Fully Automatic Calibration of LIDAR and Video Streams From a Vehicle

Stanley Bileschi
Massachusetts Institute of Technology
Cambridge, Massachusetts
bileschi@mit.edu

Abstract

This work describes a fully automatic technique to calibrate a geometric mapping between lidar and video feeds on a mobile ground-based platform. This data association is a crucial first step for any multi-modal scene understanding system which aims to leverage the complementary information of the two sensors. While several systems have been previously described which use hand-calibration or specific scenery to achieve this goal, the system described here is fully automatic and generates an accurate association without user intervention or calibration objects. The estimated parameters include the 7 classical camera parameters for a linear pinhole model, i.e., rotation, position, and focal length parameters, as well as an estimation of the radial distortion. The system uses a multi stage process to bootstrap difficult parameters based on robust estimates of easier ones.

The calibration algorithm is tested empirically using free online data supplied as part of the DARPA Urban Challenge autonomous vehicle competition [14]. Experiments are performed to illustrate the stability, and computation cost of the algorithm.

1. Introduction and Related Work

Lidar is an increasingly common type of active sensor which rapidly provides a 3D snapshot of scene geometry. Applications involving fused optical imagery and lidar include the construction of 3D models, particularly urban environments and historical heritage sites, scene understanding for robot navigation, and surveillance [2, 21, 3, 15, 23, 5]. Before detecting any objects or rendering any scenes, the data from the two sources must be fused into a common frame. For most systems, this is accomplished by carefully hand calibrating the sensors before collecting any data. This can be a laborious and error-prone process. Furthermore, if the camera parameters somehow change, as may happen by slight movements of the camera, or gross adjustments from a zoom lens or articulated mount, errors will accumulate and upstream processes will fail.

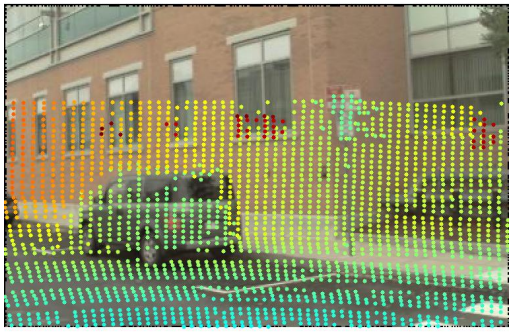

Figure 1. Lidar-Optical data association. Colorized depth information from the lidar sensor is overlaid on one frame. No hand calibration was performed. Note that the lidar sensor is higher than the camera. Best viewed in color.

Associating the two data streams is not trivial and many methods have been designed for the task. The hurdles involved in calibrating the two data streams are numerous, including occlusions, non-overlapping fields of view, and also the difficulties in matching two very different modes of data. Methods involving matching patterns of brightness, such as SIFT and other keypoint matching paradigms, will not function on the lidar data. Many calibration algorithms have nevertheless overcome these difficulties.

In [1] and [15] corresponding 2D and 3D points are hand selected to calibrate cameras to omni-directional range scanners. Corresponding straight lines or curves can be used instead of points to calibrate the sensors, as in the systems described in [11, 4, 9]. The system described in [19] associates 3D lines detected in the lidar structure to vanishing points detected in the image, producing impressive calibrations for urban modeling. The system described here uses a similar technique to refine the final calibration, matching 3D and 2D contours, but leverages temporal synchronicity to relieve the reliance on straight lines.

A different solution is described by Zhao in [24]. Their system uses advances in dense metric 3D from uncalibrated

2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops
978-1-4244-4441-0/09/$25.00 ©2009 IEEE

video to build a 3D point cloud directly from the video. The two point sets are then aligned via the iterated closest point algorithm (ICP). No calibration object is necessary, but the focal length and lens distortion are assumed known and an initial alignment must be provided. Nevertheless, this is an intriguing approach in that it may be possible to use additional video information to recover these missing parameters via autocalibration. Algorithms such as those described in [16, 17, 8, 22] have shown this ability, and this is an avenue for future research.

The contribution of this work is a robust system for automatically calibrating synchronous lidar and video signals using relative pose data from an inertial measurement unit (IMU). Comparatively few assumptions are made about the source data. Notably, no calibration objects are necessary, no camera parameters need to be assumed a priori, and no manual intervention is necessary. It is assumed that the camera principal point and center of distortion are at the image center, the pixels are square, and that the sensors fields of view overlap. After processing, the data from the two sensors are associated, and fused data is made available in both the lidar and camera coordinate frames.

Section 2 gives an overview of the system architecture and briefly describes the relevant subsystems, which are then described in order in sections 4 through 7. Empirical results and measurements, including accuracy and cost assessments are provided in Section 8.

2. Data and System Overview

The calibration system was designed around the data shared by [14], consisting of 4 video feeds and a synchronous lidar feed from a Velodyne scanning laser range finder. The video data is captured at a resolution of $[240 \times 376]$ and a frame rate of approximately 10 fps. Higher resolution video data is available but was unused. Two of the cameras (cams 2 and 3) point to the left and right, relative to the vehicle, while cams 1 and 4 view straight ahead. Camera 4 has a significantly narrower field of view than the other cameras. The lidar data is collected from 64 lasers at different pitches relative to the vehicle. The collection unit rotates around the axis perpendicular to the ground plane, sweeping out a fresh XYZ point cloud also at about 10 fps, though the data are updated asynchronously. Example data are illustrated in Fig. 2. While a scanning lidar was used in this system, a pushbroom lidar could also be used if the environment is static.

The autocalibration system was designed as a step by step process, refining successive estimates of the camera parameters. Camera parameters are calibrated to the lidar coordinate frame and are assumed to not change over the course of the data acquisition, which can be as short as 100 frames. Cameras are calibrated individually and no information is shared between them. The details of each subsystem are described below, but in brief, the camera spherical distortion is estimated first, using the detection of long contours. Next, the motion signal from the IMU is scanned for relatively linear sequences. These sequences are used to determine an epipole in the image frame corresponding to a direction of motion. Assuming for the moment that the camera and lidar are co-located, this association limits the camera's parameters to just two, rotation around the pinned direction and scaling via the focal length. An optimization procedure, described in Sec. 6, then estimates these two parameters by minimizing the reprojection and reconstruction error of tracked image points. The final stage is to refine the previous estimates and determine the camera offset in the lidar coordinate frame. This is accomplished by associating depth discontinuities with image contours and solving for the camera projection matrix satisfying these associations.

3. Notation

In this section the basic notations, coordinate frames, and sensor models are introduced. Projective geometry and homogenous coordinate systems are used.

The lidar sensor produces a set of 3D points $\mathbf{X}$ in the lidar coordinate frame, represented by homogenous 4 vectors $\mathbf{X} = [X, Y, Z, 1]^\mathsf{T}$. These points translate into points in the world coordinate frame via the euclidean $[4 \times 4]$ transformation matrix T_t, provided from the IMU at time t

The cameras are modeled as perspective projection cameras, and operate on points $\mathbf{X}$ via a $[3 \times 4]$ matrix $\mathbf{P}$ to generate image points $\mathbf{x} = [x, y, 1]^\mathsf{T}$ in the image plane.

$$\mathbf{x} = \mathbf{P}\mathbf{X} \tag{1}$$

$\mathbf{P}$ may be factored into an upper triangular projection matrix $\mathbf{K}$, a rotation matrix $\mathbf{R}$, and the camera's position in the lidar frame $\tilde{\mathbf{C}}$ as.

$$\mathbf{K} = \begin{bmatrix} f & & x_0 \\ & f & y_0 \\ & & 1 \end{bmatrix} \tag{2}$$

$$\mathbf{P} = \mathbf{K}\mathbf{R}[\mathbf{I}| - \tilde{\mathbf{C}}] \tag{3}$$

Where f is the focal length of the camera in image units, x_0 and y_0 are the principal point of the camera on the image plane, and $\mathbf{I}$ is the $[3 \times 3]$ identity matrix. Note that this model assumes square pixels with no skew. The further assumption will be made that the principal point is in the center of the image. This limits the camera model to 7 free parameters; 3 for rotation, 3 for position, and 1 for focal length in pixel units. Additionally, the camera radial distortion will be estimated, as described in the next section. Note that normalized image coordinates will be used, where $\{0, 0\}$ is the image center and $\{1, 1\}$ would be the top right corner, were the image square.

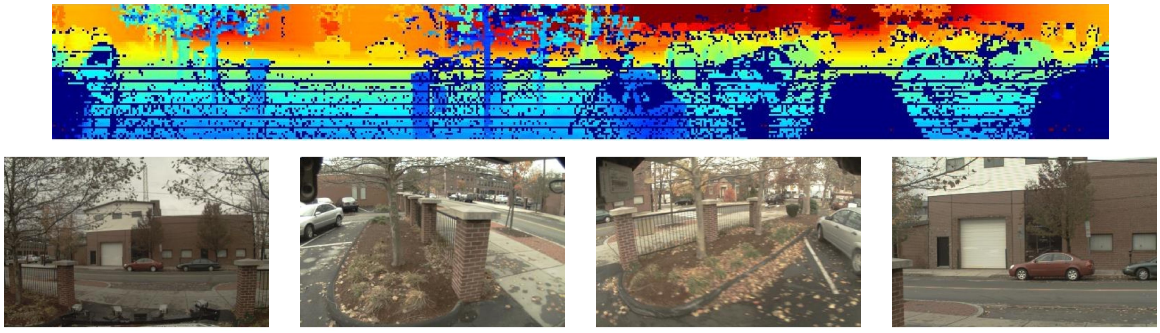

Figure 2. Data snapshot. *Top*: LIDAR range data. Color indicates depth with closer points in blue. Missing values were set to 0 for the illustration. From the left, the range image begins to the left of the vehicle, in the direction of cam2. It is easy to make out the pillars to the left and right of the gate in front of the vehicle and the two cars across the street. *Bottom*: Snapshots from the four video feeds. Note the narrower field of view in cam4. All optical imagery is $[240 \times 376]$.

4. Estimating Radial Distortion

Correcting the lens distortion is critical to building an accurate camera model. Without a good estimate of this distortion it is impossible to use linear techniques to solve for the internal camera parameters and even the optimal projection matrix will be a poor estimate of the camera's imaging function. In [7], Fitzgibbon illustrates the significant risk of systemic failure from poor distortion compensation.

A radial distortion models the lens distortion as a monotonic function $u' \leftrightarrow \mathcal{L}(u, c)$ which moves image points u nearer to or farther from some center of distortion c in the image plane. Many methods have been devised to estimate the distortion $\{\mathcal{L}, c\}$, including [7, 20, 12] among others. The method described in [18] is a good match to this project since it does not require any manual interaction or the introduction of a calibration object. This algorithm uses the property that straight lines will remain straight when imaged by a pinhole camera, but depends on the constraint that most long contours detected in the imagery are generated by straight lines. This constraint is often satisfied for cameras on ground vehicles in man-made environments.

Estimation begins by extracting image contours which are likely to be the images of straight real-world objects. Canny edge detection is employed, followed by the removal of contours which are too short or have strong estimated curvature. Points are estimated with sub-pixel precision along each image contour by fitting a 4th degree polynomial to each. The inverse distortion is applied, and each resulting undistorted contour is measured for straightness by calculating the residual error of a linear approximation. The total error $E_{\mathcal{L}}$ is estimated as the sum of the residuals weighted by the contour lengths.

In the formulation used in these experiments, $\mathcal{L}$ is parameterized as as its 4^{th} degree Taylor polynomial with the constraints that $\mathcal{L}(0) = 0$ and $\mathcal{L}(1) = 1$. This means that the undistorted image remains approximately the same size as the original, and c is assumed to be the image center. Optimization proceeds by minimizing $E_{\mathcal{L}}$ via sequential quadratic programming as implemented in MATLAB's optimization toolbox.

Fig. 3 illustrates an example undistortion from cam1. Note the telephone pole and the building edge are no longer bent inwards. Fig. 4 plots three separate estimates of $\mathcal{L}$ for each camera. Differences are accentuated by plotting the difference between $\mathcal{L}(r)$ and r. Each estimate was computed from different, nonoverlapping sections of the data. It is easy to see that the radial distortion is estimated robustly, and, as expected, the wide-angle cameras have much larger distortions from linearity than does the longer focus camera.

5. Limiting R via Epipolar Geometry

In order to begin to estimate the camera projection matrix $\mathbf{P}$, we make use of a simple association between the motion direction and the image epipoles. It is well known that given two projective cameras, $\mathbf{P}$ and $\mathbf{P}'$, the image epipoles, $\mathbf{e}$ and $\mathbf{e}'$, are the points corresponding to the projections the camera centers. Furthermore, if $\mathbf{R} = \mathbf{R}'$ and $\mathbf{K} = \mathbf{K}'$, i.e., a purely translational motion of a constant camera, then the epipoles are co-located $(\mathbf{e} = \mathbf{e}')$ and they correspond to the direction of movement.

If some point $\mathbf{X}$ projects to $\mathbf{x}$ in camera $\mathbf{P}$, then $\mathbf{x}' = \mathbf{P}'\mathbf{X}$ is constrained by the geometry to project to some point on the epipolar line connecting $\mathbf{e}$ and $\mathbf{x}$. This leads to a simple method of recovering the epipole by determining the point of intersection of all lines connecting corresponding points in the two images. Of course, due to noise and errors not all lines will converge to a single point, but one may find a suitable approximation via a least squares fit. Fig. 5 illustrates a set of tracked Harris corners. The estimated epipole is just off the left edge of the image. We

Figure 3. Distorted and automatically undistorted image pair.

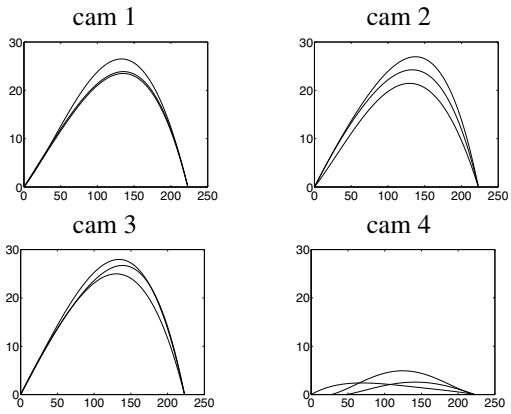

Figure 4. The distortion function $\mathcal{L}$ was estimated three times for each camera using non-overlapping sequences of 7.5 seconds each. Measurements are in pixels. The x axis plots r, the distance from the distortion (image) center. The y axis plots the difference between the undistorted radius and image radius $(r - \mathcal{L}(r))$. Distortion estimates are stable to within a few pixels.

build an accurate estimate of the epipole by tracking points for a short sequence of frames while the vehicle undergoes nearly translational motion.

$$\begin{bmatrix} e_1 \\ e_2 \\ 1 \end{bmatrix} \propto \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} R \begin{bmatrix} x_L \\ y_L \\ z_L \end{bmatrix} \tag{4}$$

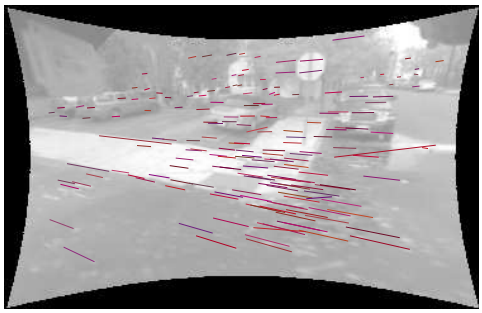

Figure 5. Image features are tracked to associate the image epipole with the translational motion direction. This association limits the space of rotation matrices.

Associating the image point $\mathbf{e}$ with the motion direction $\mathbf{D} = [x_L, y_L, z_L, 0]^\mathsf{T}$ limits the space of rotations to a two parameter family controlled by f and a free rotation around the axis $\mathbf{D}$. Eq. 4 encapsulates the situation as a set of 3 equations with 4 unknowns (3 rotation parameters plus f), but only 2 of the equations are linearly independent. The camera position $\tilde{\mathbf{C}}$ drops out of the equation. Determining $\mathbf{R}$ and f (and hence $\mathbf{K}$) is the subject of the next section.

6. Focal Length from Reconstruction and Reprojection

The next step in the calibration process to to find close approximations to the $\mathbf{K}$ and $\mathbf{R}$ matrices. This is accomplished using a technique similar to bundle adjustment [10, 6]. Image features are tracked from frame to frame, building a library of tracked points. If $\mathbf{x}_j^i$ is the projection of the j-th point at time i, then the goal is to estimate the 3D points $\mathbf{X}_j$ and camera matrices $\mathbf{P}^i$ minimizing the euclidean reprojection error E_1 over visible points.

$$E_1 = \sum_{(i,j)} (b(i,j) \times \|(\mathbf{P}^i \mathbf{X}_j - \mathbf{x}_j^i)\|_{L2}) \tag{5}$$

where $b(i,j)$ is a binary indicator variable which is one if and only if point j was detected in image i. Many variants of bundle adjustment exist, as well as tools to solve them accurately and with minimal computational cost.

In our problem we have the additional constraint that the projection matrices must meet the condition of Eq. 4. For now, the unknown camera positions $\tilde{\mathbf{C}}$ are set to be co-located with the lidar positions in world coordinates, using the transforms from the IMU. Similarly, the camera matrices are modified by the IMU rotation in world coordinates. If $\mathbf{T}_i$ is the euclidean IMU transform corresponding to $\mathbf{P}_i$, then:

$$\mathbf{T}_i = \begin{bmatrix} \mathbf{R}_{\mathbf{IMU}} & \tilde{\mathbf{C}}_{\mathbf{IMU}} \\ \mathbf{0} & 1 \end{bmatrix} \tag{6}$$

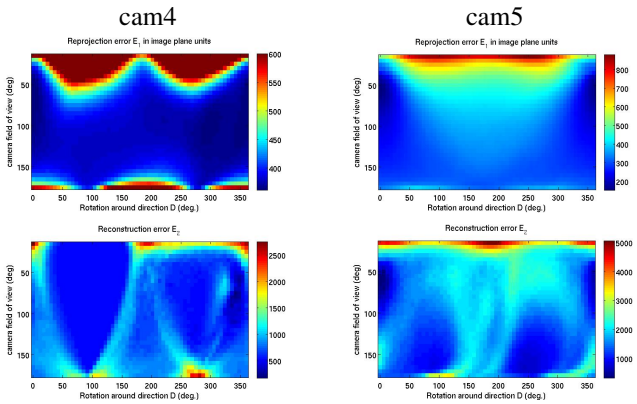

Figure 6. *TOP*. Illustration of the reprojection error E_1 for two cameras as a function of the camera focal length f and the camera rotation around the motion direction D. *BOTTOM*. Illustration of the reconstruction error E_2. The reprojection error E_1 is smooth, and usually contains a minimum near to a good camera estimate. The reconstruction error E_2 usually has a finer, more accurate minimum, but is harder to find quickly due to the non-convexities. Finding an E_1 min helps in quickly finding the E_2 min. Cam 4 min is at $(355, 80)$, Cam5:$(10, 50)$.

Figure 7. Example alignment of Cam2 using the reprojection / reconstruction technique of Sec. 6 but before discontinuity matching. The rotation and field of view are getting close, but more refinement is necessary to make use of the fused data. For illustration purposes, only the depth-discontinuous lidar locations are shown. This technique is described in Sec. 7

$$\mathbf{P^i} = \mathbf{K}\mathbf{R}_{\mathbf{IMU}}\mathbf{R}[\mathbf{I}| - \tilde{\mathbf{C}}_{\mathbf{IMU}}] \tag{7}$$

The top row of Fig. 6 illustrates E_1 for a two parameter family of cameras, variant over f and rotations around the direction $\mathbf{D}$. The set of camera matrices was generated using a full sweep over the free rotation in increments of $5°$. This mode of variation is represented by the x axis in the plots in Fig. 6. f is varied corresponding to a camera field of view of $15°$ to $175°$ in increments of $5°$. This mode of variation is represented by the y axis.

The lidar data presents the opportunity of using a second, different error measure. Rather than measuring the error in reprojection in the image, one could measure the error in the reconstruction in the lidar. Each reprojected image point $\hat{x}^i_j = P^i X_j$ is associated with a ray in world coordinates and a distance. The direction can be used to index into the lidar scan to obtain an expected distance measurement. The lidar distance can be compared to the reconstruction distance to build a lidar reconstruction error E_2.

$$E_2 = \sum_{(i,j)} (b(i,j) \times \tau(g(\|X_j - \tilde{C}^i\|_{L2}, l^i_j), \lambda)) \tag{8}$$

$$g(a,b) = max(\frac{a}{b}, \frac{b}{a}) \tag{9}$$

$$\tau(x,\lambda) = max(-1, 1 - \frac{x}{\lambda}) \tag{10}$$

where l^i_j represents the measured lidar distance in the direction of the reprojected point, $\hat{x}^i_j$. Similar to E_1, $b(i,j)$ is an indicator variable, but now indicates whether the direction corresponding to $\hat{x}^i_j$ is measured in *lidar* scan l^i_j. The maximum ratio operator g is used instead of the absolute distance so that relatively small errors from distant points do not overwhelm gross errors at near points. The tolerance function τ is very important to include so that the optimization algorithm is not rewarded or punished too strongly for moving tracked points outside of the field of view of the lidar. The parameter λ effectively sets a penalty so that inputs (distance ratios) less than λ are rewarded and ratios greater than λ are punished. λ was set to 3 for all experiments below.

The reprojection error E_1 is usually smoother over the two parameters, and contains only a few local minima, but the minima are shallow and broad. E_2 has a tighter and deeper minimum at the appropriate parameter settings, but requires a good initialization point to find it. In the experiments below, the E_1 measure is used to rapidly find a good initialization point, and then the E_2 measure is minimized to determine the camera matrix. Optimization is computed efficiently using the MATLAB optimization toolset.

The minimization of the E_1 or E_2 error measure gives a solution to all of the camera parameters except for the physical offset of the camera from the lidar sensor. A typical result is shown in Fig. 7. The alignment is close but suffers from the lack of freedom in the position parameters. For cameras mounted very close to the lidar, or with a very long focal lengths, this may not be a problem. In the next section a solution will be described which solves for the offset and refines the previously estimated camera parameters.

7. Refinement by discontinuity association

In this section, the camera projection matrix is refined using associations between analogous 2D and 3D contours. Contours along major discontinuities are detected in both modalities and the alignment of these is optimized via a search over the local camera parameter space. The brute force search (tweaked via hill-climbing) was selected rather than a more elegant solution only after several attempts to produce robust direct associations between 3D and 2D points were deemed unsuccessful. Solving for the camera projection matrix via discrete linear transform [13] or bundle adjustment proved to be too unstable given the high error rate in the contour association. The slower, more stable search over the parameter space achieved greater improvement in automatically estimating the camera parameters. The process and results are detailed below.

Discontinuities in the image data are detected with a modified canny edge detection algorithm. Edge detection takes place in the original images, and then the detections are warped into the distortion free coordinate space. Each edge pixel is labeled with its orientation and local curvature, and edges which are too short or bent are removed.

The lidar scan is processed similarly. For each time t synchronous with an image, the lidar is represented as a range image, as in Fig. 2, and depth discontinuous contours are detected in a way much like the process in [19]. Median filtering is used to fill holes up to a certain size, and contours are detected separately for depth discontinuities in the 4 cardinal directions, i.e., left-edges of objects are treated separately from right-edges. Contour points are joined together using the image 8 neighbor pattern and contours with fewer than 8 points are discarded. Care is taken such that the nearer side of the discontinuity is recorded as the contour. The output of this process is a graph of 3D points for each time t.

To associate the points in the two modalities, 3D contour points are projected into the images using the estimated camera parameters, as shown in Fig. 7. Each projected *lidar* contour point is associated with its nearest *image* contour point. Nearest neighbors are calculated using the L_2 distance with each point represented by both its image position and the sin and cos of its orientation. The orientation features are weighted by a value σ such that proximity in position is weighted more heavily than proximity in orientation. In our experiments σ is set to 0.2. Nearest neighbors are calculated efficiently using KDtrees. Since not all depth discontinuities will be visible optically, a robust heuristic is used, taking the sum of the smallest 50% of the distances to neighbors as the metric of projection quality.

This heuristic is optimized over a small range of the camera parameter space in the vicinity of the estimate from Sec. 6. A reasonable range of $\tilde{\mathbf{C}}$ was searched over which would leave the camera center within the volume of the vehicle. f was allowed to vary by 30%., and the camera rotation was allowed to change by 7° in roll, pitch, and yaw.

Figure 8. Illustration of an association between image and lidar discontinuities before optimization. The lidar discontinuities are projected into the image plane and colored by their associated orientation. The image discontinuities are drawn lightly over the image. Minimizing the distance covered by these associations optimizes the projection matrix (Sec. 7).

The results of a complete optimization calculated for a 330 frame sequence are illustrated in Fig. 9. Common failure modes of the calibration system include scenes with little or no depth discontinuity, such as a straight plane or blank wall, or far too much depth discontinuity, such as when looking through a fence or passing by dense foliage. Optimization seems to fall into a bad local minimum when fewer than about 5 frames are used to collect the discontinuity associations.

8. System Performance

The process detailed in this work involves multiple stages of refinement. A user may not be interested in applying later stages if the additional calibration accuracy is not worth the additional computational cost. Cost estimates are for non optimized MATLAB code running on a 2GHz machine with 4GB of RAM. There is reason to believe that these costs could be reduced significantly with optimized C code or via parallel processing The costs of each stage are variable depending upon the complexity of the data. Images with many discontinuities can cost multiple times more than others. Computation cost also increases linearly with number of frames included in the calibration process.

It is regrettable that the different model of radial distortion used in the hand calibration prevents the direct parameter comparison with this calibration. Adopting that formulation to produce numeric accuracy assessments of the subsequent stages is the subject of further work. We refer the reader to the Fig. 9 and the digital addendum for typical empirical results across all 4 cameras.

Stage	Cost (sec)
Epipole Motion Alignment	84
Reprojection and Reconstruction	245
Discontinuity Matching	2464

Table 1. Table of approximate computation cost for calibration stages. Code is un-optimized MATLAB on a 2GHz machine with 4GB ram for a 330 frame sequence.

9. Conclusions

The system described herein is a proof of concept of a fully automatic algorithm for associating optical video feeds to lidar feeds on a moving vehicle. This technique involves multiple stages of increasingly refined processing, and may be suited to maintaining the calibration of feeds on systems for which re-calibration is necessary but hand calibration is not an option, such as robots in the field or for associating data feeds in a live pan-tilt-zoom camera. While the alignment is not as stable as that which has been shown from hand calibrated techniques or techniques requiring dedicated calibration objects, this work shows that a hight level of data association is possible entirely automatically. This association guarantee may be enough for upstream processes to proceed on fused data.

References

[1] T. Asai, M. Kanbara, and N. Yokoya. 3D modeling of outdoor environments by integrating omnidirectional range and color images. In *3-D Digital Imaging and Modeling, 2005.*, pages 447–454, 2005. 1

[2] J. Bauer, K. Karner, K. Schindler, A. Klaus, and C. Zach. Segmentation of building models from dense 3D point-clouds. In *Proc. 27th Workshop of the Austrian Association for Pattern Recognition*, pages 253–258, 2003. 1

[3] X. Brim and F. Goulette. Modeling and calibration of coupled fish-eye CCD camera and laser range scanner for outdoor environment reconstruction. In *3-D Digital Imaging and Modeling, 2007. 3DIM'07. Sixth International Conference on*, pages 320–327, 2007. 1

[4] F. Deng, M. Hu, and H. Guan. Automatic Registration Between LIDAR and Digital Images. In *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pages 487–490, 2008. 1

[5] P. Dias, V. Sequeira, F. Vaz, and J. Gonçalves. Registration and fusion of intensity and range data for 3D modelling of real world scenes. In *Proc. of the 4th IEEE Int. Conf. on Recent Advances in 3D Digital Imaging and Modeling (3DIM03), Banff, Canada*, 2003. 1

[6] C. Engels, H. Stewenius, and D. Nister. Bundle adjustment rules. *Photogrammetric Computer Vision*, 2006. 4

[7] A. Fitzgibbon. Simultaneous linear estimation of multiple view geometry and lens distortion. In *Computer Vision and Pattern Recognition, 2001*, volume 1, 2001. 3

[8] A. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed or open image sequences. *Lecture Notes in Computer Science*, 1406:311–326, 1998. 2

[9] C. Frueh, R. Sammon, and A. Zakhor. Automated texture mapping of 3D city models with oblique aerial imagery. In *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, pages 396–403, 2004. 1

[10] Y. Furukawa and J. Ponce. Accurate camera calibration from multi-view stereo and bundle adjustment. In *CVPR 2008*, pages 1–8, 2008. 4

[11] A. Habib, M. Ghanma, and M. Tait. Integration of lidar and photogrammetry for close range applications. In *International Archives of 20th ISPRS Congress. Commission I. Istanbul/turkey, 2004c*, 2004. 1

[12] R. Hartley and S. Kang. Parameter-free radial distortion correction with centre of distortion estimation. In *ICCV*, volume 2, 2005. 3

[13] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge Univ Pr, 2003. 6

[14] J. Leonard, J. How, S. Teller, M. Berger, S. Campbell, G. Fiore, L. Fletcher, E. Frazzoli, and etal. A perception-driven autonomous urban vehicle. *Journal of Field Robotics*, 25(10):727–774, 2008. 1, 2

[15] N. Naikal, J. Kua, and A. Zakhor. Image Augmented Laser Scan Matching for Indoor Localization. *UC Berkeley Technical Report*, 2009. 1

[16] M. Pollefeys, R. Koch, and L. Gool. Self-calibration and metric reconstruction inspite of varying and unknown intrinsic camera parameters. *IJCV*, 32(1):7–25, 1999. 2

[17] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, and J. Tops. Video-to-3D. *International Archives Of Photogrametry Remote Sensing And Spatial Information Sciences*, 34(3/A):252–257, 2002. 2

[18] B. Prescott and G. McLean. Line-based correction of radial lens distortion. *Graphical Models and Image Processing*, 59(1):39–47, 1997. 3

[19] S. Stamos, L. Liu, C. Chen, G. Wolberg, G. Yu, and S. Zokai. Integrating Automated Range Registration with Multiview Geometry for the Photrealistic Modeling of Large-Scale Scenes. *IJCV*, 78:237–260, 2008. 1, 6

[20] R. Steele and C. Jaynes. Overconstrained linear estimation of radial distortion and multi-view geometry. *LCNS*, 3951:253, 2006. 3

[21] C. Strecha, W. von Hansen, L. Van Gool, and U. Thoennessen. Multi-View Stereo And Lidar For Outdoor Scene Modeling. *Photogrammetric Image Analysis*, 2007. 1

[22] B. Triggs. Autocalibration and the absolute quadric. In *CVPR*, pages 609–614, 1997. 2

[23] H. Zhao and R. Shibasaki. Reconstructing urban 3D model using vehicle-borne laser range scanners. In *Proceedings of the International Conference on 3D Digital Imaging and Modeling*, pages 349–356. Citeseer, 2001. 1

[24] W. Zhao, D. Nister, and S. Hsu. Alignment of continuous video onto 3D point clouds. *IEEE transactions on pattern analysis and machine intelligence*, 27(8):1305–1318, 2005. 1

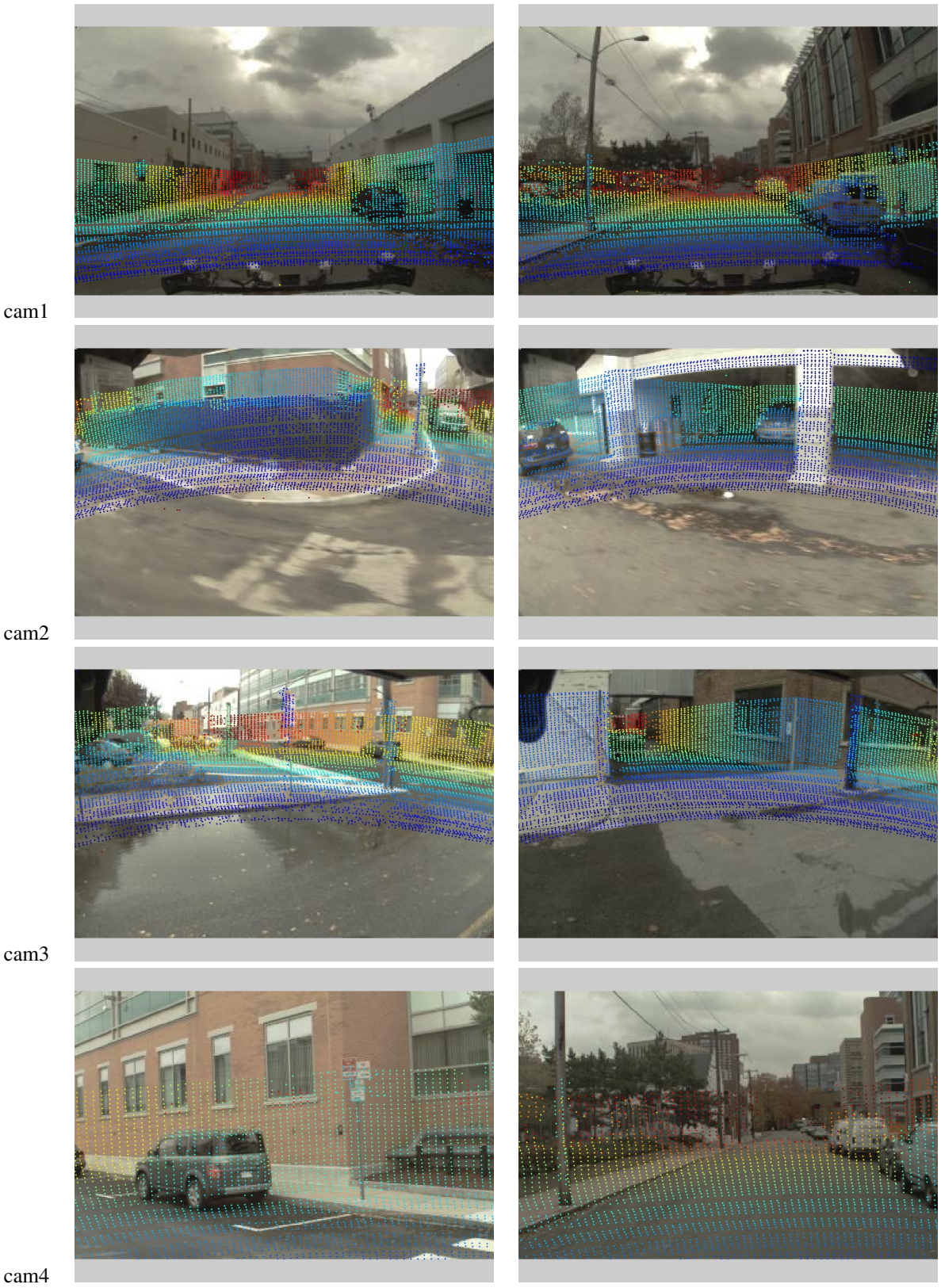

Figure 9. Illustration of the results of camera refinement via matching contours. Results are significantly improved over those from Sec. 6. Note that no manual intervention is involved in either method. Note in particular the accuracy of the telephone poles and building edges. Videos are available in the digital addendum.