Visualizing Spectral Bundle Adjustment Uncertainty

Kyle Wilson
Washington College
Chestertown, MD
kwilson24@washcoll.edu

Scott Wehrwein
Western Washington University
Bellingham, WA
scott.wehrwein@wwu.edu

Abstract

Bundle adjustment is the gold standard for refining solutions to geometric computer vision problems. This paper develops an uncertainty visualization technique for bundle adjustment solutions to Structure from Motion problems.

Propagating uncertainty through an optimization—from measurement uncertainties to uncertainties in the resulting parameter estimates—is well understood. However, the calculations involved fail numerically for real problems. Often we cope by considering only individual variances, but this ignores the important mutual dependencies between parameters. The dominant modes of uncertainty in most models are large motions involving nearly all parameters at once. These frequently look like flexions, stretchings, and bendings in the overall scene structure.

In this paper we present a numerically tractable method for computing dominant eigenvectors of the covariance of a Bundle Adjustment solution. We pay careful attention to the mismatched scales of rotational and translational parameters. Finally, we animate this spectral information. The resulting interactive visualizations (included in the supplemental) give insight into the quality and failure modes of a model. We hope that this work is a step towards broader uncertainty-aware computation for Structure from Motion.

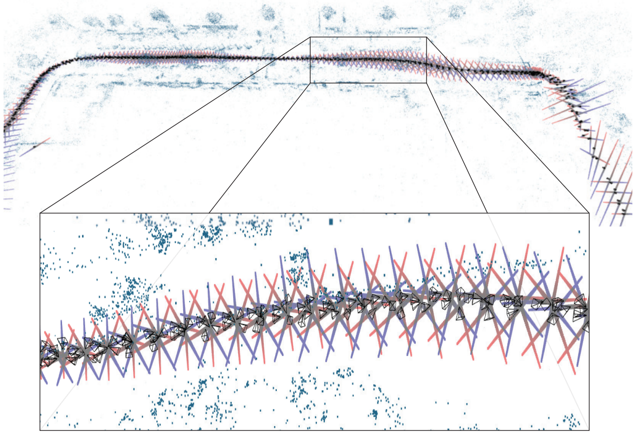


Figure 1: Two views of a large mode of uncertainty in the LADYBUG dataset. Each blue-red line shows the motion of a single camera in this mode. Notice the structure indicated by parallel lines: each set of parallel motions are cameras that are tightly coupled together. The structure of the model is revealed to be five superimposed chains of cameras. The macro-scale motions of this mode are bendings, both at the ends and in the middle of the problem. Although the corners and centerpoint appear less uncertain in this mode, the center has large motions in other nearby modes.

1. Introduction

The final step of many geometric computer vision problems is bundle adjustment: a non-linear refinement of all the model parameters. Structure from Motion problems are one of the most studied specializations of bundle adjustment. Here the goal is to jointly infer the parameters of cameras and of 3D points that the cameras observe. The theoretical machinery to propagate uncertainty though the bundle adjustment problem has been available for some time [11], but this uncertainty information has not found widespread use. In practice, the computations are numerically fraught and often expensive. Also, using the resulting covariance correctly can be nuanced [20].

This paper develops a method to visualize bundle adjustment covariance for Structure from Motion (SfM) problems. We extract and visualize the dominant modes (eigenvectors) of uncertainty in a model, such as the one shown in Figure 1. In contrast to individual parameter variances, which treat each parameter as if the others were fixed, these modes show the collective motions of highest uncertainty. These modes often have narrative interpretations that can give us useful insight into the properties and challenges of a scene. For instance, in Figure 1, we see that the scene is structured not as a single cluster, or even a single chain, but as a loose superposition of five chain-like submodels.

A quick inspection can show us clusters of cameras that are very well localized with respect to each other, but also corners and joints where large clusters are linked uncertainly. We may discover that the relative scale of large sections is poorly known, or that there is very little coupling between nighttime and daytime images. Our visualizations reveal the macro-scale model structure that dominates the overall uncertainty. We distinguish our global (whole-model) uncertainty information from local uncertainty information for small blocks of parameters. Such restricted calculations do not capture the mutual dependencies and interactions across parameter blocks on the model.

The core idea behind the visualization is extremely simple: each eigenvector is a tangent space perturbation to the model. We modulate the perturbation sinusoidally and render the resulting time-varying model in 3D. We encourage the reader to look at the supplemental (or look at the code we will release) for interactive WebGL models which are much more intuitive than static 2D renderings.

There are non-trivial numerical obstacles to computing these eigenvectors. To stay numerically stable we avoid computing any matrix inverses. Instead, we render inverses unnecessary by directly extracting the spectral information. We also must address issues of gauge and norm-modeling. Because the matrices in question are quite stiff, poor design choices lead to wildly inaccurate solutions.

Our contributions are: (1) a method for extracting the dominant modes of uncertainty of an SfM problem, and (2) a simple and effective scheme to visualize this uncertainty information. We end with a hope for further applications of spectral uncertainty information in building uncertainty-aware SfM systems. Our code and visualizations are available at https://wilsonkl.github.io/sfmflex-release.

2. Related Work

Triggs *et al.* [20] give a classic overview of the bundle adjustment problem and its applications in computer vision. They conclude with a detailed discussion of *gauge freedom*, where solutions are only determined up to a choice of coordinate systems. Covariances may look very different under different gauges, making it difficult to directly compare uncertainties of different problems.

Kanatani and Morris [12] and Absil *et al.* [1] developed Riemannian geometry theories of uncertainty propagation on manifold-valued optimization problems. At the cost of introducing some significant mathematical machinery, these approaches elegantly derive covariances of bundle adjustment problems. The standard nonlinear least squares solver package, Ceres-Solver [3], now incorporates covariance estimation, although it does not support large problems that have gauges (such as the problems we are interested in).

Several Structure from Motion systems incorporate uncertainty in their decision making. Agarwal *et al.* [2] use the traces of covariances of two-view models as a selection criterion while building their scene graphs. In the context of incremental SfM, Haner and Heyden [9] use camera parameter uncertainty for next-best-view selection. Schonberger and Frahm [18] use approximations to the covariance information to scale to larger problems.

Polic and Pajdla [17] give a method to explicitly compute entire SfM covariance matrices. After reducing the problem to only camera parameters with a Schur complement, they perform a matrix pseudo-inversion through a Taylor expansion. In a followup, Polic *et al.* [16] compute the covariances faster and more accurately by using an explicit nullspace description and a matrix block-inversion. In contrast we avoid the numerical dangers of explicit pseudo-inverses by computing eigenvectors directly from the inverse covariance.

In this paper our focus is on applying spectral uncertainty information to a visualization task.

3. Method

This section contains a full discussion of our method for accurately computing and visualizing covariance eigenvectors. We begin with background on covariances of nonlinear least squares problems (Section 3.1). Next we describe our method in several parts: reducing the problem size (Section 3.2), appropriately choosing a norm (Section 3.3), solving for the dominant eigenvectors (Section 3.4), and finally animating the eigenvectors for visualization (Section 3.5). We present a discussion and analysis of our results alongside a gallery of visualizations in Section 4.

3.1. Covariance of Nonlinear Least Squares

Bundle adjustment is an instance of nonlinear least squares. These problems have the form

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \frac{1}{2} \sum_i \|r_i(\mathbf{x})\|^2, \tag{1}$$

where the residual functions $r_i = y_i - f_i(\mathbf{x})$ describe a difference between model and measurement, and the parameter vector $\mathbf{x}$ is the quantity to be determined. Assuming the noise model $\mathbf{r}(\mathbf{x}) = \mathbf{y} - \mathbf{f}(\mathbf{x}) + \mathcal{N}(0, I)$, the maximum likelihood estimator is in fact the minimizer of Equation (1), and the covariance of the estimate is known to be the inverse of the Hessian matrix: $\mathbf{V} = \mathbf{H}^\dagger$ [11]. ($\mathbf{H}^\dagger$ denotes the Moore-Penrose pseudoinverse.) We leave the problem instance gauge-free, as the choice of a gauge-fixing constraint greatly reshapes the resulting uncertainties.

Measurement Covariance Our $\mathcal{N}(0, I)$ noise model can be improved if we have an estimate Σ_m of the covariance of the measurements (i.e., of the input to our problem). In our updated model the residuals are derived from the measurements as $\mathbf{r}(\mathbf{x}) = \mathbf{y} - \mathbf{f}(\mathbf{x}) + \mathcal{N}(0, \Sigma_m)$. The minimum of Equation (1) is no longer the maximum likelihood estimator, but it is simple to transform the problem back into a nonlinear least squares problem: since covariances are symmetric positive definite, we can write

$$\Sigma_m^{-\frac{1}{2}}\mathbf{r}(\mathbf{x}) = \Sigma_m^{-\frac{1}{2}}\mathbf{y} - \Sigma_m^{-\frac{1}{2}}\mathbf{f}(\mathbf{x}) + \mathcal{N}(0, I)$$

or

$$\tilde{\mathbf{r}}(\mathbf{x}) = \tilde{\mathbf{y}} - \tilde{\mathbf{f}}(\mathbf{x}) + \mathcal{N}(0, I) \tag{2}$$

where $\tilde{\mathbf{r}} = \Sigma_m^{-\frac{1}{2}}\mathbf{r}$, $\tilde{\mathbf{y}} = \Sigma_m^{-\frac{1}{2}}\mathbf{y}$, and $\tilde{\mathbf{f}} = \Sigma_m^{-\frac{1}{2}}\mathbf{f}$. With the residuals normalized in this way, the MLE of our model is once again the nonlinear least squares solution of $\arg\min_{\mathbf{x}} \|\tilde{\mathbf{r}}(\mathbf{x})\|^2$.

Estimating the measurement covariance Σ_m is a poorly posed problem requiring heavy regularization or assumptions. Lhuillier and Perriollat[15] use an i.i.d. assumption: $\Sigma_m = \sigma^2 I$, where $\sigma^2 = \|\mathbf{r}(\mathbf{x})\|/(2N_o - pN_c - 7)$ where N_c is the number of cameras in the scene, p is the dimensionality of the camera model, and N_o is the number of point-camera observations in the scene. Bishop [6] suggests using $\Sigma_m = \mathrm{Diag}(|\mathbf{r}(\mathbf{x})|)$. Kanatani and Morris [12] give a more principled method based on patch windows in the neighbor of image observations, but it is expensive to compute. We use Bishop's method because we consistently observed it leading to problems with better numerical conditioning.

The Gauss-Newton Approximation For problems with many parameters and residuals, the Hessian is intractably expensive to compute. Instead, it is standard to use an approximation. Writing $\mathbf{x} = (x_1, \ldots, x_n)$ and $\tilde{\mathbf{r}}(\mathbf{x}) = (\tilde{r}_1(\mathbf{x}), \ldots, \tilde{r}_m(\mathbf{x}))$, let us expand the Hessian:

$$\begin{aligned}\mathbf{H}_{jk}(\mathbf{x}) &= \frac{\partial}{\partial x_j}\frac{\partial}{\partial x_k}\tilde{\mathbf{r}}(\mathbf{x}) \\ &= \sum_{i=1}^{m}\left(\underbrace{\frac{\partial \tilde{r}_i}{\partial x_k}\frac{\partial \tilde{r}_i}{\partial x_j}}_{\text{First-derivative terms}} + \underbrace{\tilde{r}_i\frac{\partial^2 \tilde{r}_i}{\partial x_j\,\partial x_k}}_{\text{Second-derivative terms}}\right)\end{aligned}$$

The second-derivative terms are expensive to compute. The standard Gauss-Newton approximation is to assume

$$\sum_{i=1}^{m}\tilde{r}_i\frac{\partial^2 \tilde{r}_i}{\partial x_j\,\partial x_k} \approx 0, \quad \forall j, k. \tag{3}$$

This approximation works best when the residual functions r_i have low values at the solution ($\tilde{r}_i \approx 0$) and or when they have low curvature ($\frac{\partial^2 \tilde{r}_i}{\partial x_j\,\partial x_k} \approx 0$).

The Hessian is now quite simple to work with. Writing the Jacobian matrix as $\tilde{\mathbf{J}}_{ij} = \frac{\partial \tilde{r}_i}{\partial x_j}$ the Hessian becomes $\mathbf{H} \approx \tilde{\mathbf{J}}^\top\tilde{\mathbf{J}}$. Recalling that $\tilde{\mathbf{r}} = \Sigma_m^{-\frac{1}{2}}\mathbf{r}$, so that $\tilde{\mathbf{J}} = \Sigma_m^{-\frac{1}{2}}\mathbf{J}$, the covariance of a nonlinear least squares estimate given measurement covariance Σ_m is

$$\mathbf{V} = \mathbf{H}^\dagger \approx \left(\mathbf{J}^\top\Sigma_m^{-1}\mathbf{J}\right)^\dagger. \tag{4}$$

3.2. Reducing the Problem Size

The matrix $\mathbf{J}^\top\Sigma_m^{-1}\mathbf{J}$ is large and perhaps ill-conditioned, which makes it difficult to work with numerically. We use several domain-specific simplifications. First, the parameter vector for Structure from Motion problems contains several different types of variables: camera extrinsics (position, orientation), camera intrinsics (focal length, distortion coefficients), and 3D point positions. Of these, the camera extrinsics are most interesting to visualize. For computational convenience, we drop the intrinsics.

Second, we factor out the 3D points through the Schur complement[1]. Although forming the Schur complement involves a matrix inversion, the matrix in question is block-diagonal with a 3-by-3 block for each scene point, so the inversion is accurate and efficient [3, 20]. This operation reduces the dimensionality of our matrix from $9N_c + 3N_p$ to $6N_c$, where N_c and N_p are the number of cameras and points in the problem. Since N_p is often many orders of magnitude greater than N_c, this is a huge improvement.

Although $\mathbf{J}^\top\Sigma_m^{-1}\mathbf{J}$ is quite sparse, forming the Schur complement causes a lot of fill-in, typically about 40% in our experiments. This is high enough that dense methods now out-perform sparse ones. Additionally, the Schur complement is such a reduction in size that it is far more efficient to form it explicitly, rather than using a matrix-free representation.

3.3. Considerations on Units and Norms

Before any spectral computations, we must address a subtle difficulty: vector norms. First we discuss vectors that $\mathbf{V}$ can operate on.

As second-order differential objects, $\mathbf{J}^\top\Sigma_m^{-1}\mathbf{J}$ and $\mathbf{V}$ are quadratic forms in the tangent space to a solution. That is, for a tangent space motion $\mathbf{v}$, the quadratic form $\mathbf{v}^\top\mathbf{V}\mathbf{v}$ gives the variance in the direction $\mathbf{v}$. We need not restrict ourselves to looking at uncertainty in a single coordinate, which would be $\mathbf{e}_j^\top\mathbf{V}\mathbf{e}_j$.

Directions of maximum uncertainty are particularly interesting:

$$\lambda_{\max} = \max_{\|\mathbf{v}\|=1}\mathbf{v}^\top\mathbf{V}\mathbf{v} \tag{5}$$

[1] The Schur complement trick involves essentially row-reducing a block 2-by-2 matrix to compute only a block of its inverse. In this case, the block structure separates point parameters from camera parameters; see [3, 20] for more details.

However, the Euclidean norm $\|\mathbf{v}\|_2 = \left(\sum_i v_i^2\right)^{\frac{1}{2}}$ is not particularly natural. Notice that some of the v_j are spatial variables and other v_j are rotational. Directly adding these quantities mismatchs the units.

More abstractly, camera extrinsics are elements of $\mathrm{SE}(3)$, the manifold of 3D rigid motions. Perturbations to each camera belong to the associated Lie algebra $\mathfrak{se}(3)$, and our vectors $\mathbf{v}$ are elements of $\mathfrak{se}(3)^{N_c}$. The manifold $\mathrm{SE}(3)$ itself is the direct product of $\mathrm{SO}(3)$ and $\mathbb{R}^3$, the manifolds of 3D rotations and translations. Given norms on their tangent spaces $\mathfrak{so}(3)$ and $\mathbb{R}^3$, we can form a norm on $\mathfrak{se}(3)$ by combining them as follows:

$$\|(\mathtt{R}, \mathbf{t})\|_{\mathfrak{se}(3)} = \left(\alpha_r \|\mathtt{R}\|^p_{\mathfrak{so}(3)} + \alpha_t \|\mathbf{t}\|^p_{\mathbb{R}^3}\right)^{1/p} \tag{6}$$

where α_r and α_t are positive constants and $p \geq 1$. The $p = 2$ case is tremendously computationally convenient: if $\mathtt{R}$ is represented as an axis-angle 3-vector, $\|\mathtt{R}\|_2$ is the natural (geodesic) norm on $\mathrm{SO}(3)$. If we also choose the Euclidean norm on $\mathbb{R}^3$, then up to the weights α_r and α_t, the induced norm on $\mathfrak{se}(3)$ (and hence on $\mathfrak{se}(3)^{N_c}$) is a Euclidean vector norm in the tangent space.

Notice that there are no natural choices of weights α_r and α_t. They quantify a change of units between rotational and Euclidean quantities. Furthermore, because SfM problems are ambiguous up to a similarity, the Euclidean units are arbitrary. We pick weights that represent the scale of each unit, effectively non-dimensionalizing them. Let s_r be the median distance from camera rotations to the average camera rotation [10], and let s_t be the median distance from camera positions to the average camera position. Then defining a scale matrix $\mathbf{S}_1$ for a single camera as $\mathbf{S}_1 = \mathrm{Diagonal}([s_r, s_r, s_r, s_t, s_t, s_t])$, we can write

$$\|(\mathtt{R}, \mathbf{t})\|_{\mathfrak{se}(3)} = \left(\|\frac{\mathtt{R}}{s_r}\|_2^2 + \|\frac{\mathbf{t}}{s_t}\|_2^2\right)^{1/2}$$

as

$$\left\| \begin{bmatrix} \mathtt{R} \\ \mathbf{t} \end{bmatrix} \right\|_{\mathrm{SE}(3)} = \left\| \mathbf{S}_1^{-1} \begin{bmatrix} \mathtt{R} \\ \mathbf{t} \end{bmatrix} \right\|_2 \tag{7}$$

and in the product space $\mathfrak{se}(3)^{N_c}$, the norm is (using Kronecker product notation):

$$\|\mathbf{v}\|_{\mathfrak{se}(3)^{N_c}} = \left\| \left(\mathbf{I}_{N_c} \otimes \mathbf{S}_1^{-1}\right) \mathbf{v} \right\|_2 . \tag{8}$$

Going forward, we will write $\mathbf{S}_{N_c} = \mathbf{I}_{N_c} \otimes \mathbf{S}_1$.

In summary, we emphasize that there is no single natural choice for a norm on $\mathfrak{se}(3)^{N_c}$. We normalize the rotational and translational components of the space by robust estimators of the variance in those parameters to relate incommensurate units. Without this adjustment, scaling the problem instance can greatly affect the resulting covariance eigenvectors without changing the value of the cost function: scaled-down scenes yield variance almost exclusively in rotational directions, while expanded scenes have mass almost exclusively in translational directions.

3.4. Solving for the Eigenvectors

To visualize the dominant modes of uncertainty in an SfM problem we wish to compute some (say K_{eig}) of the dominant eigenvectors of the covariance matrix. These are the directions of largest effect within the covariance matrix.

As described in Section 3.2, we have greatly reduced the problem size by dropping the rows and columns representing camera intrinsics and taking a Schur complement to factor out the 3D point parameters, leaving only camera orientations and positions. The resulting Schur matrix (call it $\mathbf{Z}$) is far smaller, but also mostly dense. Its pseudoinverse (which we do not compute in full) is the pseudoinverse of the camera-parameters block of the covariance matrix.

The largest eigenvectors of $\mathbf{Z}^\dagger$ (that is, the dominant modes of covariance with respect to camera extrinsics) are the smallest non-null eigenvalues of $\mathbf{Z}$. Both $\mathbf{Z}$ and $\mathbf{Z}^\dagger$ share a nullspace corresponding to the seven-dimensional gauge ambiguity of SfM problems: the solution is only determined up to an arbitrary choice of orientation, position, and scale.

Consider this optimization problem to determine the smallest eigenvector (under our chosen norm on $\mathfrak{se}(3)^{N_c}$):

$$\mathbf{v}^* = \underset{\|\mathbf{S}_{N_c}^{-1}\mathbf{v}\|_2 = 1}{\arg\min} \ \mathbf{v}^\top \mathbf{Z} \mathbf{v} \tag{9}$$

With a change of variables, this becomes a Euclidean-norm eigenvalue problem:

$$\begin{aligned} \mathbf{u}^* &= \underset{\|\mathbf{u}\|_2 = 1}{\arg\min} \ \mathbf{u}^\top \mathbf{S}_{N_c} \mathbf{Z} \mathbf{S}_{N_c} \mathbf{u} \\ \mathbf{v}^* &= \mathbf{S}_{N_c} \mathbf{u}^* . \end{aligned} \tag{10}$$

That is, to find the largest eigenvectors of $\mathbf{Z}^\dagger$ under our norm, we solve for the smallest (non-null) eigenvectors of $\mathbf{S}_{N_c}^\top \mathbf{Z} \mathbf{S}_{N_c}$. We solve this eigenproblem using Arpack [14] via Julia under the Hermitian no-shift smallest-magnitude mode. Since we expect a 7D nullspace, we find $K_{\text{eig}} + 7$ eigenvectors and discard the seven smallest ones.

Alternatives. We also experimented with using the LOBPCG [13] solver, but it often failed. This was because while by construction, $\mathbf{S}_{N_c}^\top \mathbf{Z} \mathbf{S}_{N_c}$ is positive symmetric definite, round-off error often causes some of the nullspace eigenvalues to be negative. This causes an internal Cholesky factorization to fail.

We also experimented with Golub deflation [8] to define the nullspace explicitly and then search for eigenvectors orthogonal to it. While elegant, we found this to be numerically problematic. The resulting eigenvectors were very low quality, suggesting that the deflated system was stiff.

3.5. Visualizing the Uncertainty

Given a bundle adjustment solution $\mathcal{X} \in \mathrm{SE}(3)^{N_c}$, we compute eigenvectors $\mathbf{v_1}, \mathbf{v_2}, \ldots \in \mathfrak{se}(3)^{N_c}$ as described. Using the exp map from the tangent space to the manifold[2] we can "vibrate" each mode sinusoidally:

$$\mathcal{X}_k(t) = \mathcal{X} \exp\left[(A \sin \omega t)\mathbf{v_k}\right] \tag{11}$$

This gives periodic time-varying camera configurations for each eigenvector. The amplitude A and frequency ω can be freely adjusted until the visualization is easy to interpret.

We built a browser-based viewer to inspect our animated output. See the supplemental for interactive demos.

4. Results

We begin this section with a gallery and a qualitative discussion of our visualizations. Then we move to quantitative discussions: confirming the correctness of our eigenvectors and examining runtime. We conclude with a discussion of the eigenvalues corresponding to our eigenvectors.

4.1. Eigenvector Visualizations

Figure 2 shows some of the dominant eigenvectors of problems from the Bundle Adjustment in the Large (BAL) dataset [4]. Each scene point is rendered as a light blue dot and camera centers are shown as black frusta. Modes of uncertainty are shown as colored bars centered on each camera. The red-blue color gradients across the bars reveal the phase of oscillation. These static visualizations do not show the rotational components of the uncertainty modes. We encourage the reader to follow along with the interactive animations in the supplemental.

Ladybug dataset The left-hand column of Figure 2 shows the largest uncertainty modes of a problem from the BAL LADYBUG dataset. This dataset was collected by a Ladybug device—several cameras rigidly joined together—driven down a street. Notice that all of the modes exhibit sets of near-parallel lines. Each one of these sets corresponds to one of the Ladybug's five cameras. As a result, from an uncertainty perspective the scene consists of five distinct chains of cameras on top of each other.

All of the biggest modes of this dataset are motions of these chains. The largest mode is of translations of these chains past each other. The second mode is flexing in the tails of the scene. The end furthest from the viewport has very large vertical uncertainty. The third and fourth modes also introduce some vertical uncertainty to the main stretch of the street. Finally, the fifth mode looks like a standing sine wave with nodes at the corners and midpoint.

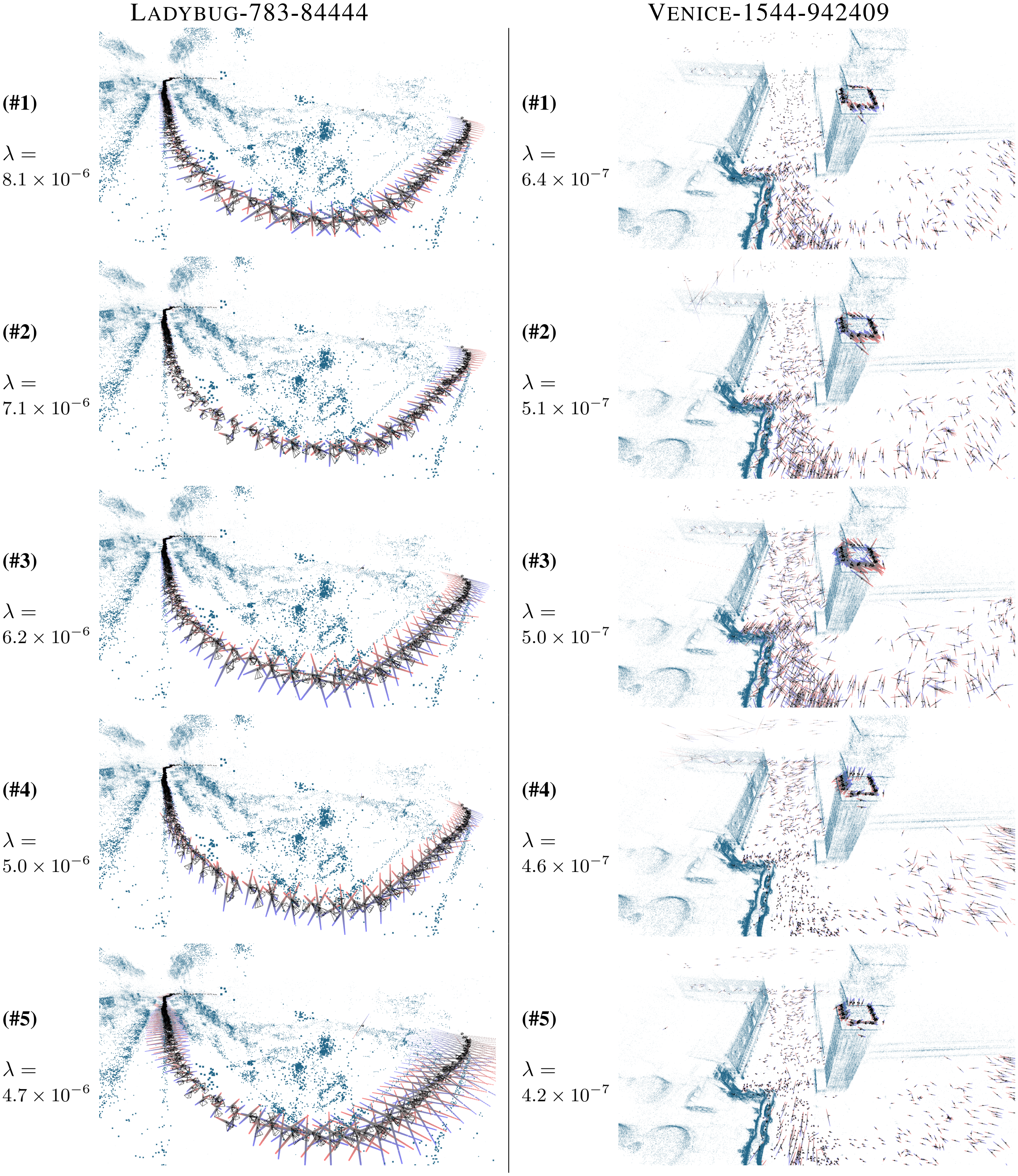
These visualizations show us that what looked like a simple linear scene is actually a bit more topologically complicated. It is composed of five loosely-coupled chains of cameras. The ends of the scene are the least certain (and the far end is very unstable.) The largest uncertainties are in the relative positions of the chains, and vertical positions are better known than horizontal ones.

Speaking practically, these visualizations suggest that including prior knowledge about the camera rig—that cameras are rigidly coupled to each other at each capture point—could sharply reduce uncertainty in the scene.

Venice dataset The right-hand column of Figure 2 shows the largest uncertainty modes on a problem from the BAL VENICE dataset. Once again we can see many interpretable features. The top of the clocktower is particularly uncertain in all of the modes. Looking more closely, the four sides of the tower are fairly rigid, but their relative scales are poorly determined. Notice that each clocktower side has large motions in modes one, two, and three, and that the sides move mostly forward and back along their respective viewing directions. The fourth and fifth modes show bending and stretching on the edges of the model: the right-hand plaza can stretch outwards (this is easier to see in the animated supplemental), while the farthest-back cameras can move from side to side, pivoting around the central tower. If we wanted to improve the model, it seems that there are no quick fixes for the uncertainty in the top of the clock tower, or for the farthest cameras. But uncertainty in the plaza could be quickly reduced if we could find some new views that see both the clock tower and the left-hand building. Likewise, the scale of the right-hand plaza could be better established by right-looking views from below the left-hand building, which are lacking in the dataset.

1DSfM datasets Figure 3 shows two uncertainty modes on two problems from the 1DSfM datasets [21]. These datasets were collected in-the-wild from Flickr keyword searches. These datasets lack the rigid structure of LADYBUG, but we can still learn about the topology of the problem. The largest modes of uncertainty in YORKMINSTER focus on elevated viewpoints, which are a minority in the scene. These aerial cameras are less well connected. The main modes of uncertainty are an "unrolling" motion around the central spire of the cathedral and a vertical translation of the top-most cameras, so some scene properties, such as the height of the tower, are particularly untrustworthy. In TOWER OF LONDON, what looks like a homogenous group of cameras actually behaves more like two groups. The largest modes of uncertainty are stretches (see Figure 3) and vertical shifts (not shown). In the stretches, the two sets of cameras move in diverging directions.

[2] For components in $\mathbb{R}^3$ the exp map is just Euclidean addition, and for components in $SO(3)$ it is given by the Rodriguez rotation formula.


LADYBUG-783-84444

VENICE-1544-942409

(#1)
$\lambda = 8.1 \times 10^{-6}$
(#2)
$\lambda = 7.1 \times 10^{-6}$
(#3)
$\lambda = 6.2 \times 10^{-6}$
(#4)
$\lambda = 5.0 \times 10^{-6}$
(#5)
$\lambda = 4.7 \times 10^{-6}$
(#1)
$\lambda = 6.4 \times 10^{-7}$
(#2)
$\lambda = 5.1 \times 10^{-7}$
(#3)
$\lambda = 5.0 \times 10^{-7}$
(#4)
$\lambda = 4.6 \times 10^{-7}$
(#5)
$\lambda = 4.2 \times 10^{-7}$

Figure 2: Visualizations of the five most dominant eigenvectors of two BAL problems. The blue dots are the scene structure portion of the model. These are motionless because we have restricted our eigenvector computations to camera extrinsic parameters. Each camera's motion is shown by a bar through the camera center. The color gradient on each bar shows the phase of oscillation. See the supplemental for WebGL animations which are more intuitive than static images.

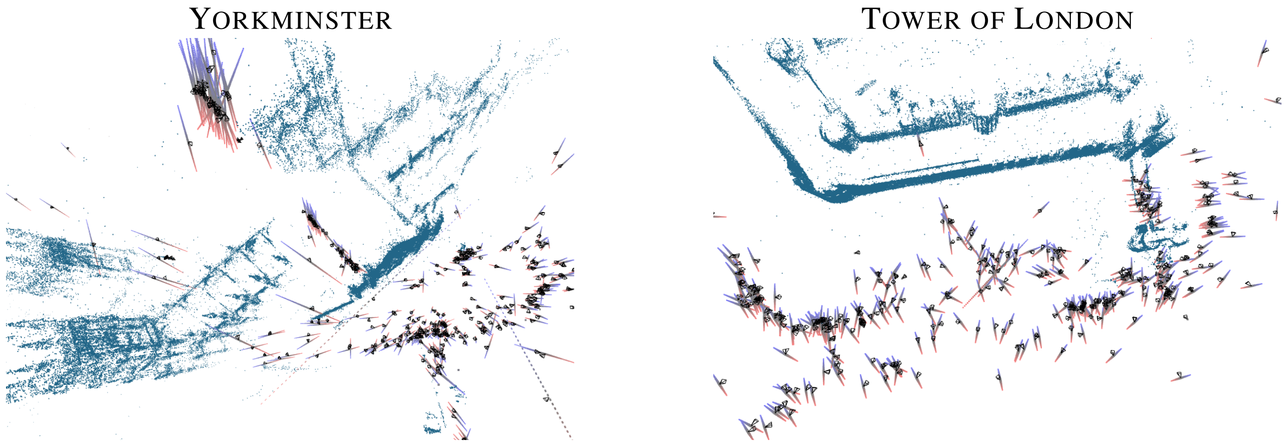


Figure 3: Visualizations of two dominant eigenvectors of 1DSfM scenes. The YORKMINSTER model is bending and unfurling around the central spire. In TOWER OF LONDON we see one group of cameras shifting forward and to the right, and an overlapping group shifting forwards and to the left.

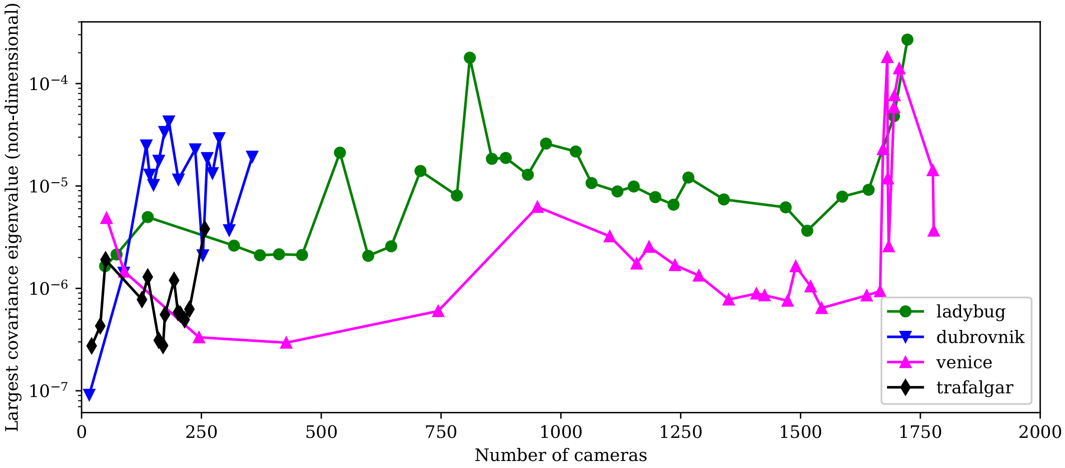


Figure 4: The largest eigenvalue (in nondimensionalized coordinates) of each of the BAL problems. Notice that largest eigenvalues are brittle: adding a few poorly-constrained cameras can create a new mode of large uncertainty.

4.2. Correctness

Although the symmetric eigenvalue problem has a relative condition number of one [19], finding the corresponding eigenvectors is not as well posed. In fact, on some BAL problems we have observed $\lambda_{\max}/\lambda_{\min}$ as large as $1e30$ (after accounting for gauge). While the conditioning of the eigenvector problem is not determined by this quantity, it is a cause for concern.

Empirically, the "eigenvectors" that we compute on larger problems lack properties that we expect eigenvectors to possess. Generally, a set of eigenpairs $\mathbf{v}_k, \lambda_k$ for a matrix $\mathbf{A}$ should satisfy at least three properties: (1) the set of eigenvectors is orthonormal, (2) the eigenvectors have small residuals ($\|\mathbf{A}\mathbf{v}_k - \lambda_k \mathbf{v}_k\| \approx 0$), and (3) $\mathbf{A}\mathbf{v}_k$ should be parallel to $\mathbf{v}_k$. ARPACK exactly enforces the first property, but the other two only hold on the smallest BAL problems.

The residuals problem (2) becomes less of a concern if we look at relative residuals. Letting γ be the average of the absolute values of the non-zero entries in the Schur matrix, $\|\mathbf{A}\mathbf{v} - \lambda\mathbf{v}\|/\gamma$ is less than $1e-5$ for all BAL problems—often much smaller. The residuals are simply on the scale of the entries in the matrix.

The parallel property (3) typically does not hold because in higher dimensions most vectors are nearly orthogonal, but this does not affect our intended use case of visualization. We rely on a variational property of eigenvectors: they minimize a certain optimization problem (i.e., Equation (5))

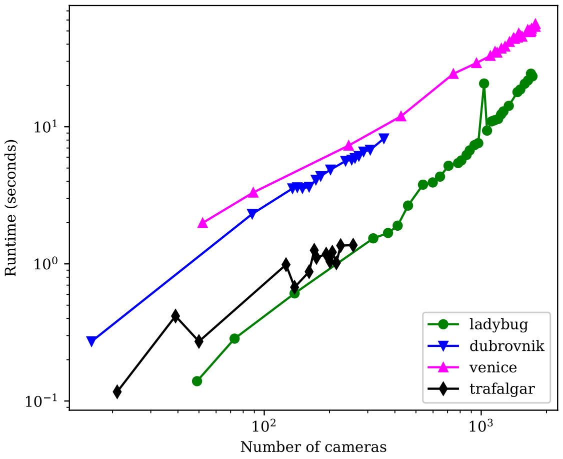


Figure 5: Runtimes for computing the 20 largest eigenvectors of the BAL problems.

called the Ritz-Rayleigh quotient. The ARPACK eigensolver converges when it has minimized the Ritz-Rayleigh quotient within a Krylov subspace. For all of the BAL problems, we observe that ARPACK converges to machine precision in six or fewer iterations. Furthermore, when we increase the size of the Krylov subspace (from 20 to 200, on a ≈1700 camera problem) the computation becomes much slower, but the results are indistinguishable. This gives us confidence that our vectors are good minimizers of the Ritz-Rayleigh quotient within the subspace, that the subspace is representative of the entire operator, and that our eigenvectors represent directions of maximum variance.

4.3. Runtime Experiment

We implemented our method in the Julia programming language [5]. The eigen-computations were performed with Arpack.jl [14] linked to OpenBLAS. Figure 5 gives runtimes on a 2015 MacBook Pro with a 2.2GHz Intel i7 processor. The runtimes include forming the reduced Schur system and solving, but exclude file I/O operations. On BAL problems ≈75% of the runtime is spent forming the Schur complement matrix—mostly sparse matrix multiplication. Only ≈25% is spent in eigen-computations. Even the largest scenes run in a bit less than one minute.

4.4. Discussion: Limited Usefulness of Eigenvalues

It would be great to be able to go beyond visualizing uncertainty to quantifying it. At a first glance, the eigenvalues that go with our eigenvectors appear to be a useful signal. However, this signal is confounded for several reasons.

First we note that maximum eigenvalues are brittle measures. See Figure 4, which shows $\lambda_{\max}$ for each of the problems in the BAL dataset. Notice that in several cases only a few cameras are added between one problem and the next, but $\lambda_{\max}$ increases two orders of magnitude.

Second, the issue of units sharply limits how much we can compare eigenvalues across different datasets. We saw in Section 3.3 that bundle adjustment problems have a gauge. It may seem that we have already dealt with this: the covariance matrix is initially invariant under rotations and translations, but not under scalings of the spatial parameters. However, our units normalization enforces scale invariance. That is, the eigensystems are invariant to all gauge transformations.

However, our choice of units was necessarily ad hoc. The median camera distance in one scene does not correspond to the same measurement for a different problem. Since BAL problems come in a series, it is mostly meaningful to compare eigenvalues within a series. (Dickscheid *et al.* [7] make this rigorous by gauge-aligning scenes that share cameras in common before comparing covariances.) But, looking at Figure 4, we can not necessarily say that Trafalgar is reconstructed more accurately than Dubrovnik. Eigenvalues can only be compared between models with the same spatial units. This difficulty could be mitigated given extra information to register our models to a metric scale. There would still be an ad hoc choice about how to compare units of rotation and translation, but the scale of the translation units would no longer be arbitrary. We leave further quantification of uncertainty to future work.

5. Conclusion and Future Directions

This work is a foray into uncertainty-aware Structure from Motion computation. We have focused on an application to uncertainty visualization: by computing the largest eigenvectors of the (Schur-reduced) covariance matrix we can animate the dominant modes of uncertainty in a solution. These eigenproblems are numerically difficult. We give a combination of problem size reduction, unit scaling treatment, gauge approach, and solver that yield acceptable quality solutions.

These visualizations provide insight into the failure modes of reconstructions. But beyond visualizations, which only help diagnose problems in SfM, we hope to work towards using uncertainty information proactively. One idea towards uncertainty-aware computation is that the eigenvectors we compute in this paper are precisely what is needed to form a low-rank approximation to the Mahalanobis distance on the space of camera extrinsics. This could be used to locate new viewpoints that specifically give information about directions of maximum uncertainty. Also, while the absolute magnitude of eigenvalues is hard to reason about, their relative rate of decay is a stable property that holds meaning between datasets.

References

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2007. 2

[2] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. Seitz, and R. Szeliski. Building rome in a day. *Communications of the ACM*, 54(10):105–112, 2011. 2

[3] S. Agarwal, K. Mierle, and Others. Ceres solver. http://ceres-solver.org. 2, 3

[4] S. Agarwal, N. Snavely, S. Seitz, and R. Szeliski. Bundle adjustment in the large. In *ECCV*, 2010. 5

[5] J. Bezanson, A. Edelman, S. Karpinski, and V. Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017. 8

[6] C. Bishop. *Pattern recognition and machine learning*. Springer Science+ Business Media, 2006. 3

[7] T. Dickscheid, T. Läbe, and W. Förstner. Benchmarking automatic bundle adjustment results. In *ISPRS*, 2008. 8

[8] G. Golub. *Some modified eigenvalue problems*. Stanford University. Computer Science Department, 1971. 4

[9] S. Haner and A. Heyden. Covariance propagation and next best view planning for 3d reconstruction. In *ECCV*, 2012. 2

[10] R. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation averaging. In *IJCV*, 2013. 4

[11] K. Kanatani. *Statistical optimization for geometric computation*. Dover, 1996. 1, 2

[12] K. Kanatani and D. Morris. Gauges and gauge transformations for uncertainty description of geometric structures with indeterminacy. In *Transactions on Information Theory*, 2001. 2, 3

[13] A. Knyazev. Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method. *SIAM journal on scientific computing*, 23(2):517–541, 2001. 4

[14] R. Lehoucq, D. Sorensen, and C. Yang. *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*, volume 6. Siam, 1998. 4, 8

[15] M. Lhuillier and M. Perriollat. Uncertainty ellipsoids calculations for complex 3d reconstructions. In *ICRA*, 2006. 3

[16] M. Polic, W. Forstner, and T. Pajdla. Fast and accurate camera covariance computation for large 3D reconstructions. In *ECCV*, 2018. 2

[17] M. Polic and T. Pajdla. Camera uncertainty computation in large 3D reconstruction. In *3DV*, 2017. 2

[18] J. Schonberger and J.-M. Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 2

[19] L. Trefethen and D. B. III. *Numerical linear algebra*, volume 50. Siam, 1997. 7

[20] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment—a modern synthesis. In *International Workshop on Vision Algorithms*, 1999. 1, 2, 3

[21] K. Wilson and N. Snavely. Robust global translations with 1DSfM. In *ECCV*, 2014. 5