Motion and velocity estimation of rolling shutter cameras

Oliver Grau and Julien Pansiot

December 10, 2012

Abstract

Most modern camera designs based on CMOS sensors do not have a global, but a rolling shutter that exposes the lines of the sensor at different times. This leads to distortions under camera or object motion and affects computer vision techniques. This article introduces the use of a linear velocity model for camera motion and estimates these additional parameters in a bundle adjustment. In experiments on synthetic and real image sequences we demonstrate how effects caused by a rolling shutter video camera can be compensated by the velocity estimation.

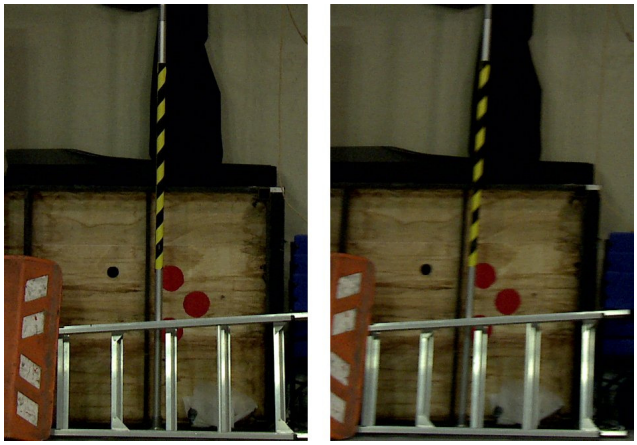

Figure 1: Two image details from the same scene. Left: static camera. Right: camera panning towards the right. The right image appears slanted because its lower lines were acquired after its upper lines. The camera has moved rightwards meanwhile, thus the lower lines seem shifted to the left.

1 Introduction

Estimation of camera movement has many applications in media production, for example in special effects to synchronise virtual with real imagery. Recent camera designs predominately use sensors based on CMOS technology. These sensors usually do not provide a global shutter as found in CCD sensors. Instead each line is exposed at a slightly different time, usually starting from the top each following line is exposed a time τ later than the previous line. The effect can be seen in fig. 1: The scene on the left is captured with a static camera. On the right the same scene is captured under a right panning camera move. As can be seen the scene is starting to show a 'tilting' distortion that gets more pronounced with increased panning speed.

Camera motion estimation is a widely studied technique in computer vision (see for example [3]). However most techniques developed assume a global shutter that captures all image pixels at the same time.

Previous work on estimating motion captured with rolling shutters has been mostly focussed on compensating the effect in 2D [2]. An approach described in [6] assumes known 3D data and approximates the rolling shutter effect over three frames as a 2D velocity vector. The approach in [1] estimates object velocity from a single frame. Hedborg et al. [5, 4] describe a technique to compensate for the rolling shutter effect with a temporal interpolation of the camera model and use it to compute structure and motion. This approach is based on image/point rectification, rather than simultaneous estimation of the camera position and velocity.

Inspired by [1] we estimate both the camera pose plus a first order linear velocity of the camera movement. We demonstrate that the velocity model significantly improves the camera pose estimation under camera movement. The explicit estimation of the velocity also offers new applications. For example, it can be used in a real-time scenario to predict the camera pose at the end of a frame, decreasing the latency by one frame.

At the back-end the estimation of the velocity is demonstrated here in a bundle adjustment (BA) framework. This is a relatively easy to implement extension of existing BA frameworks and any computer vision technique at the front-end can be used unaltered for feature matching.

The remainder of this paper is structured as follows: the next section gives a brief description of rolling shutter cameras. Section 3 describes the use of the velocity model for camera motion estimation. Section 4 presents the results of our experimental validation. The paper finishes with some conclusions.

2 Rolling-shutter cameras

CMOS technology for imaging sensors has some advantages over CCD, including lower power consumption and direct digitalisation. However, the implementation of a global shutter requires a number of additional transistors on the sensor which would reduce the pixel fill-factor and therefore the signal to noise ratio of the sensor. For this reason most CMOS sensor designs

implement a successive line read-out which leads to a rolling shutter.

In camera sensors with a global shutter all pixels are exposed in the same time interval $[t_0, t_1] = [t_0, t_0 + \delta_i]$, with the integration or shutter opening time δ_i. Ignoring the shutter opening time we assume instantaneous capture at time t_0 in the following.

On a rolling shutter pixels are exposed at different times:

$$t_i = t_0 + y\tau \tag{1}$$

t_i is the time the integration starts, t_0 time of first line starts integrating, y is the image line and τ is the time difference between lines. Most sensors read out top-to-bottom. This is the assumed read-out direction in the following. For sensors with bottom-to-top read-out eq. 1 can be formulated accordingly.

The time difference between lines τ is sensor-specific and is assumed to be known in the following estimation method. If this information cannot been found in the specifications from the camera manufacturer, it can be determined by experiment. For this purpose we use a stroboscopic flash with a precisely known flash frequency. Fig. 2 shows an image of a bright surface taken with a Sony PMW-EX3 camera at 25 frames/s at 1080p. The surface is illuminated with a series of flashes at 100 Hz. The dark segment at the top received one flash, the middle segment two flashes and the bottom three, as illustrated in fig. 3.

Figure 2: Image of a stroboscopic flash at 100Hz with a rolling shutter camera.

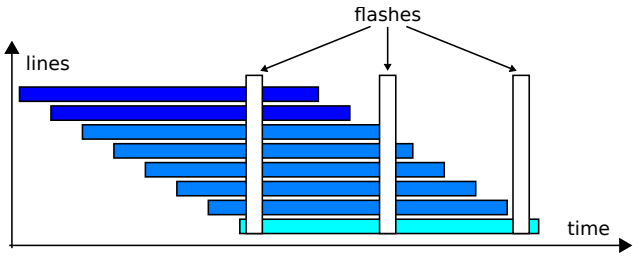

Figure 3: Rolling shutter principle. Every line is exposed at a different time, and in this case receives a different number of flashes.

For the Sony PMW-EX3 camera we can count 685 lines for the middle segment at 100 Hz flash frequency. That is the equivalent of $\tau = 14.6\mu s$. We found that this line offset is constant at all frame-rates supported by this camera type (25,50+60 fps). Therefore τ is not necessarily the inter-frame time divided by the number of lines.

3 Tracking of cameras with rolling shutter

The distortions observed in rolling-shutter cameras are caused by the relative motion of objects with respect to the camera during the capture of an image frame. Although the problem can be equivalently formulated for object movement [1] and camera movement, we assume a static scene and camera movement here.

The camera projection defines a transformation of a point $\mathbf{X}$ defined in the world coordinate system into point $\mathbf{x}$ on the image target.

In homogeneous coordinates with:

$$\mathbf{X} = (X, Y, Z)^T \mapsto \mathbf{X^h} = (X, Y, Z, W)^T$$

and:

$$\mathbf{x} = (x, y)^T \mapsto \mathbf{x^h} = (x, y, w)^T$$

the projection is then defined as:

$$\mathbf{x^h} = P\mathbf{X^h} \tag{2}$$

$$P = K R^{-1} [I_3 | -\mathbf{C}] \tag{3}$$

The vector $\mathbf{C}$ specifies the location of the camera centre point. The rotation matrix R^{-1} describes the rotation of the camera coordinate system against the world coordinate system. The parameters are specified as Euler angles. I_3 is the 3 x 3 identity matrix.

If the camera moves during the integration this will cause distortions that cannot be modelled with the previous camera model. Instead eq. 2 is formulated:

$$\mathbf{x_{t_i}}^{\mathbf{h}} = P(t_i)\mathbf{X_i^h} \tag{4}$$

The time t_i indicates when an image point is captured and can be derived from its line number y using eq. 1.

We substitute camera position $\mathbf{C}$ and orientation $\mathbf{R}^T$ defined in eq. 3 by addition of a linear velocity term:

$$\mathbf{C} = \mathbf{C_0} + t\mathbf{V} \tag{5}$$

$$\mathbf{R}^T = (R_x, R_y, R_z) = (R_{x0} + t\Omega_y, R_{y0} + t\Omega_y, R_{z0} + t\Omega_z) \tag{6}$$

The velocity terms $\mathbf{V}$ and Ω^T in eq. 5 and 6 can be solved for using a bundle adjustment framework [9]. The bundle adjustment is a minimisation framework that solves for the parameters of a camera k described by a number of m parameters $\mathbf{p_k} \in \mathbb{R}^m$. The 3D-2D projection of a scene point $\mathbf{X}$ is defined by these parameters:

$$\mathbf{X} = (X, Y, Z)^T \overset{p_k}{\mapsto} \mathbf{x} = (x, y)^T \tag{7}$$

Eq. 7 uses the camera projection as described by eq. 2 and 3 over a subset of p_k camera parameters with

m variable parameters, i.e. the parameters to solve for[1]. This allows us to solve for external and internal camera parameters and also for the 3D scene points (in a structure-and-motion set-up) as needed. See for example [9] for details on bundle adjustment.

The camera observes a scene or a set of marker points $\mathbf{M_l}$ that are captured in the camera image(s) as point o_l. The scene or marker points can be projected into the image using the camera projection model provided in eq. 7.

That gives two equations per observed point:

$$o_{i_x} - m_{i_x} = 0 \tag{8}$$

$$o_{i_y} - m_{i_y} = 0 \tag{9}$$

In total we get M equations (2 times number of observation points) that are stored in the vector $\mathbf{e}$:

$$\mathbf{e} = \mathbf{o} - \mathbf{m} \tag{10}$$

The parameters to solve for in the bundle adjustment problem can be estimated by a least squares minimisation:

$$\min_{p_k} \sum (\mathbf{o} - \mathbf{m})^2 \tag{11}$$

$$\mathbf{m_i} = f(\mathbf{p_k}, \mathbf{M_i}) \tag{12}$$

The function $f(.)$ is described by the projection of the points M_i into the image. We substitute $f(.) := f(\mathbf{p})$ and use the following approximation:

$$f(\mathbf{p} + \delta_\mathbf{p}) \approx f(\mathbf{p}) + \mathbf{J}(\mathbf{p})\delta_p \tag{13}$$

With the Jacobian $\mathbf{J}(\mathbf{p}) \in \mathbb{R}^{m \times n}$:

$$(\mathbf{J}(\mathbf{p}))_{ij} = \frac{\partial f_i(\mathbf{p})}{\partial p_j} \tag{14}$$

The Jacobian is stored in a $m \times n$ matrix. Standard numerical methods, like Levenberg-Marquart exist to minimise the system expressed as eq. 11. The problem is therefore reduced to finding an analytical solution for the Jacobian $\mathbf{J}(\mathbf{p})$. Solutions for camera pose and structure exist, see for example [8].

To extend the bundle adjustment by the velocity model the camera position and orientation will be substituted by eq. 5 and 6. This can be solved with a symbolic mathematical package or by applying the chain rule.

4 Results

This section presents the results of some experiments to evaluate the performance of the velocity estimation described in the the previous section. We performed an experiment with synthetic and real data to demonstrate that the co-estimation of velocity increases the quality of the tracked camera parameters.

[1]Other parameters are kept constant.

4.1 Synthetic data

In this first experiment we generated a synthetic data set. We used 100 3D points with known position on a 1×1 metre planar grid with 2 metre distance to the camera. This constellation would be similar to a camera calibration set-up with a calibration chart.

The 3D scene points were then projected into the image considering the rolling-shutter effect. A sequence was generated by rotating the camera around its optical axis (camera roll). The roll angle was varied between -120 and +120 degrees with a sinusoidal function. Fig. 4 shows a synthesised image of the projected 100 grid points simulating the effect of the rolling shutter in the middle of the sequence (90 degrees rotation), which shows the most distortions due to the rolling shutter.

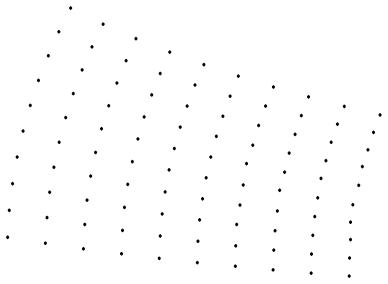

Figure 4: Rolling shutter simulation: reprojection of an orthogonal grid with roll motion. The distortion is clearly visible.

We then estimated the full camera pose parameters (camera position, pan, tilt and roll) with the bundle adjustment approach described in section 3. The estimation was repeated without estimation of the velocity parameters and with velocity estimation.

Fig. 5 shows the estimated roll parameter and the known camera roll. The calibration using velocity estimation is very close to ground truth (better than 0.6 degrees). The estimation without considering velocity shows an offset. This offset is caused by the velocity estimation referring to the start time of the frame. The estimation without velocity is dragged to the middle of the frame causing the offset, but otherwise looks reasonably accurate.

Fig. 6 and 7 show the results of the estimation of pan and tilt and the camera position, with and without velocity respectively. The calibration with velocity produces only small errors, while the results without velocity show significant errors due to the rolling shutter effect. The errors in pan and tilt rotations are in the order of 17 degrees. For the position, an erroneous motion in the order of up to 60 cm is estimated (compared to camera-point distance of 2 meters), although the synthesised camera is not moving.

4.2 Video sequence

The system was also tested against a set of real video sequences. The sequences were recorded using

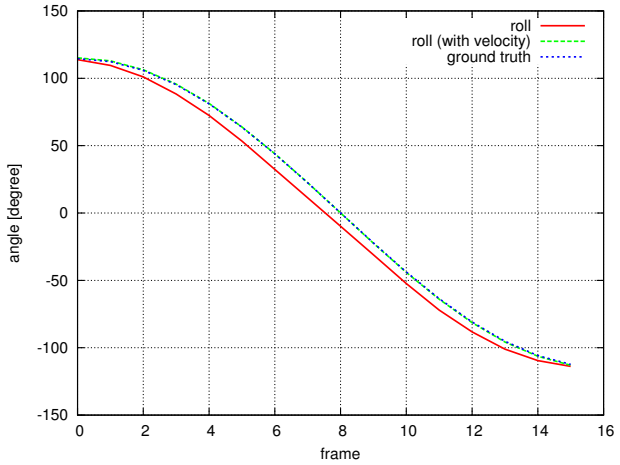

Figure 5: Roll angle with and without velocity and ground truth with synthetic data.

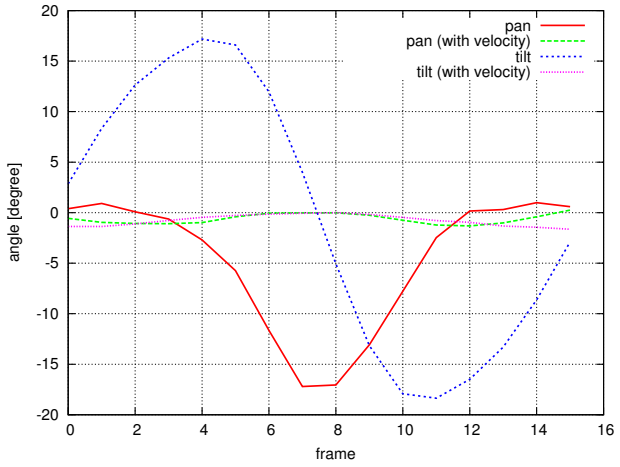

Figure 6: Pan and tilt angle estimation with synthetic data. Ground truth is 0 degrees for both pan and tilt.

a Sony PMW-EX3 video camera mounted on a camera pedestal. The pedestal is heavy enough and is fitted with bearings to ensure relatively smooth motion. The camera was panned manually during the recording of short sequences at different velocities. The distortion effects caused by the rolling shutter are illustrated in fig. 1.

SIFT features [7] were then extracted from the sequences and used as an input to the calibration. Since the camera was rotating around an axis near its nodal point, insufficient displacement occured to provide the parallax required for a Structure-from-Motion (SfM) type of algorithm. Therefore, the features detected on the first frame were artificially clamped onto a plane at a distance equal to the average camera-object distance in the actual scene.

In this experiment, only the pan of the camera varied over time. Nevertheless, the calibration was configured to estimate all 6 positional and rotational DOF as well as, optionally, the pan angular velocity to show how the bundle adjustment does cope with rolling shutter distortion.

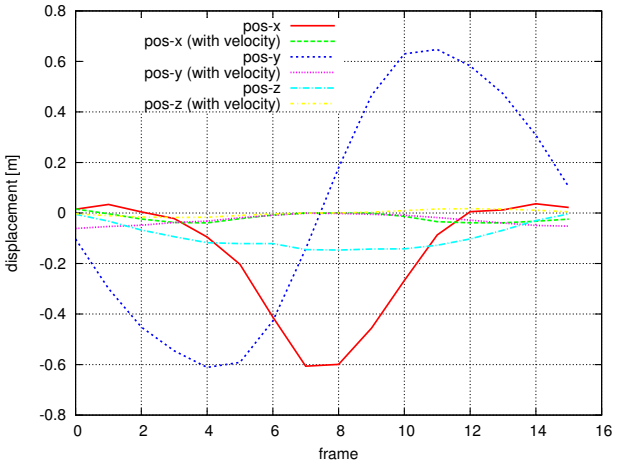

Figure 7: Position of camera estimated with synthetic data. Ground truth is 0 metres in all 3 directions.

The pan angle estimated with and without velocity is illustrated in fig. 8. The tilt and roll angles are illustrated in fig. 9. Both of them are closer to their expected value of zero with velocity estimation.

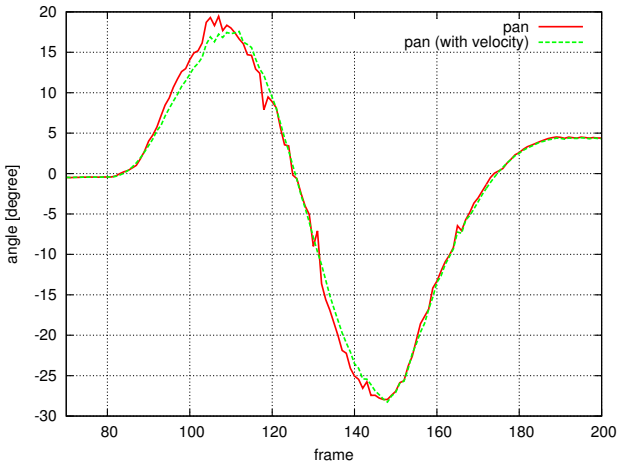

Figure 8: Pan angle estimated with and without velocity. The estimation is smoother with velocity.

The estimated position is illustrated in fig. 10 and 11. The displacement on the vertical axis (Y) should be minimal, since the camera is moving in the horizontal plane. It can been seen in fig. 10 that the consideration of velocity improves its estimation. Similarly, the camera optical axis (Z) should not be subject to much motion given the rotation angles. More significant residual motion is visible on the side axis (X). Part of it can be accounted for as non-nodal rotation making the actual camera nodal point shift sideways with panning rotation.

Furthermore, obvious similarities can be observed between the angles and the position when the rolling shutter is not accounted for (i.e. without velocity estimation). In particular, it is striking that the tilt angle (fig. 9) is negatively correlated with the Y position (fig. 10). This is because the distortion introduced by

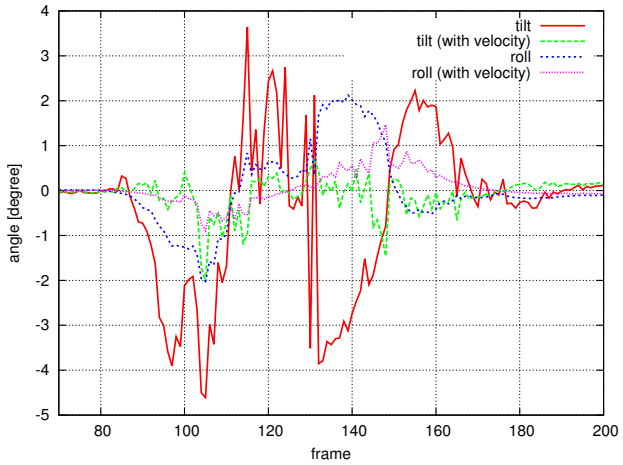

Figure 9: Tilt and roll angles estimated with and without velocity. Angles expected to be near 0 degrees.

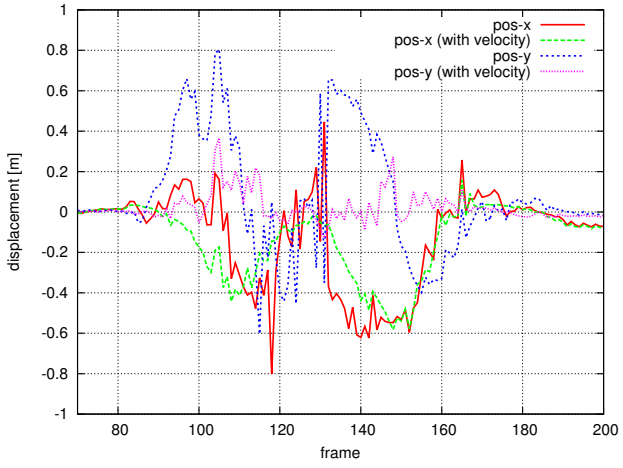

Figure 10: X and Y displacement estimated with and without velocity. Some variation around 0 is expected, as the camera was not rotated around its nodal point.

the rolling shutter effect can be somewhat similar to a perspective effect, as illustrated in fig. 4. Therefore, the bundle adjustment algorithm tends to minimise the residual error by simultaneously rotating and counter-shifting the camera to keep the look-at point constant. In this way, a perspective effect is created that looks like the rolling shutter distortion.

A similar effect is visible with the pan angle (fig. 8) and the X position (fig. 10). The effect is less pronounced since the pan is the actual angle of motion. Nevertheless, the peaks/troughs around frames 118 and 165 in both figures are clearly visible.

In either case, this side effect has been greatly reduced by the introduction of velocity estimation.

4.3 Computational cost

The velocity estimation comes at a computational cost. In the general case, the Bundle Adjustment complexity will be $O(n^3)$ with the number of parameters. In practice, whilst the original estimate ran in 333ms average (including I/O operations), the introduction of the velocity model increased this time to 870ms.

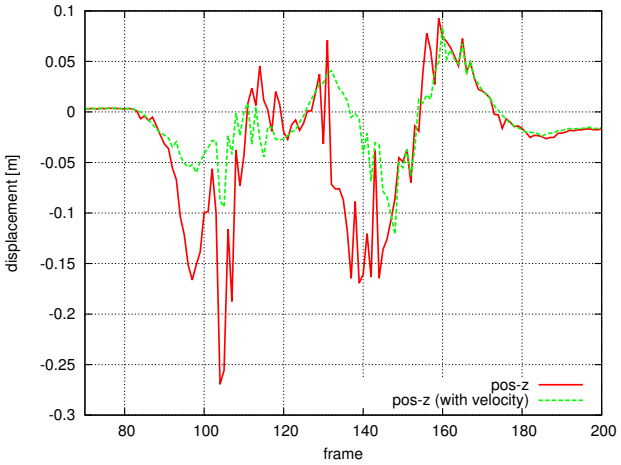

Figure 11: Z displacement estimated with and without velocity. Displacement expected to be close to 0 meters.

5 Conclusions

This contribution presented a framework to estimate the pose and velocity of a rolling-shutter camera. Experimental results have demonstrated that this gives far better estimates of the pose parameters than without velocity. The translational and angular velocity compensate very well for the effects caused by the rolling shutter. Our approach for extending a bundle adjustment framework is relatively easy to implement.

The experimental validation was focussing on camera rotation. The velocity estimation can be less stable when attempting to fit more DOF with features poorly distributed in the image space. Future work will look into translational and other motion classes. So far we are quite confident that the linear velocity model as introduced is sufficient to model the effects of the rolling shutter for most 'practical' cases of camera motion. However, it would be also be of interest to investigate higher order velocity models.

References

[1] O. Ait-Aider, N. Andreff, J. Lavest, and P. Martinet. Simultaneous object pose and velocity computation using a single view from a rolling shutter camera. *Computer Vision–ECCV 2006*, pages 56–68, 2006.

[2] J. Chun, H. Jung, and C. Kyung. Suppressing rolling-shutter distortion of cmos image sensors by motion vector detection. *Consumer Electronics, IEEE Transactions on*, 54(4):1479–1487, 2008.

[3] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[4] J. Hedborg, P.-E. Forssén, M. Felsberg, and E. Ringaby. Rolling Shutter Bundle Adjustment. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[5] J. Hedborg, E. Ringaby, P.-E. Forssén, and M. Felsberg. Structure and Motion Estimation from Rolling Shutter Video. In *IEEE International Conference onComputer Vision Workshops (ICCV Workshops), 2011*, pages 17–23. IEEE Xplore, 2011.

[6] G. Klein and D. Murray. Parallel tracking and mapping on a camera phone. In *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, pages 83–86. Ieee, 2009.

[7] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, Nov. 2004.

[8] N. Snavely and et al. Bundler: Structure from motion (sfm) for unordered image collections. http://phototour.cs.washington.edu/bundler/.

[9] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment - a modern synthesis. *Vision algorithms: theory and practice*, pages 153–177, 2000.