absolute pose with rolling shutter

The computation of absolute camera pose using 2D and 3D point correspondences under the rolling shutter effect (RnP) brings new challenges. Standard PnP for perspective cameras uses the projection function

$$\lambda_i \mathtt{x}_i = \mathtt{R}\mathtt{X}_i + \mathtt{C} \tag{1}$$

where $\mathtt{R}$ and $\mathtt{C}$ is the rotation and translation bringing a 3D point $\mathtt{X}_i$ from world coordinate system to the camera coordinate system with $\mathtt{x}_i = [r_i, c_i, 1]^\top$, and scalar $\lambda_i \in \mathbb{R}$. For RS cameras, every image row will be captured at different time and hence at different positions when the camera is moving during the image capture. $\mathtt{R}$ and $\mathtt{C}$ will therefore be functions of the image row r_i being captured.

$$\lambda_i \mathtt{x}_i = \begin{bmatrix} r_i \\ c_i \\ 1 \end{bmatrix} = \mathtt{R}(r_i)\mathtt{X}_i + \mathtt{C}(r_i) \tag{2}$$

Next, we will describe functions $\mathtt{R}(r_i)$ and $\mathtt{C}(r_i)$.

III. Rolling shutter camera models

In this section we will consider several rolling shutter camera models and investigate their applicability for the RnP problem. For the camera translation we choose a simple constant velocity model which was used in [29], [23], [24], [12], [2]. Good results achieved by previous works suggest that constant velocity is a sufficient approximation for the short time-span of a frame capture. We can write $\mathtt{C}(r_i)$ as

$$\mathtt{C}(r_i) = \mathtt{C} + (r_i - r_0)\mathtt{t} \tag{3}$$

where $\mathtt{C}$ is the camera center corresponding to the perspective case, i.e. when $r_i = r_0$, and $\mathtt{t}$ is the translational velocity. By changing r_0 we can set the image line for which the rolling shutter model reduces to a perspective camera model. For our purposes we choose r_0 to be the middle row of the image, which will be justified later.

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE Transactions on Pattern Analysis and Machine Intelligence
The final version of record is available at http://dx.doi.org/10.1109/TPAMI.2019.2894395

A. SLERP model

To accommodate for camera rotation during frame capture we could interpolate between two orientations. A very popular method to interpolate rotations is SLERP [30], which was used in [12]. It works with quaternions and the formula to interpolate between two rotations represented by $\mathtt{q}_0$ and $\mathtt{q}_1$

$$SLERP(\mathtt{q}_0, \mathtt{q}_1, t) = \mathtt{q}_0 \frac{\sin(\Omega - t\Omega)}{\sin\Omega} - \mathtt{q}_1 \frac{\sin t\Omega}{\sin\Omega} \quad (4)$$

where $\Omega = \arccos\left(\mathtt{q}_0{}^\top \mathtt{q}_1\right)$. It is a linear interpolation on the sphere of quaternions, which in practice means that there will be a constant angular velocity. This is a nice property, which could even hold true in some special cases in reality (e.g. a camera mounted on a rotating platform or a car, which is turning with constant angular velocity). However, the presence of sine and cosine prevents us from using this model directly in a polynomial solver. We could substitute the sine and cosine with new variables and obtain a polynomial equations but that leads to high order polynomials and too complicated computations to get a fast solver. Moreover, with the increasing number of variables the solution becomes numerically unstable. This parameterization can be, however, utilized in the subsequent optimization step, such as bundle adjustment in [12].

B. Euler vector model

Euler vector is co-directional with the rotation axis and the length of the vector is equal to the rotation angle. One can then write a rotation matrix associated with Euler vector $\mathtt{e}$ using the Rodriguez formula

$$\mathtt{R}(\mathtt{e}) = \mathtt{I} + \left[\frac{\mathtt{e}}{||\mathtt{e}||}\right]_\times \sin||\mathtt{e}|| + \left[\frac{\mathtt{e}}{||\mathtt{e}||}\right]_\times^2 (1 - \cos||\mathtt{e}||), \quad (5)$$

where $[\cdot]_\times$ is a skew symmetric matrix. Using Euler vector to describe camera rotation during capture yields the same advantages and drawbacks as using SLERP, i.e. multiplying the Euler vector by r_i provides constant rotational velocity, but we have to deal with sine and cosine which makes it not practical for a minimal solver. It is, however, more practical than SLERP since we don't have to carry around two quaternions. The minimal solver we present in this paper is based on a linearization of this model, which makes it easier to convert between those two and therefore easily initialize subsequent local optimization.

C. Cayley transform model

Another way to represent rotations is the Cayley transform [11]. For any vector $\mathtt{a} = [x, y, z]^\top \in \mathbb{R}^3$ there is a map

$$\mathtt{R}(\mathtt{a}) = \frac{1}{K}\begin{bmatrix} 1+x^2-y^2-z^2 & 2xy-2z & 2y+2xz \\ 2z+2xy & 1-x^2+y^2-z^2 & 2yz-2x \\ 2xz-2y & 2x+2yz & 1-x^2-y^2+z^2 \end{bmatrix} \quad (6)$$

where $K = 1+x^2+y^2+z^2$ which produces the rotation matrix corresponding to the quaternion $w + ix + jy + kz$ normalized so that $w = 1$. The vector $\mathtt{a}$ is a unit vector of the axis of rotation scaled by $\tan\theta/2$, where θ is the rotation angle, thus 180 degree rotations are prohibited.

D. R6P formulation with Cayley transform model

We can prescribe (2) such that $\mathtt{R}$ is a combination of two rotations written using Cayley transform as

$$\lambda_i \begin{bmatrix} r_i \\ c_i \\ 1 \end{bmatrix} = \mathtt{R}((r_i - r_0)\mathtt{w})\mathtt{R}(\mathtt{v})\mathtt{X}_i + \mathtt{C} + (r_i - r_0)\mathtt{t} \quad (7)$$

to represent the camera initial orientation by $\mathtt{v}$ and the change of orientation during frame capture by $(r_i - r_0)\mathtt{w}$. This represents a rotation around the axis $\mathtt{w}$ which is close to uniform in angular velocity around $r_i = r_0$. Equation (7) is a rational polynomial and we must multiply it by $1 + x^2 + y^2 + z^2$ for both $\mathtt{R}(\mathtt{v})$ and $\mathtt{R}(\mathtt{w})$ to get a pure polynomial for the polynomial solver. We obtain a system of polynomial equations of degree five in 18 (3+3+3+3+6 for $\mathtt{C}$,$\mathtt{v}$,$\mathtt{t}$,$\mathtt{w}$ and $\lambda_1 \dots \lambda_6$ respectively) variables which contains 408 monomials. Such a system is difficult to solve and the Gröbner basis solution for this system involves eliminating a 8000x8000 matrix, which is time consuming and numerically very unstable. Due to these reasons we will not consider this model as feasible for our purposes.

E. R6P formulation with linearized model - R6P-1lin

To reduce the degree of the polynomials and the number of monomials we can use a linearization of rotation matrices. We will linearize $\mathtt{R}(\mathtt{w})$ around the initial rotation $\mathtt{R}(\mathtt{v})$ using the first order Taylor expansion such that

$$\lambda_i \begin{bmatrix} r_i \\ c_i \\ 1 \end{bmatrix} = (\mathtt{I} + (r_i - r_0)[\mathtt{w}]_\times)\,\mathtt{R}(\mathtt{v})\mathtt{X}_i + \mathtt{C} + (r_i - r_0)\mathtt{t} \quad (8)$$

where $[\mathtt{w}]_\times$ is the skew-symmetric matrix

$$[\mathtt{w}]_\times = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix} \quad (9)$$

Such model, with the rolling shutter rotation linearized, was used in [23]. This function will deviate from the reality with increasing rolling shutter effect. However, we have observed that this model is usually sufficient for the amount of rolling shutter rotation present in real situations. It is practical to set r_0 as the middle row of the image, so that the error of the model is spread over the image symmetrically. We now have a system of degree three polynomials in 18 variables with 64 monomials. In section IV-C we will show how to simplify the equations and produce a viable polynomial solver.

F. R6P formulation with double linearized model - R6P-2lin

Let us simplify the model even further by linearizing also the initial rotation. We obtain

$$\lambda_i \begin{bmatrix} r_i \\ c_i \\ 1 \end{bmatrix} = (\mathtt{I} + (r_i - r_0)[\mathtt{w}]_\times)(\mathtt{I} + [\mathtt{v}]_\times)\mathtt{X}_i + \mathtt{C} + (r_i - r_0)\mathtt{t} \quad (10)$$

which are simpler polynomial equations of degree two and 28 monomials. The model has an obvious drawback and that is, unlike $\mathtt{w}$ representing the rolling shutter motion and being

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.
Copyright (c) 2019 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE Transactions on Pattern Analysis and Machine Intelligence

presumably small, $\mathtt{v}$ can be arbitrary. Therefore, the model's accuracy would depend on the initial orientation of the camera in the world frame. A possible solution is to force $\mathtt{v}$ to be close to zero and we propose a way how to do this in section V . In section IV we will show how to simplify equations (10) to make the computation more efficient and produce a fast polynomial solver.

IV. R6P for RANSAC

In previous section we presented several rolling shutter camera models and in this section we show how to produce efficient solvers for the single and double linearized models suitable for RANSAC environment. To solve the polynomial equations of the two viable models (10,8) we use the Gröbner basis method [7]. This method for solving systems of polynomial equations has been recently used to create very fast, efficient and numerically stable solvers to many difficult problems. The method is based on polynomial ideal theory and special bases of the ideals called Gröbner bases [7].

A. Preparing the equations

We start with the first part that is similar for both solvers. The minimal number of 2D-to-3D point correspondences necessary to solve the absolute pose rolling shutter problem is six. For six point correspondence, both models (10,8) result in a quite complex system of $3 \times 6 = 18$ equations in 18 unknowns $(\lambda_1, \dots, \lambda_6, \mathtt{v}, \mathtt{w}, \mathtt{C}, \mathtt{t})$. Such a system is not easy to solve for the Gröbner basis method and therefore it has to be simplified.

To simplify the input system (10,8) we first eliminate the scalar values λ_i by multiplying all equations (10,8) from the left by the skew symmetric matrix

$$\begin{bmatrix} 0 & -1 & c_i \\ 1 & 0 & -r_i \\ -c_i & r_i & 0 \end{bmatrix}. \tag{11}$$

This leads to 18 equations, from which only 12 are linearly independent. For the double linearized model (10) they contain 22 monomials and 12 unknowns, i.e., the rotation parameters $\mathtt{w}, \mathtt{v}$ and the translation parameters $\mathtt{C}$ and $\mathtt{t}$.

For the single linearized model (8) we also need to multiply the equations by $(v_1^2+v_2^2+v_3^2+1)$ to get rid of the denominator coming from the cayley parameterization of $\mathtt{R}(\mathtt{v})$ (6). We need to keep in mind that $\mathtt{C}$ and $\mathtt{t}$ get multiplied as well and we obtain $\hat{\mathtt{C}} = \mathtt{C}(v_1^2 + v_2^2 + v_2^2 + 1)$ and $\hat{\mathtt{t}} = \mathtt{t}(v_1^2 + v_2^2 + v_2^2 + 1)$ and after solving for $\hat{\mathtt{C}}$ and $\hat{\mathtt{t}}$ we need to use $\mathtt{v}$ to obtain $\mathtt{C}$ and $\mathtt{t}$. The equations now contain 86 monomials in $\mathtt{w}, \mathtt{v}, \hat{\mathtt{C}}, \hat{\mathtt{t}}$.

The 12 linearly independent equations are linear in the unknown translation parameters $\mathtt{C}$ and $\mathtt{t}$ (or $\hat{\mathtt{C}}, \hat{\mathtt{t}}$ for the single linearized model). Therefore, they can be easily eliminated from these equations. This can be done either by performing Gauss-Jordan (G-J) elimination of a matrix representing the 12 linearly independent equations or by expressing the six translation parameters as functions of the rotation parameters $\mathtt{w}$ and $\mathtt{v}$ and substituting these expressions to the remaining six equations.

After the simplification we obtain a system of six equations in six unknowns and 16 monomials for the double linearized model and 80 monomials for the single linearized models. This system has 20 solutions for the double linearized model and 64 solutions for the single linearized model.

The system of six equations in six unknowns can be directly solved using the Gröbner basis method and the automatic generator of Gröbner basis solvers [20]. The Gröbner basis solver generated using the automatic generator [20] performs one G-J elimination of the elimination template matrix. This matrix contains coefficients which arise from specific measurements, i.e., six 2D-to-3D point correspondences. Then the solutions to the rotation parameters $\mathtt{w}$ and $\mathtt{v}$ are found from the eigenvectors of the multiplication matrix created from the rows of the eliminated template matrix.

Such a Gröbner basis solver for the R6P rolling shutter problem using the double linearized model was proposed in [3], it requires the G-J elimination of a 196×216 matrix and computing the eigenvectors of a 20×20 matrix and it runs about 1.7ms. For the single linearized model the generated Gröbner basis solver was too large and unstable. In the next part we will describe how to simplify the problem further to produce better solvers.

B. R6P-2lin solver for the double linearized model

Here we present a much smaller solver to the R6P-2lin rolling shutter problem than the one proposed in [3]. First, note that six equations in six unknowns obtained from (10) by eliminating the scalar values λ_i and the translation parameters $\mathtt{C}$ and $\mathtt{t}$ are bilinear in the rotation parameters, i.e. they are linear with respect to $\mathtt{w}$ and $\mathtt{v}$. Therefore, we can rewrite these six equations as

$$\mathtt{M}(\mathtt{w}) \begin{bmatrix} \mathtt{v} \\ 1 \end{bmatrix} = \mathtt{0} \tag{12}$$

where $\mathtt{M}(\mathtt{w})$ is a 6×4 matrix whose elements are linear polynomials in $\mathtt{w} = (w_1, w_2, w_3)$.

Since $\mathtt{M}(\mathtt{w})$ has a null vector, it must be rank deficient. Therefore, all the 4×4 sub-determinants of $\mathtt{M}(\mathtt{w})$ must equal zero. This results in $\binom{6}{4} = 15$ polynomial equations which only involve the rotation parameters $\mathtt{w}$. These fifteen polynomial equations can be written in a matrix form as

$$\mathtt{M}\mathtt{m} = \mathtt{0}, \tag{13}$$

where $\mathtt{M}$ is a 15×35 coefficient matrix and $\mathtt{m}$ is a 35×1 vector of monomials in three unknowns w_1, w_2 and w_3.

Note, that in this way we have eliminated additional three unknowns $(v_1, v_2$ and $v_3)$ from our original system (10). Similar technique for eliminating unknowns from a bilinear system of polynomial equations was recently used in [34] to solve a minimal problem of estimating the motion of a multi-camera rig.

The system of fifteen polynomial equations in three unknowns (13) can be solved without the automatic generator of Gröbner basis solvers [20]. In this case, after performing G-J elimination of a coefficient matrix $\mathtt{M}$ we directly obtain a Gröbner basis for the ideal generated by the input fifteen polynomial equations. Therefore, to construct a special 20×20

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE Transactions on Pattern Analysis and Machine Intelligence
The final version of record is available at http://dx.doi.org/10.1109/TPAMI.2019.2894395

multiplication matrix [7] $\mathtt{A}$ it is sufficient to perform G-J elimination of a 15×35 coefficient matrix $\mathtt{M}$ (13). Then the solutions to the rotation parameters $\mathtt{w}$ are found from the eigenvectors of the multiplication matrix $\mathtt{A}$ extracted from the eliminated $\mathtt{M}$. Solution to the remaining unknowns $(\mathtt{v}, \mathtt{C}, \mathtt{t})$ are found by back-substitution. The R6P-2lin solver constructed this way in C++ takes about 0.3 ms on a 2.5 GHz i7 CPU.

C. R6P-1lin solver for the single linearized model

For the single linearized model (8) we can use a similar approach as in the case of the double linearized model to obtain equations in only three unknowns. With the single linearized model the equations after the simplifications described in section IV-A are linear with respect to $\mathtt{w}$. This time we can rewrite these six equations as

$$\mathtt{M}(\mathtt{v}) \begin{bmatrix} \mathtt{w} \\ 1 \end{bmatrix} = \mathtt{0} \tag{14}$$

where $\mathtt{M}(\mathtt{v})$ is again a 6×4 matrix which elements are now second degree polynomials in $\mathtt{v} = (v_1, v_2, v_3)$. Again we have $\binom{6}{4} = 15$ polynomial equations in the rotation parameters $\mathtt{v}$ coming from the 4×4 subdeterminants of $\mathtt{M}(\mathtt{v})$ that must be equal to zero. However, since the elements of $\mathtt{M}(\mathtt{v})$ are now quadratic in $\mathtt{v}$ this yields equations of degree 8.

These fifteen polynomial equations can again be written in a matrix form as

$$\mathtt{M}\mathtt{m} = \mathtt{0}, \tag{15}$$

where $\mathtt{M}$ is a 15×165 coefficient matrix and $\mathtt{m}$ is a 165×1 vector of monomials in three unknowns v_1, v_2 and v_3.

Unfortunately this time we introduced a two-dimensional family of false solutions when we eliminated $\mathtt{w}$. These solutions correspond to the first three columns of $\mathtt{M}(\mathtt{v})$ becoming linearly dependent. Then there will exist vectors in the nullspace of $\mathtt{M}(\mathtt{v})$ on the form,

$$\mathtt{M}(\mathtt{v}) \begin{bmatrix} \mathtt{w} \\ 0 \end{bmatrix} = \mathtt{0}, \tag{16}$$

which are not solutions to the original system. Studying these solutions revealed that they are all complex and satisfy

$$1 + v_1^2 + v_2^2 + v_3^2 = 0. \tag{17}$$

These do not correspond to valid scaled rotation matrices, e.g. the solution $\mathtt{v} = (i, 0, 0)$ corresponds to

$$\mathtt{R}(\mathtt{v}) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & -2i \\ 0 & 2i & 2 \end{bmatrix}. \tag{18}$$

Fortunately the 15 polynomials were all divisible by (17). After dividing we have 15 equations of degree 6, which are only satisfied by the original 64 solutions.

Using the recent automatic generator technique from [21] we created a minimal solver for this system. The minimal solver uses an elimination template of size 99×163 to recover the 64×64 multiplication matrix $\mathtt{A}_{\mathtt{v}_3}$ [7]. Solution to v_3 can then be extracted fromthe eigenvalues of $\mathtt{A}_{\mathtt{v}_3}$ and solutions to v_1 and v_2 from its corresponding eigenvectors. After finding the solutions in $\mathtt{v} = \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix}^\top$ the remaining unknowns $(\mathtt{w}, \mathtt{C}, \mathtt{t})$ are found by back-substitution. The R6P-1lin solver constructed this way in C++ takes about 1.4 ms on a 2.5 GHz i7 CPU.

D. Pruning the solutions and improving performance

Usually only one of the 20 or 64 solutions of R6P-2lin and R6P-1lin is geometrically feasible, i.e., is real and of a reasonable values of parameters. Specifically, if we consider only reasonable values of the rolling shutter angular velocity $\mathtt{w}$ we can eliminate many solutions that are not feasible. Authors of [23] used the same linearization for $\mathtt{w}$ and showed that when $||\mathtt{w}|| > 0.05$ the model loses its accuracy. We decided to discard solutions with $||\mathtt{w}|| > 0.2$ which corresponds to angular velocity of approximately 11 degrees per frame. Solutions beyond this threshold are not interesting, since they are far from the linearization point. In our experiments, this criterion successfully eliminated 90-95% of solutions to be verified by RANSAC, which sped up the process significantly by avoiding lots of work in model verification.

For the R6P-1lin a significant portion of the computation time is spent computing the eigendecomposition of the 64x64 multiplication matrix $\mathtt{A}_{v_3}$. The multiplication matrix $\mathtt{A}_{v_3}$ is constructed in such way, that the eigenvalues are the solutions for the variable v_3. To avoid expensive eigendecomposition of $\mathtt{A}_{v_3}$ we can form the characteristic polynomial of the matrix $\mathtt{A}_{v_3} - \lambda\mathtt{I}$, e.g. by the Danilevsky method [8] and then find its roots using the Sturm sequences [16]. The roots of the characteristic polynomial will correspond to the eigenvalues of $\mathtt{A}_{v_3}$. This method of speeding up the computation was described in [5].

Note that the above method only gives us the values of the third Cayley parameter v_3 for the real solutions. To recover the full solutions we compute the eigenvectors corresponding to these eigenvalues (i.e. the values of v_3). This can be done by for each solution solving the 64×64 linear system,

$$(\mathtt{A}_{v_3} - v_3\mathtt{I})\mathtt{x} = 0. \tag{19}$$

The values for the remaining parameters v_1 and v_2 can then be extracted from $\mathtt{x}$ which has the following form (up to scale),

$$\mathtt{x} = [v_3^7,\ v_2^3 v_3^3,\ v_1^2 v_3^4,\ \ldots,\ v_1,\ v_2,\ v_3,\ 1] \in \mathbb{R}^{64}. \tag{20}$$

However, since we already know the value of v_3 and we know the structure of eigenvector $\mathtt{x}$, we can use this to slightly speed up the back-substitutions. Inserting the values for v_3 into $\mathtt{x}$, instead yields a 21×21 linear system in the unknowns

$$\hat{\mathtt{x}} = [v_1^5,\ v_1^4 v_2,\ v_1^3 v_2^2,\ \ldots,\ v_2^2,\ v_1,\ v_2,\ 1] \in \mathbb{R}^{21}. \tag{21}$$

The resulting linear system is smaller since some monomials collapse, e.g. the monomials $v_1^2 v_3^2$, $v_1^2 v_3$ and v_1^2 end up only yielding the single unknown v_1^2 in the new linear system.

V. R6P-2LIN - GETTING CLOSE TO THE LINEARIZATION POINT

As mentioned in section III, the double linearized model will only be a good approximation when close to the linearization point. That is the case when $\mathtt{R}(\mathtt{v})$ is close to $\mathtt{I}$. We can enforce

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.
Copyright (c) 2019 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE Transactions on Pattern Analysis and Machine Intelligence
The final version of record is available at http://dx.doi.org/10.1109/TPAMI.2019.2894395

this condition if we have an approximation $\mathtt{R}_a$ to $\mathtt{R}(\mathtt{v})$. Then we can transform the 3D points as

$$\hat{\mathtt{X}}_i = \mathtt{R}_a \mathtt{X}_i \quad i = 1, \dots, 6 \tag{22}$$

and replace $\mathtt{X}_i$ in (10) by $\hat{\mathtt{X}}_i$. Such solution $\mathtt{R}_l$ should then be close to $\mathtt{I}$ and we can obtain $\mathtt{R}(\mathtt{v})$ as $\mathtt{R}_l\mathtt{R}_a$. To get such approximation we can use for example an inertial sensor which is often present in cellphones, cameras or on-board a robot or UAV. However, we don't want to limit ourselves to having additional sensor information so we propose to obtain $\mathtt{R}_a$ using a standard P3P algorithm [10]. We will show in the experiments to which limits this approach works and that it can indeed provide a sufficient approximation for our solver to work well.

Notice that this approach requires only an approximation to camera orientation not the camera position.

VI. R6P-1lin - Staying in the domain

Since Cayley parameterization is not able to describe cameras rotated by 180 degrees, this poses a singular case for the R6P-1lin solver. In practice, the accuracy of the solutions returned by the solver will decrease when close to 180 degrees rotations. In general, not many dataset will contain such camera poses, but there is a general method to minimize the probability of the algorithm failing due to the presence of a singular case. Similar to section V we can pre-rotate the 3D points $\mathtt{X}_i \quad i = 1, \dots, 6$ as in (22), but this time we use a random rotation matrix $\mathtt{R}_a$. Using this approach we assure that the probability of being close to the singular case will be as low as possible and identical for all datasets. Moreover, when used in RANSAC, we can pre-rotate with different matrix $\mathtt{R}_a$ in each round, making the probability that we will encounter only singular cases for a given camera virtually zero. This approach was used in the DLS absolute pose solutions presented by [15], [25].

VII. Experiments

We conducted several experiments on synthetic as well as real datasets. The synthetic experiments were aimed at analyzing the properties of the double and single linearized rolling shutter camera models, which brings an interesting insight on how the solvers will behave under different conditions. On the real datasets, in the absence of ground truth, we focused on the number of matches classified as inliers using RANSAC. This corresponds to a typical application of absolute pose algorithm, where we want our model to be able to fit as many matches as possible while avoiding the mismatches. We compared our R6P solvers only to P3P solver [10], since to the best of our knowledge it is the only alternative with the same applicability. It is important to note that [12] can be used only on video sequences, [2] requires 9 co-planar points and [23] uses global optimization which is sensitive to mismatches and due to speed of Gloptipoly (in the orders of seconds) too slow for the RANSAC paradigm.

A. Synthetic data

In the synthetic datasets, a calibrated camera was considered with field of view of 45 degrees. It was randomly placed in a distance of $\langle 1; 3.3\rangle$ from the origin, observing a group of 3D points randomly placed in a cube with side length 2 centered around the origin. Camera initial orientation was different based on the type of experiment. The rolling shutter movement was simulated using the angle axis parameterization, because using the double linearized model or the single linearized model for generating data would allow R6P-2lin or R6P-1lin respectively to always find the exact solution. Although it would be interesting to simulate different kind of camera motions, it is generally agreed that camera motion during tens of miliseconds can be mostly treated as constant velocity motion [24], [23], [29], [12]. Therefore we generated the synthetic image projections using constant velocity camera motion to analyze the expected behavior of the solvers and verified the solvers later on real datasets with arbitrary camera motions. Six points were randomly chosen from the projections to solve for the camera parameters. Since R6P-2lin and R6P-1lin can return up to 20 and 64 real solutions respectively, the one closest to the ground truth was always chosen, as it would be probably chosen in the RANSAC estimation.

1) Handling the RS effect: The first experiment focused on varying the two rolling shutter parameters, i.e. the translational and angular velocity. The camera orientation is kept $\mathtt{R} = \mathtt{I}$ so we avoid the effect of the initial camera orientation linearization for R6P-2lin. We varied the angular velocity from 0 to 30 degrees per frame. Angular velocity 30 deg/frame means that the camera moved by 30 degrees between acquiring the first and the last row. The translational velocity was varied from 0 to 1 which is approximately 50% of the average distance of the camera center from the 3D points. The results are shown in Fig. 2. As expected, because the model does not exactly fit the data (as will be the case in real data), with increasing rolling shutter effect the performance of the solver decreases. However, the results are very promising since even at higher angular velocities the solver still delivers fairly precise results. At 28 deg/frame the mean orientation error is still below half a degree and the position error is less than half a percent. When varying the translation velocity only, we found the solver to be giving exact results up to the numerical precision, which was expected, since the model fits exactly the data.

2) Effect of the double linearization in R6P-2lin: The second experiment was focused on finding out how well R6P-2lin behaves around its linearization point, i.e. when $\mathtt{R} \neq \mathtt{I}$. Rolling shutter parameters were uniformly chosen from values $\langle 0; 20\rangle$ deg/frame for the angular velocity and $\langle 0; 0.2\rangle$ for the translational velocity, which is approximately ten percent of the average distance of camera center from the 3D points. Camera orientation was varied in the interval of $\langle 0; 30\rangle$ degrees. Results are in Fig. 3 and they show that R6P-2lin is very prone to error when being far from its linearization point, with the mean camera orientation error going up to five degrees and mean relative camera center error approaching 0.3 when the camera is rotated 30 degrees away from the linearization point. $\mathtt{C}$ and $\mathtt{R}$ are computed quite accurately,

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.
Copyright (c) 2019 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE Transactions on Pattern Analysis and Machine Intelligence
The final version of record is available at http://dx.doi.org/10.1109/TPAMI.2019.2894395

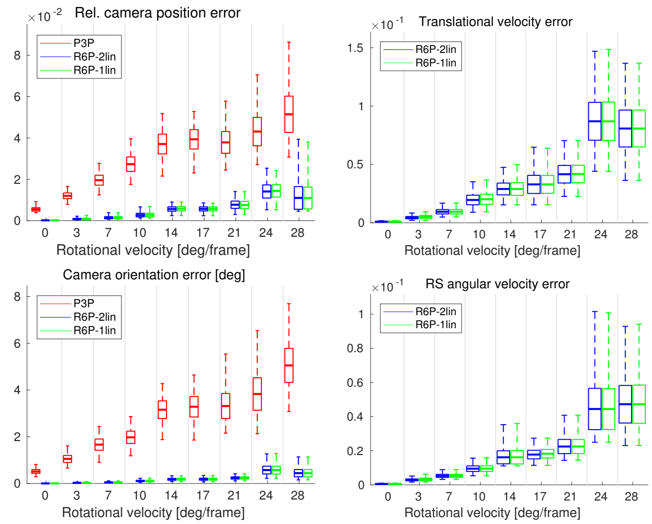


Fig. 2: Experiment 1 - results of the estimated camera pose and velocity for varying RS motion, increasing the camera rotation as well as camera translation velocity. The camera orientation for R6P-2lin is kept at $R = I$ to avoid the effect of the linearized camera orientation.

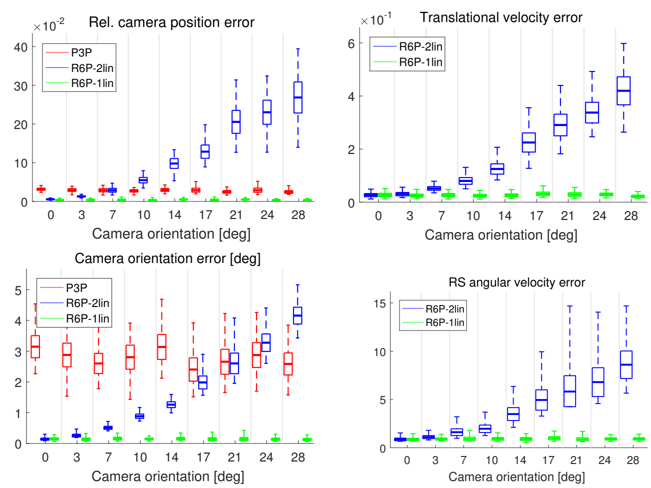


Fig. 3: Experiment 2 - varying camera orientation, showing the effect of the double linearization of R6P-2lin. The camera angular and translational velocities are randomly chosen to not exceed 20 deg/frame and 0.2 respectively.

when R is within approximately 6 degrees from I. It suggests that if we can use some standard non-RS method, such as P3P to find an initial R_0 to align the data, we can then apply our solver to get a more accurate camera pose. We tested this approach in experiment 3. R6P-1lin is not affected by the camera orientation as expected.

Results in Fig. 3 hint that the linearization is the key issue for R6P-2lin solver and that around 7 degrees of distance from $R = I$ our solver is surpassed in precision of estimating the camera center and at 14 to 17 degrees in the precision of estimating camera orientation. Note the interesting discrepancy between the error in camera position and orientation. An interesting thing to notice from these two experiments is that

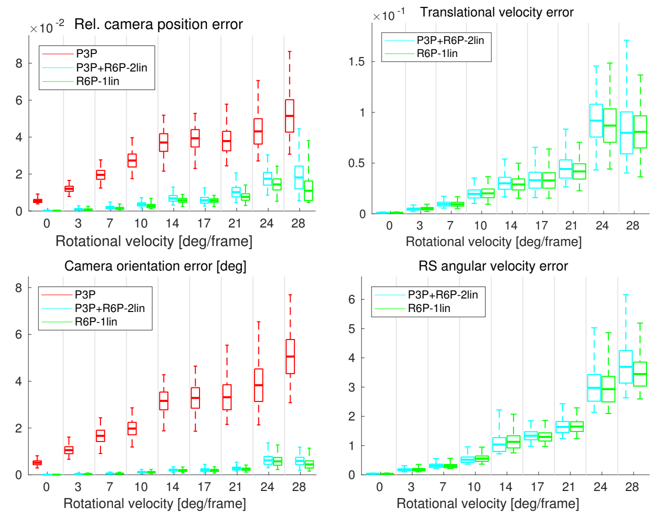


Fig. 4: Experiment 3 - increasing camera motion and comparing the single linearized model (R6P-1lin) to the double linearized model (R6P-lin2) initialized by P3P. A significant improvement is made using R6P-2lin after being initialized with P3P. R6P-2lin initialized by P3P provides comparable performance to R6P-1lin but is outperformed by R6P-1lin on large RS effects because the initial orientation provided by P3P is not good enough.

the global P3P solver was capable of bringing the camera orientation within six degrees of the ground truth, even under large RS effect. At that point, if we apply R6P-2lin solver, the precision should improve significantly to values below 0.5 deg.

3) Initialization of R6P-2lin by P3P: The purpose of experiment 3 is to verify that P3P can provide sufficient initialization for R6P-2lin. The camera orientation was chosen randomly and the RS parameters were increasing as in experiment 1. We compared P3P, R6P-1lin and R6P-2lin initialized by P3P by using the orientation provided by P3P to rotate the scene as described in section V. The results in Fig. 4 confirm our hypothesis. If the global P3P, or any other method, is able to compute the camera orientation within 6 degree error then R6P-2lin improves the solution to an average error below one degree. Interesting observation is that unlike in experiment 2 here the precision is significantly better for both camera center and orientation. It should also be noted, that R6P-1lin outperforms R6P-2lin as the RS effect increases, due to the deteriorating initialization provided by P3P.

B. Real data

We show two real world examples where our R6P solvers provide benefits. The first experiment is focused on estimating the camera absolute pose in the wild where the pose needs to be computed from image features possibly containing outliers. In a typical Structure from Motion application, the more 3D-2D matches are verified as inliers by the geometric camera model, the more 3D points will be reconstructed and the model better interconnected.

The second experiment shows an augmented reality scenario where known markers are placed in 3D space and they are

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.
Copyright (c) 2019 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE Transactions on Pattern Analysis and Machine Intelligence
This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication. The final version of record is available at http://dx.doi.org/10.1109/TPAMI.2019.2894395

detected automatically in the image. As camera absolute pose is computed using these markers, virtual objects can be placed into the scene. This represents a situation with a lot of camera movement and the need for a quick solver to run in real-time.

C. *Structure from motion*

In the first experiment we use datasets from [13] where an Iphone 4 camera was placed together with a global shutter Canon camera on a rig. Videos were taken when moving this rig by hand or walking around. We therefore have for each dataset two sets of images of the same scene. One set is with rolling shutter effect and one with global shutter.

1) Obtaining 2D-3D correspondences: To see the behavior of our method on real data, we needed to obtain 3D to 2D correspondences for the rolling shutter images. We decided to do a reconstruction using a standard SfM pipeline [32] using the global shutter images first. Then, we matched the rolling shutter images with the global shutter matches that had a corresponding 3D point in the global shutter 3D model. That way we obtained the correspondences between 2D rolling shutter features and 3D global shutter points. It was verified visually that this approach provided 2D-3D correspondences with a very small number of mismatches, i.e. 2D correspondences being matched to wrong 3D points. This is probably due to the fact, that all 3D points have already gone through an SfM pipeline and only good 3D points which were successfully matched in several cameras remained. Still, some mismatches were present, but according to our experiments, this number was not higher than 10%.

2) Evaluation: To evaluate our method, we measured the number of inliers, i.e. the 2D-3D correspondences in agreement with the model, after performing RANSAC. This is an important measure, since a common use of PnP is to calculate the camera pose and tentative 3D points for triangulation. The more points will be classified as inliers the more points will appear in the reconstruction and will support further cameras.

We first applied P3P to obtain $\mathtt{R}_a$ in equation (22), transformed the 3D points and then used our R6P-2lin solver as described in section V. Since our data contained only few mismatches, 1000 iterations of RANSAC proved to be enough to obtain a good camera pose. To reduce randomness of RANSAC results, we averaged the numbers over 100 successive RANSAC runs. The inlier threshold in RANSAC was set to 0.002 of the image diagonal length which in this case was approximately 2 pixels. As it is seen in Fig. 5, R6P-2lin is able to classify more points as inliers compared to P3P. The difference is significant especially when camera moves rapidly and/or the scene is close to the camera. This result confirms our expectation that as the camera movement during the capture becomes larger the need for a rolling shutter model is more significant. Datasets seq20 and seq22 contained more camera motion and therefore show a larger gap between results of P3P and R6P-2lin.

A good example is in dataset seq20, where the camera is fairly still in the beginning, then undergoes a rapid change in orientation (going upwards following the trunk of a palm tree), stops and then goes down again. The number of inliers returned by R6P-2lin and P3P when the camera is still is comparable, although higher for R6P-2lin since there are some RS distortions caused by handshake. As soon as the camera starts moving, the number of inliers for P3P drops drastically, sometimes even below 10% of the number of matches. R6P-2lin, in contrast, manages to keep the number of inliers above 90% of the number of matches. That is a huge difference.

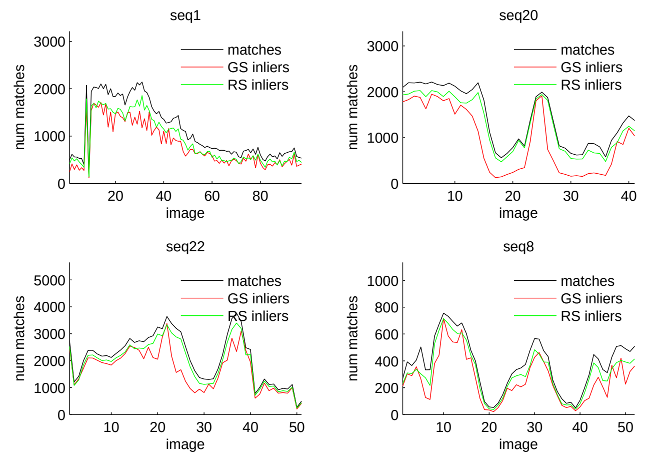


Fig. 5: Examples of experiments on real data. Number of inliers after running 1000 rounds of RANSAC, averaged over 100 RANSAC runs. Number of 2D-3D matches from global shutter images to rolling shutter images are in black, number of inliers obtained by P3P are in red and number of inliers obtained by R6P-2lin are in green. The results are averaged over 100 runs to reduce randomness.

Important observation is, that even though P3P fails to classify more than 80% of the matches as inliers it still provides a sufficient estimate of the camera orientation for the R6P-2lin to produce much better result. A detailed visualization of one of the results on seq20 is given in Fig. 6. We don't visualize the results of R6P-1lin because they were close to those of R6P-2lin and the detailed results can be found in table I.

3) Very fast motion: In previous experiments we evaluated the number of inliers as an indicator how well each solver performs. In this experiment we aim at showing the practical impact of using a perspective camera model on heavily distorted RS images and the improvement we can obtain by using R6P solvers.

We tested our solvers on heavily distorted RS images caused by fast camera motion that can arise in practice. A racing drone, carrying a GoPro camera and performing quick maneuvers is a good example of such data. We reconstructed the model of a building from approximately a hundred of still images from a classic digital camera combined with several images from the GoPro camera mounted on a drone which contained heavy RS distortion. This is a typical scenario where user has a number of images from which certain amount contains distortions caused by RS and fast movement.

We used a high quality state of the art SfM pipeline COLMAP [31] to reconstruct the scene, removing the radial distortion first. The building was reconstructed well, due to large amount of image data coming from the still images. Even though there were too few highly distorted RS images to

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.
Copyright (c) 2019 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE
The final version of record is available at http://dx.doi.org/10.1109/TPAMI.2019.2894395

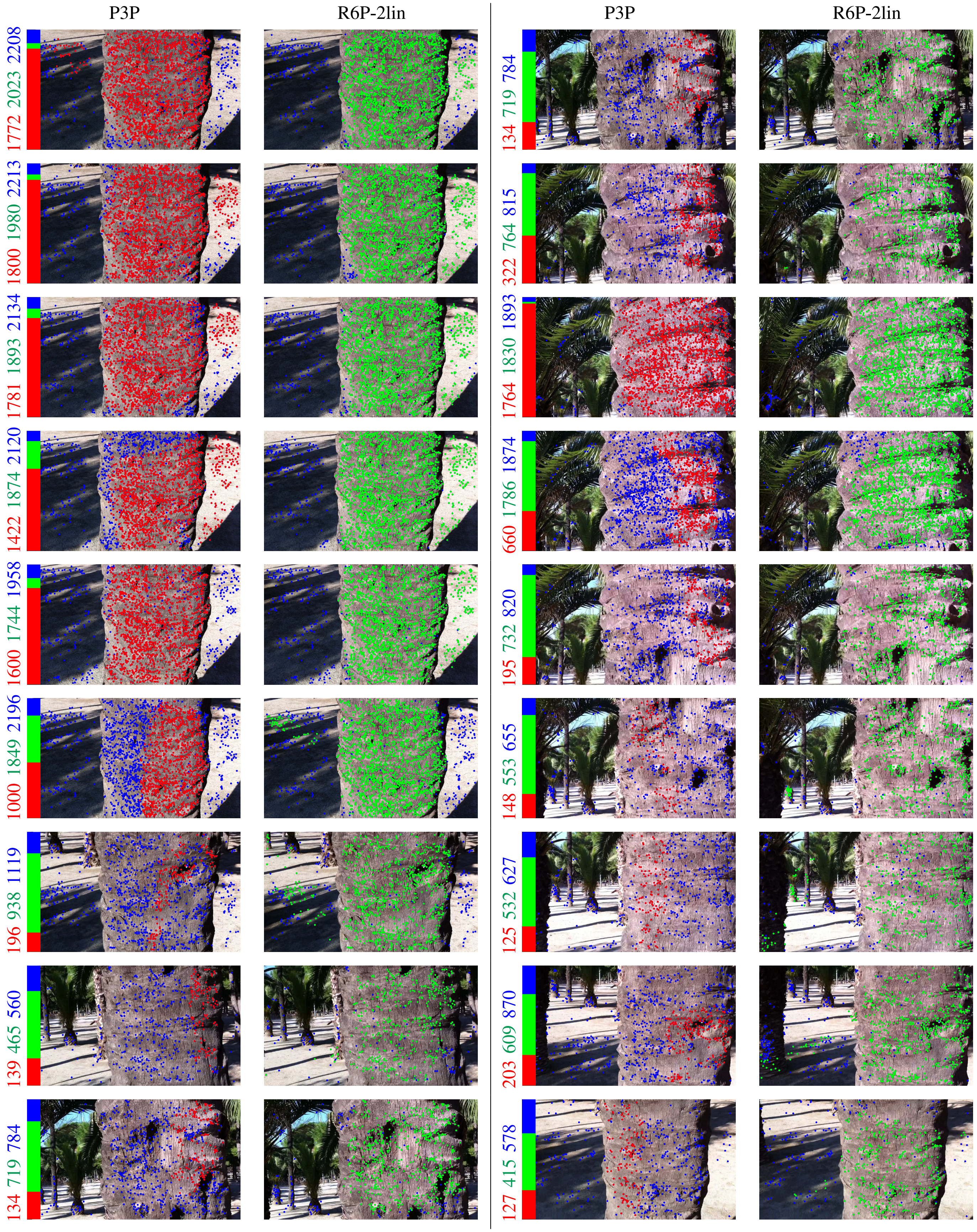


Fig. 6: Results on dataset seq20, matched correspondences are in blue, inliers after RANSAC using P3P and R6P-2lin are in red and green respectively. The actual numbers of inliers are displayed on side of each image pair.

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.
Copyright (c) 2019 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE Transactions on Pattern Analysis and Machine Intelligence
The final version of record is available at http://dx.doi.org/10.1109/TPAMI.2019.2894395

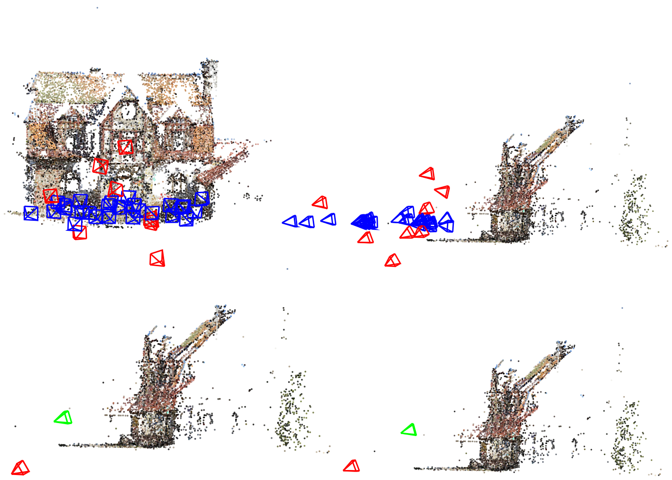


Fig. 7: 3D reconstruction of a building from large number of GS images (blue camera poses) and several images containing high level of RS distortion (red camera poses). Several RS cameras were clearly reconstructed wrong, having the pose under ground. The poses estimated using R6P-2lin (green) are much more realistic.

skew the model, the cameras coming from the RS images were reconstructed in clearly bad poses (red cameras in Figure 7). You can see that several RS images were reconstructed under ground.

We then created a new model using only the still images and registered the RS distorted images from GoPro camera to this model using method described in section VII-C1. The poses estimated using R6P-2lin (green cameras in Figure 7) roughly correspond to the poses where images were taken.

D. R6P and local optimization

An interesting question is whether one should use one of the two proposed R6P solvers or to locally optimize using one of the available Rolling-Shutter models starting from a P3P initialization. A popular approach called LO-RANSAC [6] uses the local optimization step after each accepted hypothesis in RANSAC. A camera model obtained by any method can be locally optimized using the points classified as inliers again using Bundle Adjustment (BA). We compared several meaningful approaches that represent possible practical use of the R6P algorithms and alternatives using local optimization. The key steps are following:

- P3P/R6P-2lin/R6P-1lin - RANSAC loop the corresponding solver
- LO-P - Local optimization inside RANSAC loop, followed by BA. Perspective model.
- LO-RS - Local optimization inside RANSAC loop, followed by BA. RS model.

We used LO-RANSAC and BA either with perspective or RS camera model with true rotation model from section III-B. The local optimization step was implemented using Google Ceres [1].

We tested many possible practical combinations of algorithms and the results are shown in table I in terms of minimal and average number of inliers over the entire sequences. Methods in the first, fourth and fifth column represent the most straightforward use of the P3P and R6P solvers respectively with no local optimization. Method in the second column represents the best result that can be obtained using a perspective camera model, utilizing LO-P. Third column represent the case when we avoid using R6P solvers, but locally optimize using a RS model. Approaches in the sixth trough eighth column use R6P solvers with LO-RS, therefore utilizing everything available to achieve the best results.

We observe again that R6P greatly outperforms P3P in terms of inliers found. P3P with LO-RANSAC and BA using perspective camera model does not improve over P3P itself, signalizing that RS model is certainly needed for this type of data. A significant improvement is made when using P3P and optimizing with RS camera model. Still, it does not achieve the performance of R6P solvers without LO. On most of the datasets, R6P itself provides higher number of inliers as you can see in Fig. 8. It is mostly apparent in the minimum number of inliers, which indicates critical cases when the camera movement is large. This can be explained by the P3P not being able to identify a large enough set of inliers, that would provide a good set to optimize the RS model on.

There is a measurable difference between P3P + R6P-2lin and R6P-1lin in the favor of the latter, signalizing that P3P as an initialization of R6P-2lin can hinder the performance of the RS solver in situations where the RS effect is large and that R6P-1lin overcomes this problem.

Performance of R6P can be further improved by applying local optimization with RS model. Using R6P solvers with subsequent local optimization with the true RS rotation model III-B provides the best performance across all datasets.

E. RANSAC threshold vs inliers

In this section we compare the number of inliers obtained by P3P and R6P-1lin with various RANSAC thresholds. As seen in the previous section on the real datasets P3P struggles to identify large portion of inliers on images with RS distortions compared to R6P-1lin with the same threshold. One could argue that increasing the threshold would allow P3P to capture more inliers, perhaps the same amount as in the case of R6P-1lin. To analyze this, we observed the number of inliers in each iteration of RANSAC under various thresholds. We chose dataset seq20 as a representative and average the results over 100 runs of RANSAC. The results in figure 9 show that in order to obtain the same percentage of inliers the threshold for P3P would have to be raised to 20 pixels, which is a significant increase compared to 2 pixels for R6P-1lin. Such large threshold could potentially lead to contamination by outliers.

F. Performance

To provide an idea about the performance of the proposed solvers in comparison to their alternatives we measured the time required by the sub-tasks in the RANSAC routine for each of the methods used in section VII-C. These sub-tasks include computing the camera pose, verifying the hypotheses and performing local optimization. Whereas computing the camera pose will always take approximately the same

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.
Copyright (c) 2019 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE Transactions on Pattern Analysis and Machine Intelligence
The final version of record is available at http://dx.doi.org/10.1109/TPAMI.2019.2894395

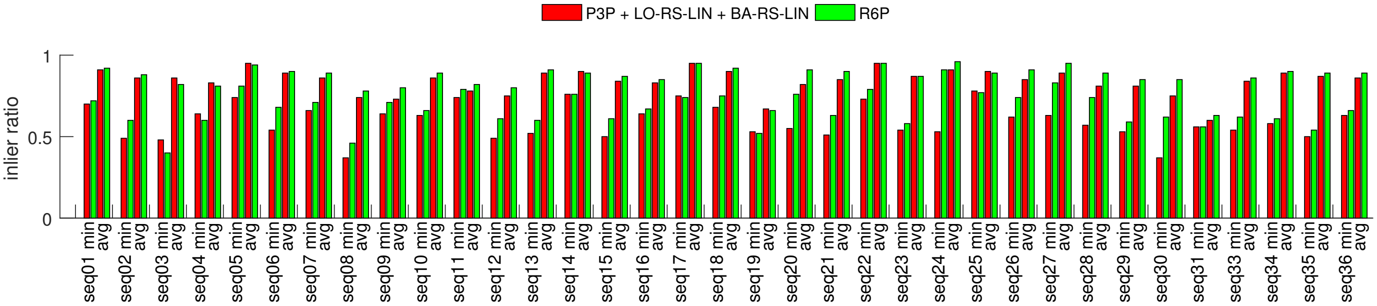


Fig. 8: Comparison of the minimal and average ratio of inliers provided by R6P-2lin versus P3P with local optimization using RS model. Notice that R6P alone without local optimization is better on most datasets. In 28 datasets R6P-2lin provided higher minimal inlier ratio and in 25 dataset better average inilier ratio. In the remaining datasets, the R6P-2lin and locally optimized P3P provided almost identical results. The results correspond to columns 3 and 4 in table I.

	P3P		P3P LO-P		P3P LO-RS		P3P R6P-2lin		R6P-1lin		P3P R6P-2lin LO-RS		P3P LO-RS R6P-2lin LO-RS		R6P-1lin LO-RS	
	min	avg	min	avg	min	avg	min	avg	min	avg	min	avg	min	avg	min	avg
seq01	0.53	0.77	0.48	0.76	0.70	0.91	0.72	0.92	0.72	0.91	**0.77**	**0.94**	0.76	**0.94**	**0.77**	**0.94**
seq02	0.34	0.71	0.34	0.76	0.48	0.86	0.60	0.88	0.60	0.88	**0.64**	**0.90**	**0.64**	**0.90**	**0.64**	**0.90**
seq03	0.39	0.76	0.40	0.77	0.60	0.89	0.72	0.89	0.73	0.89	0.75	**0.92**	**0.76**	**0.92**	0.75	**0.92**
seq04	0.54	0.76	0.56	0.77	0.71	0.87	0.65	0.88	0.67	0.88	**0.73**	**0.92**	**0.73**	**0.92**	**0.73**	**0.92**
seq05	0.37	0.82	0.38	0.82	0.67	0.94	0.83	0.94	0.83	0.95	**0.86**	**0.97**	**0.86**	0.96	**0.86**	**0.97**
seq06	0.46	0.81	0.48	0.82	0.52	0.89	0.66	0.89	0.68	0.90	0.75	0.94	**0.78**	**0.95**	0.75	0.94
seq07	0.44	0.69	0.44	0.70	0.65	0.85	0.72	0.89	0.70	0.89	**0.76**	**0.92**	**0.76**	**0.92**	**0.76**	**0.92**
seq08	0.32	0.62	0.28	0.60	0.34	0.74	0.46	0.78	0.49	0.78	**0.50**	0.80	**0.50**	**0.81**	**0.50**	0.80
seq09	0.28	0.42	0.29	0.43	0.58	0.72	0.70	0.80	0.71	0.80	**0.78**	**0.84**	**0.78**	**0.84**	**0.78**	**0.84**
seq10	0.47	0.69	0.48	0.70	0.62	0.85	0.66	0.89	0.67	0.89	**0.70**	**0.91**	**0.70**	**0.91**	**0.70**	**0.91**
seq11	0.57	0.64	0.58	0.65	0.72	0.76	0.78	0.81	0.81	0.82	0.81	**0.85**	**0.83**	**0.85**	0.81	**0.85**
seq12	0.27	0.57	0.28	0.58	0.54	0.75	0.59	0.80	0.60	0.80	**0.63**	**0.83**	0.62	**0.83**	**0.63**	**0.83**
seq13	0.41	0.74	0.42	0.74	0.53	0.89	0.60	0.91	0.63	0.92	**0.65**	**0.93**	**0.65**	**0.93**	**0.65**	**0.93**
seq14	0.55	0.84	0.56	0.84	0.73	0.90	0.77	0.89	0.75	0.89	**0.79**	**0.91**	0.78	**0.91**	**0.79**	**0.91**
seq15	0.46	0.68	0.46	0.68	0.51	0.84	0.64	0.87	0.62	0.87	**0.65**	0.89	**0.65**	**0.90**	**0.65**	0.89
seq16	0.50	0.69	0.52	0.70	0.65	0.83	0.68	0.85	0.70	0.85	0.73	**0.88**	**0.74**	**0.88**	0.73	**0.88**
seq17	0.65	0.78	0.66	0.80	0.75	**0.96**	0.74	0.95	0.76	0.95	0.78	**0.96**	**0.79**	**0.96**	0.78	**0.96**
seq18	0.53	0.74	0.55	0.75	0.66	0.90	0.74	0.92	0.75	0.92	**0.80**	**0.94**	0.79	**0.94**	**0.80**	**0.94**
seq19	0.48	0.63	0.49	0.64	0.53	0.68	0.51	0.66	0.53	0.67	**0.57**	**0.71**	**0.57**	**0.71**	**0.57**	**0.71**
seq20	0.20	0.55	0.21	0.56	0.52	0.80	0.81	0.89	0.82	0.90	0.82	**0.93**	**0.83**	**0.93**	0.82	**0.93**
seq21	0.31	0.59	0.32	0.60	0.51	0.84	0.64	0.90	0.63	0.90	**0.67**	**0.92**	**0.67**	**0.92**	**0.67**	**0.92**
seq22	0.48	0.81	0.48	0.82	0.67	0.94	0.81	0.95	0.81	0.95	**0.88**	**0.97**	**0.88**	**0.97**	**0.88**	**0.97**
seq23	0.37	0.73	0.38	0.74	0.52	0.87	0.57	0.87	0.59	0.88	0.62	**0.90**	**0.63**	**0.90**	0.62	**0.90**
seq24	0.34	0.78	0.36	0.79	0.82	0.95	0.92	0.96	0.92	0.96	**0.95**	**0.98**	**0.95**	**0.98**	**0.95**	**0.98**
seq25	0.47	0.74	0.48	0.75	0.79	0.90	0.76	0.89	0.77	0.90	**0.82**	**0.92**	**0.82**	**0.92**	**0.82**	**0.92**
seq26	0.21	0.58	0.22	0.59	0.64	0.84	0.74	0.91	0.74	0.91	0.78	**0.93**	**0.79**	**0.93**	0.78	**0.93**
seq27	0.26	0.61	0.26	0.61	0.63	0.87	0.83	0.95	0.83	0.95	**0.85**	**0.96**	**0.85**	**0.96**	**0.85**	**0.96**
seq28	0.24	0.56	0.27	0.57	0.47	0.78	0.74	0.89	0.75	0.89	**0.77**	**0.91**	**0.77**	**0.91**	**0.77**	**0.91**
seq29	0.37	0.67	0.38	0.67	0.50	0.81	0.60	0.85	0.59	0.85	**0.64**	**0.88**	0.62	**0.88**	**0.64**	**0.88**
seq30	0.20	0.49	0.21	0.50	0.33	0.72	0.62	0.85	0.62	0.85	**0.66**	**0.88**	**0.66**	**0.88**	**0.66**	**0.88**
seq31	0.41	0.50	0.42	0.51	0.53	0.59	0.55	0.63	0.56	0.63	**0.60**	**0.67**	0.58	**0.67**	**0.60**	**0.67**
seq33	0.29	0.68	0.30	0.69	0.52	0.83	0.61	0.87	0.61	0.87	**0.66**	**0.89**	**0.66**	**0.89**	**0.66**	**0.89**
seq34	0.32	0.79	0.33	0.80	0.73	0.94	0.87	0.96	0.87	0.96	**0.89**	**0.97**	**0.89**	**0.97**	**0.89**	**0.97**
seq35	0.34	0.72	0.35	0.73	0.50	0.87	0.54	0.89	0.54	0.89	**0.59**	**0.91**	0.58	**0.91**	**0.59**	**0.91**
seq36	0.40	0.75	0.40	0.76	0.51	0.88	0.56	0.89	0.56	0.89	0.58	**0.91**	**0.60**	**0.91**	0.58	**0.91**

TABLE I: Comparison of different uses of P3P and R6P solvers. Table shows the minimum and average number of inliers found by the approaches described in section VII-D. R6P itself provides most of the time significantly better results than running P3P and LO-RANSAC and BA with RS model as visualized by the red and green colors in columns 3,4 and 5. The best results overall, marked by a bold font, are provided by R6P and subsequent local optimization with RS model.

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.
Copyright (c) 2019 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE Transactions on Pattern Analysis and Machine Intelligence
The final version of record is available at http://dx.doi.org/10.1109/TPAMI.2019.2894395

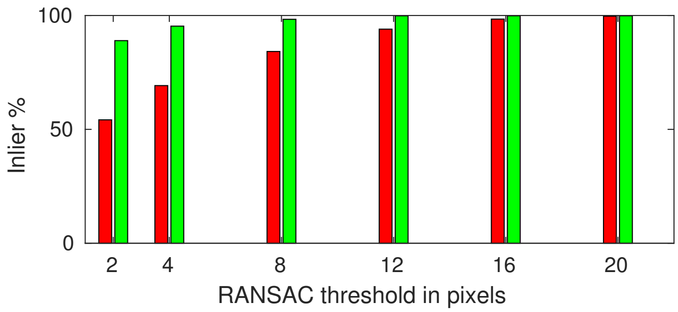


Fig. 9: Average number of inliers found by P3P and R6P RANSAC using different thresholds.

P3P	P-ver	R6P-2lin	R6P-1lin	RS-ver	LO-P	LO-RS
0.5	280	300	1400	800	9000	17000

TABLE II: Average timings (in microseconds) for different camera pose estimation tasks. 1000 correspondences assumed for the verification. The verification step also includes the average number of 2 hypotheses returned for both P3P and subsequent R6P-2lin or R6P-1lin which makes a total amount of 2000 correspondences to be verified.

Method	Pose	Verif.	LO	Verif.	Total
P3P	0.5ms	280ms			280.5ms
P3P LO-P	0.5ms	280ms	90ms	1.4ms	373.3ms
P3P LO-RS	0.5ms	280ms	170ms	4ms	454.5ms
P3P R6P-2lin	300.5ms	1080ms			1380.5ms
P3P R6P-2lin LO-RS	300.5ms	1080ms	170ms	4ms	1527.5ms
P3P LO-RS R6P-2lin LO-RS	300.5ms	1080ms	260ms	5.4ms	1645.9ms
R6P-1lin	1400ms	800ms			2200ms
R6P-1lin LO-RS	1400	800ms	170ms	4ms	2374ms

TABLE III: Average timings for different methods assuming 1000 rounds of RANSAC and 1000 correspondences per image. The amount of time spent by verification already includes the average number of hypotheses returned by the solvers.

time, verification and local optimization will require different amount of time depending on the number of correspondences. The timings we show for verification and local optimization are for 1000 3D-2D correspondences which we chose as a reasonable representative case.

The timings in table II show that, while P3P is very fast compared to R6P, the verification is actually much more expensive than running P3P. The local optimimzation step is even more expensive, an order of magnitude slower than running R6P-2lin. Depending on the application, these numbers will add to the total computation time which will depend on the algorithms used, number of RANSAC iterations, number of correspondences and number of local optimization steps. To give a better intuition about the total time complexity we provide a table III of run-times of the different methods from section VII-C based on the assumption that there are 1000 correspondences in the image, 1000 RANSAC steps and that local optimization is used 10 times during the LO-RANSAC procedure.

G. Augmented reality

We used the Aruco markers in a regular grid to provide an environment with known 3D-2D correspondences. A camera was used to take video of the scene, with random movement, simulating a person looking around. In total around 300 markers were present in the scene with approximately 120 markers detected in each image on average. On each frame we ran a 20 rounds of RANSAC ensuring at least 30Hz for the absolute pose estimation by R6P-1lin and providing robustness to approximately 20% outlier contamination and we calculated the reprojection errors for the detected markers.

As soon as the camera started moving, the estimate provided by P3P started to be visually inaccurate. R6P provided a much more stable reprojection of the virtual objects. The farther the virtual object was from the detected markers from which the

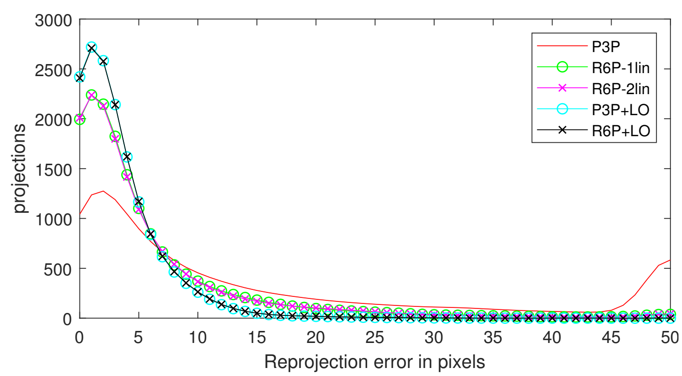


Fig. 10: Reprojection errors on the detected aruco markers using the camera poses obtained by P3P, R6P-lin2, R6P-lin1 and P3P with subsequent local optimization (Bundle adjustment).

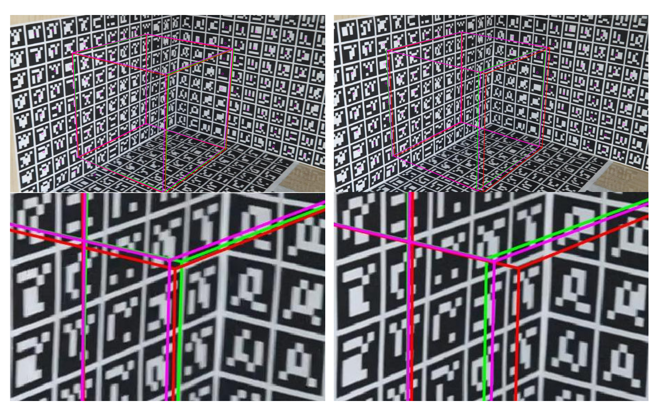


Fig. 11: Placing virtual objects in the scene using absolute camera pose calculated from P3P (red), R6P (green) or P3P with subsequent local optimization (magenta). Two subsequent frames from a video sequence are dislplayed, showing the effect that RS camera motion has on the classic P3P algorithm. In the closeups you can see how the cube projected using P3P deviates (right) from its original pose (left) when the motion starts.

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.
Copyright (c) 2019 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE Transactions on Pattern Analysis and Machine Intelligence
The final version of record is available at http://dx.doi.org/10.1109/TPAMI.2019.2894395

pose was computed, the more significant was the error of P3P. For an example of this effect, see figure 11.

To express this quantitatively. We calculated the reprojection error on the detected markers in each image. The results in figure 10 show that the pose computed by R6P provides overall much smaller reprojection errors. R6P provides almost as good performance as a P3P with subsequent local optimization.

VIII. Conclusion

In this paper, we addressed the problem of absolute pose for cameras with rolling shutter. We presented several different models which capture camera translation and rotation during frame capture. Two of them were found to be feasible to be used in an efficient polynomial solver. Using these solvers, camera position, orientation, translational velocity and angular velocity can be computed using six 2D-to-3D correspondences. The R6P-1lin solver, based on Cayley parametereization of the camera orientation is a first self-sufficient minimal solution to the rolling shutter camera absolute pose problem. The R6P-2lin solver is faster, but uses a linear approximation to the camera orientation and therefore requires an initial guess of the camera orientation. We showed on synthetic as well as real datasets that standard P3P algorithm is able to provide this initialization. Further, having an initial guess on camera orientation, such as from an inertial measurement unit present in cellphones or UAV's, one could use the faster R6P-2lin solver directly. Both of the presented solvers improve on the precision of the camera absolute pose estimate when rolling shutter effect is present in the images, delivering average camera orientation error under half a degree (compared to six degrees for P3P) and relative camera position error under 2% (compared to 6% for P3P) even for large rolling shutter distortions in our synthetic experiments. The solvers were verified to work on real data, delivering increased number of inliers when using R6P over P3P in RANSAC. We evaluated the effects of non-linear refinement with both linearized and non-linearized rolling shutter rotation models and have shown that R6P provides higher number of inliers than P3P with subsequent non-linear refinement in most cases.

Acknowledgment

This research was supported by the European Regional Development Fund under the project IMPACT (reg. no. CZ.02.1.01/0.0/0.0/15_003/0000468), EC H2020-ICT-731970 LADIO project and ESI Fund, OP RDE programme under the project International Mobility of Researchers MSCA-IF at CTU No. CZ.02.2.69/0.0/0.0/17_050/0008025.

References

[1] S. Agarwal, K. Mierle et. al. Ceres Solver http://ceres-solver.org
[2] O. Ait-aider, N. Andreff, J. M. Lavest, U. Blaise, P. C. Ferr, and L. U. Cnrs. Simultaneous object pose and velocity computation using a single view from a rolling shutter camera. In *In Proc. European Conference on Computer Vision*, pages 56–68, 2006.
[3] C. Albl, Z. Kukelova, and T. Pajdla. R6P - Rolling Shutter Camera Absolute Pose. In *CVPR*, pages 2292–2300, 2015.
[4] M. André Ameller, B. Triggs, and L. Quan. Camera pose revisited: New linear algorithms. *In 14eme Congres Francophone de Reconnaissance des Formes et Intelligence Artificielle. Paper in French*, page 2002, 2002.
[5] M. Bujnak, Z. Kukelova, T. Pajdla. Making minimal solvers fast *In 2012 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR'12, pages 1506-1513, 2012.
[6] O. Chum, J. Matas, and J. Kittler. *Locally Optimized RANSAC*, pages 236–243. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.
[7] D. Cox, J. Little, and D. O'Shea. *Using Algebraic Geometry*. Graduate Texts in Mathematics. Springer, 2005.
[8] A.M. Danilevskii. The numerical solution of the secular equation (Russian). *Matem. Sbornik*, 44(2), 1937.
[9] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.
[10] R. Haralick, D. Lee, K. Ottenburg, and M. Nolle. Analysis and solutions of the three point perspective pose estimation problem. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91., IEEE Computer Society Conference on*, pages 592–598, Jun 1991.
[11] M. Hazewinkel. *Encyclopaedia of mathematics*. Springer-Verlag, Berlin New York, 2002.
[12] J. Hedborg, P.-E. Forssén, M. Felsberg, and E. Ringaby. Rolling shutter bundle adjustment. In *CVPR*, pages 1434–1441, 2012.
[13] J. Hedborg, E. Ringaby, P.-E. Forssen, and M. Felsberg. Structure and motion estimation from rolling shutter video. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 17–23, 2011.
[14] D. Henrion, J.-B. Lasserre, and J. Lofberg. Gloptipoly 3: Moments, optimization and semidefinite programming. *Optimization Methods Software*, 24(4-5):761–779, Aug. 2009.
[15] J.A. Hesch, S.I. Roumeliotis. A Direct Least-Squares (DLS) method for PnP. In *IEEE International Conference on Computer Vision*, ICCV'11, pages 383–390, Barcelona, Spain, 2011.
[16] D. Hook, P. McAree. Using Sturm Sequences To Bracket Real Roots of Polynomial Equations. *Graphic Gems I*, Academic Press, 416-423, 1990.
[17] C. Jia and B. L. Evans. Probabilistic 3-d motion estimation for rolling shutter video rectification from visual and inertial measurements. In *MMSP*, pages 203–208. IEEE, 2012.
[18] G. Klein and D. Murray. Parallel tracking and mapping on a camera phone. In *Proceedings of the 2009 8th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '09, pages 83–86, Washington, DC, USA, 2009. IEEE Computer Society.
[19] Z. Kukelova, M. Bujnak, J.Heller, and T. Pajdla. Singly-bordered block-diagonal form for minimal problem solvers. In *ACCV'14*, 2014.
[20] Z. Kukelova, M. Bujnak, and T. Pajdla. Automatic generator of minimal problem solvers. In D. A. Forsyth, P. H. S. Torr, and A. Zisserman, editors, *Computer Vision - ECCV 2008, 10th European Conference on Computer Vision, Proceedings, Part III*, volume 5304 of *Lecture Notes in Computer Science*, pages 302–315, Berlin, Germany, October 2008. Springer.
[21] V. Larsson, K. Åström and M. Oskarsson Efficient solvers for minimal problems by syzygy-based reduction In *CVPR'17*, 2017.
[22] V. Lepetit, F. Moreno-Noguer, and P. Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal of Computer Vision*, 81(2):155–166, 2009.
[23] L. Magerand, A. Bartoli, O. Ait-Aider, and D. Pizarro. Global optimization of object pose and motion from a single rolling shutter image with automatic 2d-3d matching. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part I*, ECCV'12, pages 456–469, Berlin, Heidelberg, 2012. Springer-Verlag.
[24] M. Meingast, C. Geyer, and S. Sastry. Geometric Models of Rolling-Shutter Cameras. *Computing Research Repository*, abs/cs/050, 2005.
[25] G. Nakano Globally Optimal DLS Method for PnP Problem with Cayley parameterization. In *Proceedings of the British Machine Vision Conference* , BMVC'15, pages 78.1–78.11 2015. BMVA Press.
[26] L. Quan and Z. Lan. Linear n-point camera pose determination. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(8):774–780, Aug 1999.
[27] G. Reid, J. Tang, and L. Zhi. A complete symbolic-numeric linear method for camera pose determination. In *Proceedings of the 2003 International Symposium on Symbolic and Algebraic Computation*, ISSAC '03, pages 215–223, New York, NY, USA, 2003. ACM.
[28] E. Ringaby and P.-E. Forssén. Efficient video rectification and stabilisation for cell-phones. *International Journal of Computer Vision*, 96(3):335–352, 2012.
[29] O. Saurer, K. Koser, J.-Y. Bouguet, and M. Pollefeys. Rolling shutter stereo. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 465–472, Dec 2013.
[30] K. Shoemake. Animating rotation with quaternion curves. *SIGGRAPH Comput. Graph.*, 19(3):245–254, July 1985.
[31] J. L. Schönberger and J.-M. Frahm Structure-from-Motion Revisited In

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.
Copyright (c) 2019 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2894395, IEEE Transactions on Pattern Analysis and Machine Intelligence
The final version of record is available at http://dx.doi.org/10.1109/TPAMI.2019.2894395

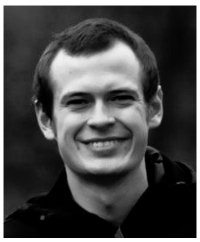
CVPR'16, 2016.

[32] N. Snavely, S. M. Seitz, and R. Szeliski. Modeling the world from internet photo collections. *Int. J. Comput. Vision*, 80(2):189–210, Nov. 2008.

[33] B. Triggs. Camera pose and calibration from 4 or 5 known 3d points. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 278–284 vol.1, 1999.

[34] J. Ventura, C. Arth, and V. Lepetit. An efficient minimal solution for multi-camera motion. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[35] Y. Wu and Z. Hu. Pnp problem revisited. *Journal of Mathematical Imaging and Vision*, 24(1):131–141, 2006.



Cenek Albl received his master's degree in robotics from the Czech Technical University and degree in space science and technology from Luleå University of Technology in 2011. He is currently pursuing his PhD degree in computer vision at Czech Technical University, Department of Cybernetics. He is focusing on Structure from Motion, camera calibration and rolling shutter cameras.

Zuzana Kukelova received the masters degree in mathematics from the Faculty of Mathematics, Physics and Informatics, Comenius University, Bratislava, and her PhD degree at the Department of Cybernetics at the Czech Technical University in Prague. Recently she finished her postdoc at Microsoft Research Cambridge. She is interested in algebraic geometry and computer vision, where she focuses on minimal problems and methods for solving systems of polynomial equations.

Viktor Larsson received his master's degree in computer science in 2013 from Lund University in Sweden. He is currently pursuing his PhD degree in computer vision at the Centre for Mathematical Sciences at Lund University. His research is focused on geometric problems in computer vision and using mathematical tools for solving these.

Tomas Pajdla received the MSc and PhD degrees from the Czech Technical University, in Prague. He works in geometry and algebra of computer vision and robotics, with emphasis on nonclassical cameras, 3D reconstruction, and industrial vision. He contributed to introducing epipolar geometry of panoramic cameras, noncentral camera models generated by linear mappings, generalized epipolar geometries, and to developing solvers for minimal problems in structure from motion. He is a member of the IEEE.

0162-8828 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: ETH BIBLIOTHEK ZURICH. Downloaded on March 03,2020 at 13:47:56 UTC from IEEE Xplore. Restrictions apply.
Copyright (c) 2019 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.