

# Towards Pointless Structure from Motion: 3D reconstruction and camera parameters from general 3D curves

Irina Nurutdinova  
Technische Universität Berlin  
irina.nurutdinova@tu-berlin.de

Andrew Fitzgibbon  
Microsoft, Cambridge, UK  
awf@microsoft.com

## Abstract

Modern structure from motion (SfM) remains dependent on point features to recover camera positions, meaning that reconstruction is severely hampered in low-texture environments, for example scanning a plain coffee cup on an uncluttered table.

We show how 3D curves can be used to refine camera position estimation in challenging low-texture scenes. In contrast to previous work, we allow the curves to be partially observed in all images, meaning that for the first time, curve-based SfM can be demonstrated in realistic scenes.

The algorithm is based on bundle adjustment, so needs an initial estimate, but even a poor estimate from a few point correspondences can be substantially improved by including curves, suggesting that this method would benefit many existing systems.

## 1. Introduction

While 3D reconstruction from 2D data is a mature field, with city-scale and highly dense reconstruction now almost commonplace, there remain some serious gaps in our abilities. One such is the dependence on texture to resolve the aperture problem by supplying point features. For large-scale outdoor scanning, or for cameras with a wide field of view, this is not an onerous requirement, as there is typically enough texture in natural scenes to obtain accurate camera poses and calibrations. However, many common environments in which a naïve user might wish to obtain a 3D reconstruction do not supply enough texture to compute accurate cameras. In short, we cannot yet scan a “simple” scene such as a coffee cup on a plain table.

Consider figure 1, showing a simple scene captured with a consumer camera. Without the calibration markers, which are there so that we can perform ground-truth experiments, there are no more than a few dozen reliable interest points. Furthermore, some of those points are on T-junctions, so correspond to no real 3D point, further hindering standard

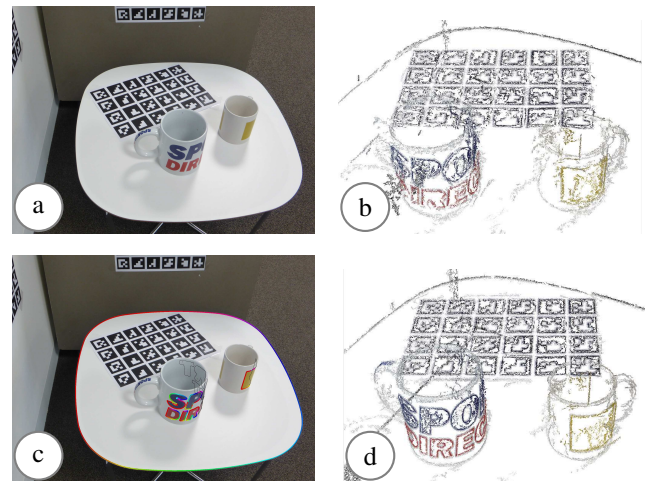


Figure 1: (a) A typical low-texture scene (one of 21 “Cups” images, markers are for ground-truth experiments). Sparse point features lead to poor cameras, and poor PMVS2 reconstruction (b). Reprojection of the 16 curves used to upgrade cameras (c), yielding better dense reconstruction (d).

SfM reconstruction. However, the *curves* in the images are a rich source of 3D information. For curved surfaces, reconstructing surface markings provides dense depth information on the surface. Even in areas where no passive system can recover depth, such as the white tabletop in this example, the fact of having a complete curve boundary, which is planar, is a strong cue to the planarity of the interior. The silhouette curves (which we do not exploit in this work) are yet another important shape cue.

Related work is covered in detail in §2 but in brief, while some previous research has looked at reconstruction of space curves, and other efforts have considered camera pose estimation from line features, very few papers address the simultaneous recovery of general curve shape and camera pose. Those that do have either assumed fully visible curves (or precisely, known visibility masks) in each image, or have shown only qualitative camera recovery under an

orthographic model. This paper is the first to show improvement of a state-of-the-art dense reconstruction pipeline (VisualSFM [21] and PMVS2 [8]) by using curves.

Our primary contribution is a formulation of curve-based bundle adjustment with the following features:

1. curves are not required to be fully visible in each view;
2. not limited to plane curves—our spline parameterization allows for general space curves;
3. point, curves, and cameras are optimized simultaneously.

A secondary, but nevertheless important, contribution lies in showing that curves *can* provide strong constraints on camera position, so should be considered as part of the SfM pipeline. Finally, we note that nowhere do we treat lines as a special case, but we observe that many of the curves in our examples include line segments, so that at least at the bundle adjustment stage they may not need to be special-cased.

A potential criticism of our work is that “it’s just bundle adjustment”—we wrote down a generative model of curves in images and optimized its parameters. However, we refute this: despite the problem’s long standing, the formulation has not previously appeared.

Limitations of our work are: we assume the image curves are already in correspondence, via one of the numerous previous works on automatic curve matching [19, 12, 2, 5, 10, 12]. We also cannot as yet recover motion without some assistance from points, hence our title is “towards” point-less reconstruction. However, despite these limitations, we will show that curves are a valuable component of structure and motion recovery.

## 2. Related work

The venerable term “Structure from Motion” is of course a misnomer. It is the problem of simultaneously recovering the 3D “structure” of the world *and* the camera “motion”, that is the extrinsic and intrinsic calibration of the camera for each image in a set.

On the “structure” side, curves have often been studied as a means of enriching sparse 3D reconstructions (figure 2 attempts to illustrate). Baillard *et al.* [2] showed how line matching could enhance aerial reconstruction, and Berthilsson *et al.* [3] showed how general curves could be reconstructed. Kahl and August [12] showed impressive 3D curves reconstructed from real images, and recently Fabri and Kimia [5] and Litvinov *et al.* [15] showed how reconstruction of short segments could greatly enhance 3D reconstructions. Other work introduced more general curve representations such as nonuniform rational B-splines [23], subdivision curves [11], or curve tracing in a 3D probability distribution [20].

The key to this paper, though, is the use of curves to obtain or improve the “motion” estimates. Two classes of

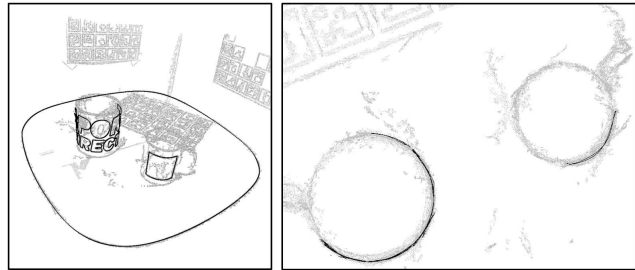


Figure 2: Two views of final curves, overlaid on dense point reconstruction. Curves provide a more semantically structured reconstruction than raw points.

existing work pertain: multiple-view geometry for a small number of views, and curve-based bundle adjustment. It has been known for decades that structure and motion recovery need not depend on points. The multiple-view geometry of lines [6], conics [13], and general algebraic curves [14] has been worked out. For general curved surfaces, and for space curves, it is known that the epipolar tangencies [17] provide constraints on camera position. However, these various relations have a reputation for instability, possibly explaining why no general curve-based motion recovery system exists today. Success has been achieved in some important special cases, for example Mendonça *et al.* [16] demonstrated reliable recovery of camera motion from occluding contours, in the special case of turntable motion.

Success has also been achieved in the use of bundle adjustment to “upgrade” cameras using curves. Berthilsson *et al.* [3] demonstrated camera position recovery from space curves, with an image as simple as a single “C”-shaped curve showing good agreement to ground truth rotation values. However, as illustrated in figure 3, because they match model points to data, they require that each curve be entirely visible in every image. Prasad *et al.* [18] use a similar objective to recover nonrigid 3D shape for space curves on a surface, and optimize for scaled orthographic cameras, but show no quantitative results on camera improvement. Fabri and Kimia [5] augment the bundle objective function to account for agreement of the normal vector at each point, but retain the model-to-data formulation, dealing with partial occlusion by using curve fragments rather than complete curves. Cashman and Fitzgibbon [4] also recover cameras while optimizing for 3D shape from silhouettes, but deal with the occluding contour rather than space curves, and use only scaled orthographic cameras.

### 2.1. Model-to-data versus data-to-model

A key distinction made above was that most existing algorithms (all but [23, 4]) match model to data, probably because this direction is easily optimized using a distance transform. By this we mean that the energy function they

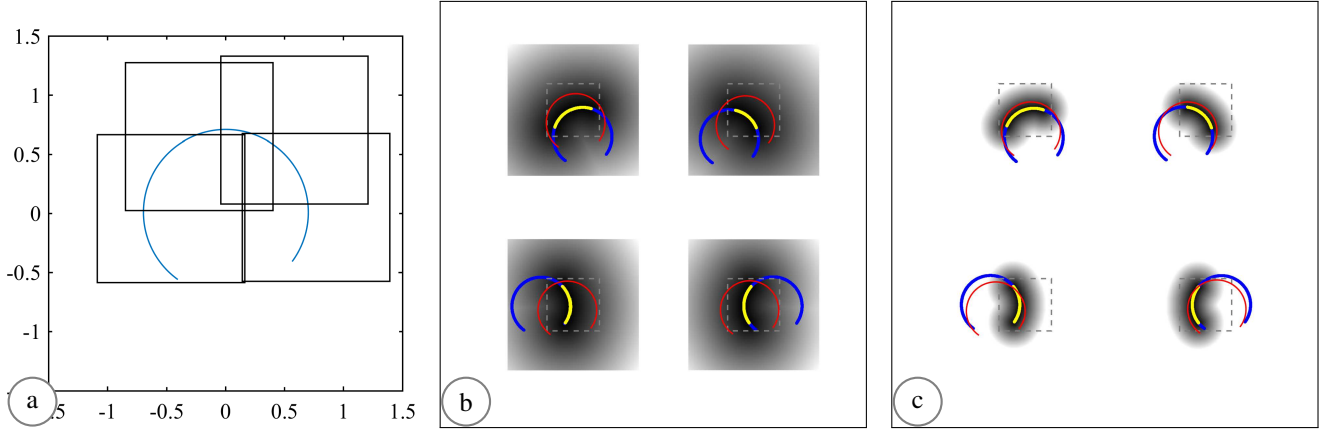


Figure 3: The problem with the model-to-data objective (§2.1). (a) A 2D setup with a single curve (blue) viewed by four cameras (fields of view shown black) parameterized by 2D translation and optionally scale. (b) Illustration of the cost function surfaces in each image for the model-to-data objective used in most previous work [5, 18, 3]. Each image is represented by a distance transform of the imaged points (yellow dots), and the bundle objective sums image-plane distances. Even when varying only camera translation, the global optimum for the L2 objective (red curves) is far from the truth (blue). If varying scale as well as translation (analogous to our perspective 3D cameras), the situation is even worse: the global optimum is achieved by setting the scale to zero. (c) Attempting to model occlusion using a robust kernel reduces, but does not eliminate the bias. All existing model-to-data techniques must overcome this by explicitly identifying occlusions as a separate step. If that step is based on rejection of points with large errors, it is already subsumed in (c), i.e. it doesn’t remove the bias.

minimize in bundle adjustment involves a sum over the points of the 3D curve, rather than summing over the image measurements. However, as illustrated in figure 3, in the presence of any partial occlusion, the global optimum of this formulation does not coincide with the ground truth, and indeed can be strongly biased. Existing work can avoid the bias by knowing, for each view, exactly what subset of the model curve is visible, but this is in general unknowable and any attempt to do it using image-plane distances *must* amount to some form of robust estimator (fit, reject large values, re-fit), which does not fix the bias. As shown below (and by [23, 4]), the data-to-model objective can be efficiently minimized. Thus, to our knowledge ours is the first work to optimize camera parameters from curves using an intrinsically occlusion-resistant objective function.

### 3. Algorithm

We consider  $N$  views of a scene that contains  $M$  3D curves, *e.g.* texture edges or sharp object outlines. Our goal is to reconstruct 3D curves from their image projection and to refine the camera calibration using these curves. Each 3D curve is partly visible in several views.

We define a 3D curve as a function  $C$  mapping from an interval  $\Omega \subset \mathbb{R}$  to  $\mathbb{R}^3$ . We will use curves whose shape is parameterized, typically by a set of control vertices,  $\mathbf{X} = \{X_1, \dots, X_K\} \subset \mathbb{R}^3$ , and we will write  $C(t; \mathbf{X})$  to show the dependence on both the curve parameter  $t$  and the shape

parameters  $\mathbf{X}$ . In this work we will use a piecewise cubic spline, defined as

$$C(t; X_1, \dots, X_K) = \sum_{k=0}^3 X_{[t]_k} \phi_k(t - [t]) \quad (1)$$

for fixed piecewise smooth basis functions  $\phi_{0..3}$ , and  $\Omega$  is the interval  $[1, K - 2)$ . Note that despite the apparently discontinuous floor operator, this is a very simple function, piecewise cubic in  $t$  and linear in  $\mathbf{X}$ , whose derivatives with respect to  $t$  and  $\mathbf{X}$  are well defined at each  $t \in \Omega$  because the basis functions are chosen so that the spline is smooth. The definition above uses fixed uniform knots, but in practice it is easy to switch to NURBS [23] or subdivision curves [4] if desired. Our code is quite agnostic to the parameterization of  $C$ .

We also consider a quite general representation of cameras, projecting 3D points to 2D images. Each dataset will comprise  $N$  images, and image  $n$  has corresponding unknown camera parameters  $\theta_n$ , comprising for example translation, rotation, focal length and radial distortion. A projection function  $\pi$  is defined which applies a vector of camera parameters  $\theta$  to a 3D point  $X$  and projects into  $\mathbb{R}^2$ :

$$\pi(X; \theta) \in \mathbb{R}^2 \quad (2)$$

The image data comprises a list of curve segments, numbered 1 to  $S$ . As noted above, we assume correspondence will be provided by an existing algorithm, even

though in practice one would almost certainly iterate reconstruction and correspondence finding, this does not change the mechanics of our optimization. Let the  $M$  3D curves' unknown shape parameters be written  $\mathbf{X}_m = \{X_{m1}, \dots, X_{mK}\}$  for  $m = 1..M$ .

A detected 2D curve segment  $s$  has associated image index  $n_s$  (we will sometimes write this  $n(s)$  if fonts get too small) and 3D curve index  $m_s$  (or  $m(s)$ ), and a list of 2D points of length  $P_s$ :

$$Z_s = \{\mathbf{z}_{sp}\}_{p=1}^{P_s} \quad (3)$$

Note that because of occlusion, there could be multiple detections in one image corresponding to the same model curve, which simply means two detections  $s, s'$  may have  $n(s) = n(s'), m(s) = m(s')$ . The problem statement and occlusion sensitivity are unaffected.

### 3.1. Curve and camera bundle adjustment

The objective we would like to optimize is to match the data to the model, that is to sum over the data, and measure the distance from each image point to its closest point on the model. The curve bundle adjustment objective is simply the sum over all detected curves

$$E(\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_N, \mathbf{X}_1, \dots, \mathbf{X}_M) = \sum_s E_s(\boldsymbol{\theta}_{n(s)}, \mathbf{X}_{m(s)}) \quad (4)$$

For curve  $s$ , the objective is the sum of closest-points

$$E_s(\boldsymbol{\theta}, \mathbf{X}) = \sum_{p=1}^{P_s} \min_t \|\mathbf{z}_{sp} - \pi(C(t; \mathbf{X}); \boldsymbol{\theta})\|^2 \quad (5)$$

When matching model to data, the equivalent closest-point operation can be easily implemented using a distance transform on the image [18, 5, 11]. However, as the  $t$ -minimization here is against the *model* curve, the distance transform would be 3D, and would need to be updated on every optimizer iteration. An analytic solution is nontrivial, involving a rootfinding operation for every edgel. However, a simple solution is obtained by using the “lifting” technique described in [18]. The  $t$  variables (or “correspondences”) are renamed

$$\sum_{p=1}^{P_s} \min_{t_{sp}} \|\mathbf{z}_{sp} - \pi(C(t_{sp}; \mathbf{X}); \boldsymbol{\theta})\|^2 \quad (6)$$

after which the min and sum can swap (this is exact)

$$\min_{T_s} \sum_{p=1}^{P_s} \|\mathbf{z}_{sp} - \pi(C(t_{sp}; \mathbf{X}); \boldsymbol{\theta})\|^2 \quad (7)$$

where  $T_s = [t_{s1}, \dots, t_{sP_s}]$  is the vector of all correspondences for detected segment  $s$ . The final step is to gather the correspondences into an overall objective

$$E(\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_N, \mathbf{X}_1, \dots, \mathbf{X}_M, T_1, \dots, T_S)$$

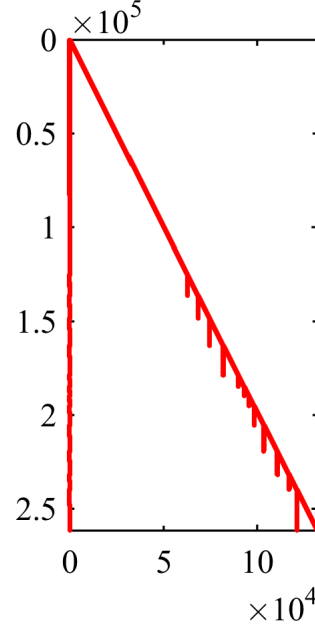


Figure 4: Jacobian sparsity pattern for “Cups” example

Thus our system which used to have say  $3MK$  unknowns in addition to point-based bundle adjustment, now has  $3MK + 100S$  additional unknowns (assuming  $P_s \approx 100$  on average). This apparent explosion in the number of unknowns is however compensated by the simplification of the objective. This is a non-linear least squares problem which can be readily optimized using the Levenberg-Marquardt algorithm, provided we take care to make use of the sparsity of the Jacobian. An example Jacobian is depicted in figure 4.

#### 3.1.1 Regularization

Control of the smoothness of the reconstructed curve is primarily by choosing the number  $K$  of control vertices of the spline. Too small a value means that complex curves cannot be modelled, while too large a value could give rise to overfitting. In practice, overfitting is less of a problem, so we set  $K = 12$  for all curves in all experiments. Ultimately of course this should be set automatically, but the fact that a constant works for a range of situations is encouraging.

One further option for regularization is enabled by the explicit exposing of the correspondences  $t_{sp}$ . Because successive points on an image curve are likely to be close in 3D (even in extreme cases like a 2D cusp corresponding to a smooth 3D section), we have experimented with adding a regularizer

$$E_{\text{reg}}(T_1, \dots, T_S) = \sum_s \sum_{p=1}^{P_s-1} \|t_{sp} - t_{s,p+1}\|^2 \quad (8)$$



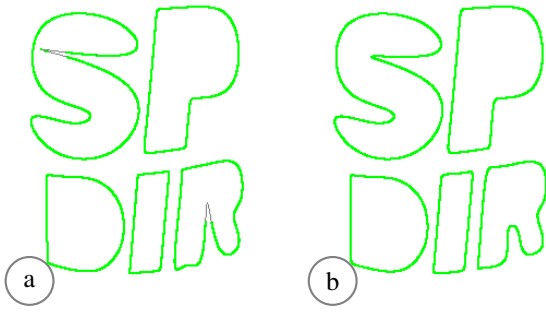


Figure 5: Points on the spline that have correspondences in the images are shown in green, full spline in black. (a) Without regularization some parts of the spline may not have correspondence in the images. (b) With regularization the 3D spline projects more completely onto the images.

In practice this acts like a “spring” term, keeping correspondences to a subset  $\Omega$ , with a beneficial effect on overfitting. The effect of the regularization is illustrated in figure 5. It is implemented by adding residuals to the Levenberg-Marquardt objective, and introduces another large but very sparse block in the Jacobian.

### 3.2. Implementation: Initial estimate

Given matched 2D curves and approximate cameras, it is straightforward to use the epipolar constraint to generate feature correspondences and create a 3D reconstructed curve. However, we can benefit from the engineering of existing dense multiview stereo systems such as PMVS2 [8], and simply identify 3D points which project near to our matched curves in two or more views. Although noisy, these points are sufficient to initialize our system (see figure 6).

In order to provide an initial estimate for the correspondences  $\{t_{sp}\}$ , we project the 3D points the view where the curve is least occluded. It is worth noticing that our method does not require a view where curve is fully visible. This is used only to obtain a very rough ordering on the 3D points on the initial spline, allowing the algorithm to sample control points in the correct order. The points are then uniformly sampled on the 2D curve and control points are initialized using the 3D dense stereo point with projection closest to this sampled point. This initial 3D spline is projected onto all images in which the curve appears, and  $t$  is initialized to the  $t$  value of the closest point. As initial camera calibration may be inaccurate, the initial spline can project far from the curve. Therefore we first align the 2D curve with the projection of the initial spline using an iterated closest point algorithm (ICP). Optionally we use a dynamic programming strategy [7, 4] to refine the curves in a second optimization pass (see figure 7), but the results on camera improvement did not use this.

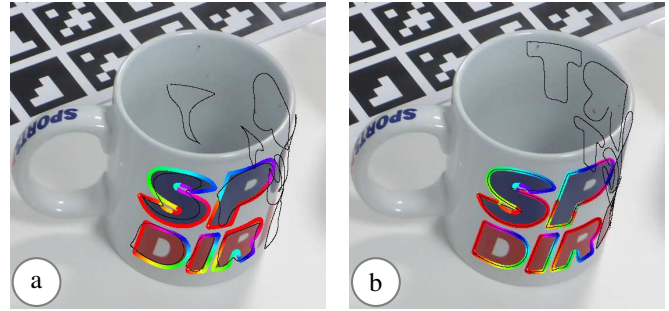


Figure 6: (a) Initial estimates for splines and correspondences  $T$ . The rainbow colourmap on the image at edge  $p$  in curve  $s$  indicates the value of  $t_{sp}$ . These are before dynamic programming, indicating that the initial  $t$  estimates are quite good, even when the shape is poor (see the far letter “T”). (b) Final optimized curve, without priors on  $T_s$ . Some non-monotonicity is visible on the “S” but this does not prevent us from getting improved camera estimates.



Figure 7: Curve refinement using dynamic programming. (a) Curve initialized using DP. (b) Final optimization result. (c) Reinitialized using optimized control points and cameras. (d) Final optimized curves. The fidelity of the complex curves (text) improves, but we begin to see overfitting on the simple rectangle. Future work will adaptively adjust the number of control points.

### 3.3. Implementation: Ceres solver

The Levenberg Marquardt algorithm was implemented in C++ using the Ceres solver framework [1]. We implemented analytic derivatives for speed and accuracy, and used the dynamic sparsity option to deal with the non-constant sparsity of our Jacobian. A typical problem size (for the “Cups” example) was 400,000 residuals, 140,000 parameters, and 5,000,000 nonzero Jacobian entries. Opti-

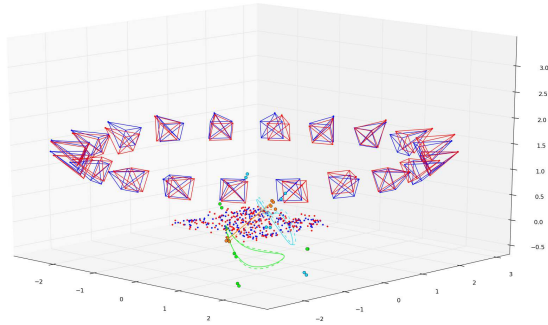


Figure 9: Synthetic scene. Blue cameras are ground truth cameras and red cameras are used for initialization. Dashed splines are initial estimates.

mization took 3 minutes to run 65 LM iterations.

## 4. Experimental results

The method was evaluated on synthetic and real datasets.

### 4.1. Synthetic setup

We generated a synthetic scene (figure 9) viewed by 20 cameras distributed uniformly on a circle. The scene contains 3D points  $P$  sampled nearly on the plane, simulating a common case in indoor 3D reconstruction where the floor is textured but there is little texture above the floor level. To these points were added several randomly generated splines. These 3D points were projected into images of size  $400px \times 300px$ . To generate the image curves we densely sampled points on the splines and projected them onto the views. We added zero-mean Gaussian noise with standard deviation of 0.2 pixels to all image measurements. We demonstrate the results on an example with three splines which were sampled at 400 points each and 200 points sampled nearly on the plane. In order to measure the effect of including curves in the camera estimation, we sampled a set of test points that were not included in our optimization. The test points were uniformly sampled inside the cube  $[-1, 1][[-1, 1][[-1, 1]$  enclosing the scene.

We initialized unknown parameters of the optimization with perturbations of the ground truth values, obtained by adding Gaussian noise. For camera translation and orientation, 3D points and spline control points we use a standard deviation of 0.05. Initial cameras are shown in figure 9 next to the ground truth cameras shown in blue and initial splines are drawn with dashed line in figure 10. We also randomly perturbed the correspondences  $t_{sp}$  with Gaussian noise with standard deviation of 0.1.

We compared three different cases. First, we run standard bundle adjustment using only points. In the second case we added curves to the result and optimized jointly

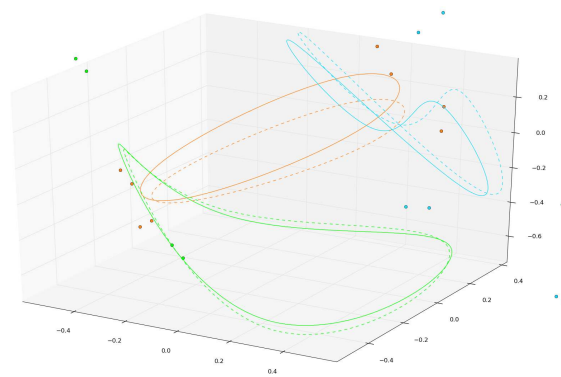


Figure 10: Initial estimates for splines.

with points and curves. In the third case, we first optimize for curves having fixed the cameras, and then run a joint optimization, a strategy one might reasonably consider for a real world scene. After the optimization, camera translation and angular error were computed with respect to the known ground truth. We also computed the 3D errors for the test points. As the computed reconstruction is unique up to a similarity transform, in order to compute camera errors and 3D errors of test points we first solved for the transformation from the reconstruction to the ground truth coordinate system by minimizing the sum of 3D Euclidean distance computed at the test points and camera centers. This can potentially underestimate the error, but not in a way that favours any particular algorithm. In addition, we measured reprojection error on the test points, as this is a metric we can also compute on test points in real images. If we see a correlation between reprojection error and the 3D errors in the synthetic set, we may hope that this is a useful indicator of performance on real data. Reprojection errors are computed by performing optimal triangulation of the points based on noise-free projections and reporting the residual.

We did experiments with different numbers of points. Each experiment was repeated 20 times having different measurement noise and different set of points visible in the images, as well as different initialization for points and curves. The mean and average RMS errors are visualized in figure 11. As expected, the accuracy of the point-based calibration deteriorates as the number of visible points decreases. Adding curves to the optimization reduces that error significantly even when many points are available, and more when the number of points reduces. It is also noteworthy that strategy of first optimizing curves (one half-iteration of block coordinate descent) is not as effective as simply jointly optimizing all parameters. We also note that reprojection error is a good proxy for the 3D errors.

We did another series of experiments that demonstrates the robustness of our approach to occlusions (see figure 12).

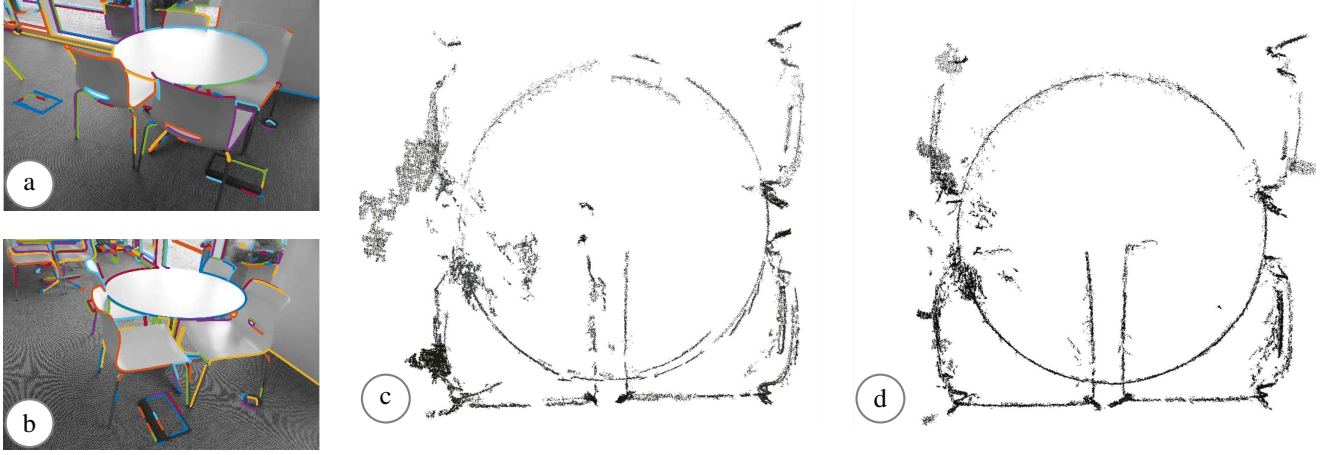


Figure 8: (a,b) Two views of “Canteen” scene, with detected edges overlaid. (c) Dense reconstruction with points-only cameras. (d) Dense reconstruction with points-and-curves cameras. The ground plane has been clipped to show the tabletop and chairs better.

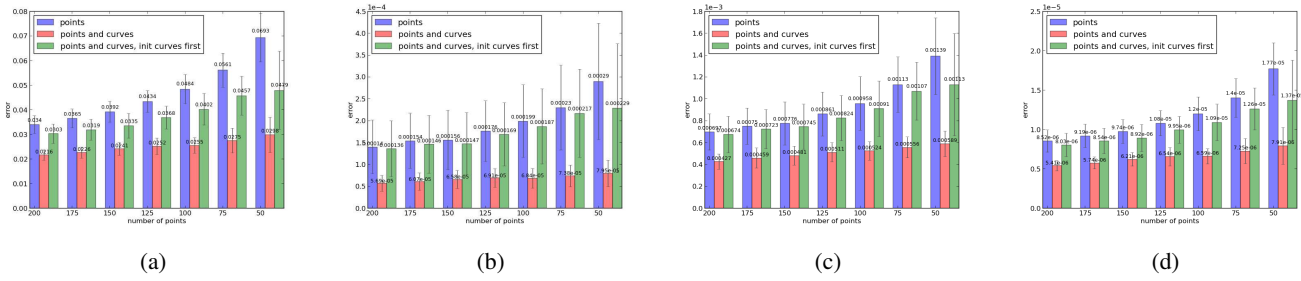


Figure 11: **Accuracy with decreasing numbers of point features.** RMS errors of bundle adjustment with points only (red bars), with points and curves (blue), and with curves initialized by block coordinate descent (green). a) Reprojection error of the test points b) 3D error of the test points c) Camera translation error d) Camera angular error. Adding curves clearly reduces all errors, particularly as the number of points decreases (note descending scale on X axis). Note too that the trend revealed by reprojection error follows the more stringent 3D metrics.

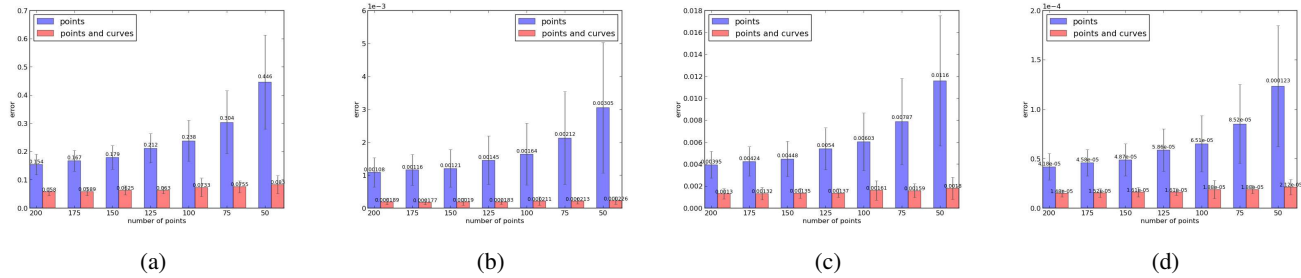


Figure 12: **Accuracy under increasing occlusion.** Curves are 25% occluded, and point tracks are 5 frames. RMS errors with points only (blue) and points and curves (blue). a) Reprojection error of the test points b) 3D error of the test points c) Camera translation error d) Camera angular error.

The results were evaluated in the first two of the three above mentioned cases. Each set of points was visible in only 5 consecutive views and only 75% of the curve was visible in each image, forming 2-3 curve segments. As a result,

the error of the points-only bundle has increased a lot and exhibits much larger variation, whilst adding curves to it keeps the errors low despite occlusions.

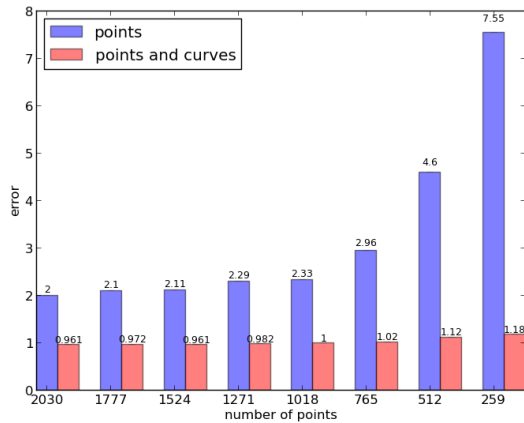


Figure 13: Reprojection error on test markers on the “Cups” scene. Even with large numbers of points, adding curves improves the test error, and as the number of points reduces, the improvement is increasingly significant.



Figure 14: Three of ten views of a modern building (top). This is a very difficult case for curve-based matching as most of the curves are nearly parallel to epipolar lines. Our system produces a reasonable reconstruction of the curves (bottom), but does not significantly improve the camera estimates. Note that the building does actually have a rhomboidal floorplan.

## 4.2. Real data

We acquired images of a real scene (figure 1) that doesn’t have enough texture to be reliably reconstructed, but has 3D curves. The image sequence consists of 21 images with resolution 2000x1500. We placed markers in the scene in order to evaluate the accuracy of our approach. Initial camera calibration was obtained using publicly available structure-from-motion software VisualSFM [21, 22]. Curves on the images were found using a local implemen-

tation of the Canny edge detector with subpixel refinement and Delaunay-based edge linking. Corresponding curves were picked with the help of GUI tool, where occasional false connections made by the edge linker were split. We used the border of the table, the large letters on one cup and the (curved) rectangle on another one.

To evaluate the algorithm, we used corners of the markers as test points. The corners were detected using AR-Toolkit [9] followed by a local implementation of subpixel refinement. Then the corners were linearly triangulated and further optimized using bundle adjustment while keeping the cameras fixed (i.e. MLE triangulation).

Figure 13 shows the reprojection error of the markers for different number of detected 3D points. As with the synthetic sequence, the test reprojection error with points and curves is consistently low, even removing 90% of the points. Referring back to figure 1, the low reprojection error translates to a visual improvement in the dense stereo reconstruction, so we have some confidence that the cameras are indeed improved.

Finally, figure 14 shows a particularly challenging example, where the camera motion is restricted to the ground plane, and the 3D curves are mostly approximately parallel to the ground. While we observe no improvement or disimprovement in the cameras, we do get the benefit of curves in the reconstruction, linking together an unorganized point cloud, and regularizing the structure along the curves without oversmoothing perpendicular to the curve.

## 5. Discussion

We have shown that incorporating curve correspondences into bundle adjustment can yield valuable improvements in camera estimation over using points alone. In contrast to previous work, our objective function for bundle adjustment is a sum over data, rather than over the model, so is intrinsically more resistant to occlusion. While components of our objective have been found in previous works, ours is the first to combine the full perspective camera model with the data-to-model cost, and to show improvements in the output of a state-of-the-art dense stereo system using our cameras.

Future work has two strands. Practically, we should incorporate an automated matcher in order to build an end-to-end system. Theoretically, our objective function does not penalize overly complex curve models. It would be useful to automatically adapt model complexity so that the reconstructed curves are more attractive.

## References

- [1] S. Agarwal, K. Mierle, and Others. Ceres solver. <http://ceres-solver.org>, 2015. 5
- [2] C. Baillard, C. Schmid, A. Zisserman, and A. Fitzgibbon. Automatic line matching and 3d reconstruction of buildings



- from multiple views. In *ISPRS Conference on Automatic Extraction of GIS Objects from Digital Imagery*, volume 32, pages 69–80, 1999. 2
- [3] R. Berthilsson, K. Åström, and A. Heyden. Reconstruction of general curves, using factorization and bundle adjustment. *International Journal of Computer Vision*, 41(3):171–182, 2001. 2, 3
- [4] T. J. Cashman and A. W. Fitzgibbon. What shape are dolphins? Building 3D morphable models from 2D images. *IEEE Trans. Pattern Anal. Mach. Intell.* 2, 3, 5
- [5] R. Fabbri and B. Kimia. 3D curve sketch: Flexible curve-based stereo reconstruction and calibration. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1538–1545, 2010. 2, 3, 4
- [6] O. Faugeras and B. Mourrain. On the geometry and algebra of the point and line correspondences between  $n$  images. In *Proc. ICCV*, pages 951–956, 1995. 2
- [7] M. Frenkel and R. Basri. Curve matching using the fast marching method. In *Proc. EMMCVPR*, pages 35–51, 2003. 5
- [8] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(8):1362–1376, 2010. 2, 5
- [9] HIT Lab, University of Washington. AR Toolkit, 2015. <http://hitl.washington.edu/artoolkit>. 8
- [10] F. Jung and N. Paparoditis. Extracting 3d free-form surface boundaries of man-made objects from multiple calibrated images: a robust, accurate and high resolving power edgel matching and chaining approach. *Intl Arch of Photogrammetry Remote Sensing and Spatial Information Sciences*, 34(3/W8):39–46, 2003. 2
- [11] M. Kaess, R. Zboinski, and F. Dellaert. MCMC-based multi-view reconstruction of piecewise smooth subdivision curves with a variable number of control points. In *Proc. European Conf. on Computer Vision*, pages 329–341. 2004. 2, 4
- [12] F. Kahl and J. August. Multiview reconstruction of space curves. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1017–1024. IEEE, 2003. 2
- [13] F. Kahl and A. Heyden. Using conic correspondences in two images to estimate the epipolar geometry. In *Computer Vision, 1998. Sixth International Conference on*, pages 761–766. IEEE, 1998. 2
- [14] J. Y. Kaminski and A. Shashua. Multiple view geometry of general algebraic curves. *International Journal of Computer Vision*, 56(3):195–219, 2004. 2
- [15] V. Litvinov, S. Yu, and M. Lhuillier. 2-manifold reconstruction from sparse visual features. In *IEEE Intl Conf 3D Imaging (IC3D)*, pages 1–8, 2012. 2
- [16] P. R. Mendonça, K.-Y. K. Wong, and R. Cipolla. Camera pose estimation and reconstruction from image profiles under circular motion. In *Proc. ECCV*, pages 864–877. Springer-Verlag, 2000. 2
- [17] J. Porrill and S. Pollard. Curve matching and stereo calibration. *Image and Vision Computing*, 9(1):45–50, 1991. 2
- [18] M. Prasad, A. W. Fitzgibbon, A. Zisserman, and L. Van Gool. Finding Nemo: Deformable object class modelling using curve matching. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2010. 2, 3, 4
- [19] C. Schmid and A. Zisserman. The geometry and matching of lines and curves over multiple views. *International Journal of Computer Vision*, 40(3):199–233, 2000. 2
- [20] D. Teney and J. Piater. Sampling-based multiview reconstruction without correspondences for 3d edges. In *Proc. IEEE 3DIM/PVT*, pages 160–167, 2012. 2
- [21] C. Wu. VisualSFM: A visual structure from motion system. 2, 8
- [22] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. Multicore bundle adjustment. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2011. 8
- [23] Y. J. Xiao and Y. Li. Optimized stereo reconstruction of free-form space curves based on a nonuniform rational B-spline model. *J Optical Soc Amer*, 22(9):1746–1762, 2005. 2, 3