2011 IEEE International Conference on Robotics and Automation
Shanghai International Conference Center
May 9-13, 2011, Shanghai, China

Automatic Alignment of a Camera with a Line Scan LIDAR System

Oleg Naroditsky, Alexander Patterson IV and Kostas Daniilidis

***Abstract*— We propose a new method for extrinsic calibration of a line-scan LIDAR with a perspective projection camera. Our method is a closed-form, minimal solution to the problem. The solution is a symbolic template found via variable elimination and the multi-polynomial Macaulay resultant. It does not require initialization, and can be used in an automatic calibration setting when paired with RANSAC and least-squares refinement. We show the efficacy of our approach through a set of simulations and a real calibration.**

I. INTRODUCTION

The problem of calibrating a LIDAR-camera sensor rig is very important in robotics applications. A camera provides a dense, color image of the environment, and LIDAR gives a sparse, but accurate, depth map. Fusion of visual and distance information is challenging when there is parallax between the two sensors and when distance consists of a single line scan. Unlike in structured light techniques, the laser is not visible in the image, hence we have to invent a way to associate features on the line scan with features in the image if we want to eliminate a manual selection of the features.

This paper gives a complete solution to this problem, allowing automatic, initialization-free calibration of the sensors, assuming only overlapping fields of view. We demonstrate via a series of synthetic and real experiments that our method, which is based on minimal solutions and RANSAC [1], is numerically stable and accurate for realistic calibration scenarios.

Our method relies on automatically computed correspondences between lines in the image and 3D points returned by the LIDAR. We formulate and solve the "minimal problem" using these correspondences. We show that the minimum number of constraints is six, and that there are at most four distinct solutions. We derive a closed-form solution to the problem using variable elimination and resultants.

II. RELATED WORK

The closest and most cited work to ours is by Zhang and Pless [2] who match a scanline to a checkerboard. When moving a checkerboard, traditional camera calibration [3] can extract the normal to the checkerboard with respect to a global camera reference, while detection of a line in the laser profile enables association of 3D-points with the calibration plane. The algorithm starts with a linear initialization, suffering under the well known effects of linearization like finding 3x3 matrices satisfying the data equation and then finding the closest special orthogonal matrix. Mei and Rives [4] have applied the same principle to catadioptric images but they exploit the association of a 3D-line (in terms of direction and an offset) to a calibration plane. It is worth noticing that the equation associating the plane normal to the 3D-line direction is of the form $\mathbf{n}^\top R\mathbf{d} = 0$, is algebraically the same as ours after eliminating the translation. However, the authors use, similarly to [2], the association of points.

The authors are with the GRASP Laboratory, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA. {narodits,aiv,kostas}@cis.upenn.edu

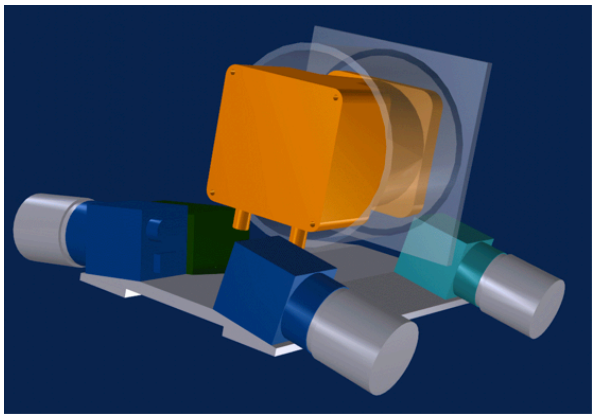

Fig. 1. A capture rig incorporating four cameras and a Hokuyo LIDAR. Our method is intended to automatically calibrate such systems.

When a laser system produces a full depth map at once, the only challenge is associating features. In [5], a similar to [2] association of 3D points to camera planes is used. In [6] an IMU enables the registration of line scans into a 3D LIDAR and the relative transformation is found via hand-eye calibration [7]. Scaramuzza [8] uses the association of hand-clicked points in a full 3D map with points in catadioptric images.

When we say we solve the "minimal problem", we mean a geometric problem that uses the lowest number of constraints required to obtain a finite number of solutions. When paired with RANSAC, minimal problems are optimally suited automatic model estimation from noisy data with outliers, since they minimize the probability of choosing an outlier as part of the model. Consequently, minimal problems in computer vision have received a great deal of attention, and our work follows the algorithms beginning with the five-point solution to the structure-from-motion problem [9], problems in 3D reconstruction [10] and camera calibration [11].

III. PROBLEM DESCRIPTION

Let us formally define the problem of camera-to-LIDAR calibration. We are given a rig consisting of a calibrated camera (intrinsic parameters are known) and scan line LIDAR that are rigidly mounted with respect to each other.

978-1-61284-385-8/11/$26.00 ©2011 IEEE

Our goal is to find the rigid transformation $[R|\mathbf{t}]$ such that given a 3D point $\mathbf{x} = [x_1, x_2, x_3]^\top$ obtained by the LIDAR, we can compute the corresponding point $\mathbf{y} = [y_1, y_2, y_3]^\top$ in the camera's coordinate system (and then in the image via the intrinsic calibration) as $\mathbf{y} = R\mathbf{x} + \mathbf{t}$.

A single LIDAR datum consists of a depth and angle at which the depth was sensed. We define the coordinate system for the LIDAR as follows. The origin is the center of laser sensor rotation, and the plane of laser rotation is the Y-Z plane. Consequently, we can always express a 3D LIDAR point as $\mathbf{x} = [0, x_2, x_3]^\top$.

As with any calibration problem from sensor data, we must collect correspondences between the readings of different sensors. In this case, we construct a calibration target containing a single black to white transition (see Figure 2), which we detect as a line segment in the image and a point in the LIDAR's luminance output for a single line scan (see Figure 3). We discuss feature detection in detail in Section VI-B. A line in the image corresponds to a plane in the world containing the line and the center of projection of the camera. We now have a correspondence between a 3D point in the LIDAR coordinate system and a plane in the camera's coordinate system. Thus our constraint is that the LIDAR point, taken into the camera's coordinate system, must lie on the corresponding plane. We express this constraint as

$$\mathbf{n}^\top (R\mathbf{x} + \mathbf{t}) = 0, \tag{1}$$

where $\mathbf{n}$ is the normal to the plane in the camera coordinate system and $\mathbf{x}$ is the LIDAR point.

Fig. 2. A single camera frame from the calibration data set showing the calibration object. The object consists of a black line on a white sheet of paper. We detect the white-to-black transition looking from the top of the image.

IV. THE POLYNOMIAL SYSTEM

In this section we show how to formulate the problem as a set of polynomial equations. Our original point correspondence constraints have the form

$$\mathbf{n}_i^\top (R\mathbf{x}_i + \mathbf{t}) = 0 \tag{2}$$

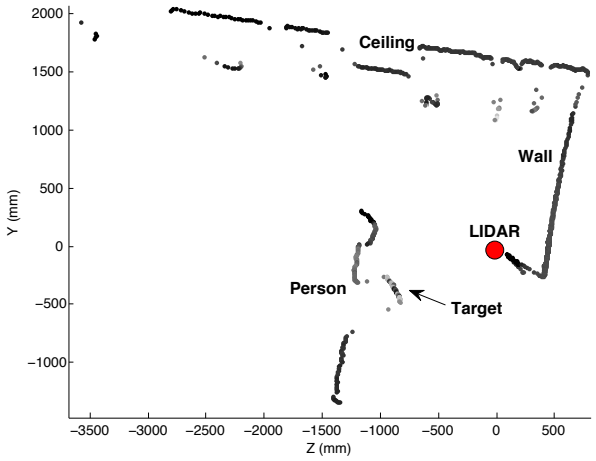

Fig. 3. A portion of a LIDAR scan showing a person holding the calibration target. The points are colored by their intensity returns. The LIDAR's scan plane is close to vertical, and its origin is marked by a circle.

for $i = 1, \ldots, 6$, where $R \in \mathbf{SO}(3)$ and $\mathbf{t}$, $\mathbf{n}_i$, $\mathbf{x}_i$ and $\mathbf{t}$ are 3-vectors. Let $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$ be the columns of R. Since we set up the LIDAR coordinate system in such a way that the first component of each $\mathbf{x}_i$ is zero, we do not have an explicit dependence on $\mathbf{r}_1$. By taking the cross product

$$\mathbf{r}_1 = \mathbf{r}_2 \times \mathbf{r}_3, \tag{3}$$

we can recover the first column of R from the other two. Since R is orthonormal, we can add the following constraints

$$\begin{aligned} \mathbf{r}_3^\top \mathbf{r}_3 &= 1 \\ \mathbf{r}_2^\top \mathbf{r}_2 &= 1 \\ \mathbf{r}_3^\top \mathbf{r}_2 &= 0. \end{aligned} \tag{4}$$

We expand the constraints in equation (2), to obtain linear equations of the form

$$\begin{bmatrix} \mathbf{a}_i^\top & \mathbf{a}_i'^\top \end{bmatrix} \begin{bmatrix} \mathbf{r}_2 \\ \mathbf{r}_3 \end{bmatrix} + \mathbf{n}_i^\top \mathbf{t} = 0 \tag{5}$$

for $i = 1, \ldots, 6$, where the constant vectors $\mathbf{a}_i$ and $\mathbf{a}_i'$ are expressed in terms of input data in the following way:

$$\begin{bmatrix} \mathbf{a}_i \\ \mathbf{a}_i' \end{bmatrix} = \begin{bmatrix} \mathbf{n}_i x_{i2} \\ \mathbf{n}_i x_{i3} \end{bmatrix}. \tag{6}$$

Our system now has 6 linear, homogenous equations in 9 unknowns corresponding to the point-to-plane correspondences, and three second order equations constraining the rotation matrix. In the next two sections we will present a closed-form solution to this polynomial system.

V. THE CLOSED-FORM SOLUTION

Since we have linear constraints (2), and require a six-degree-of-freedom solution, it is not surprising that we need a minimum of six correspondences. It may seem surprising at first, however, that this polynomial system can be solved in closed-form, despite having eight solutions. First, we briefly outline the following steps to eliminate eight of the variables

from the system, solve a univariate polynomial and then back substitute:

1) Symbolically solve for the translation component $\mathbf{t}$ using the first three linear equations, and substitute into the three remaining linear equations.
2) Symbolically solve three of the six remaining equations for the components of $\mathbf{r}_2$ and substitute into remaining equations.
3) Use the Macaulay resultant (see Chapter 7, §6 in [12]) to symbolically eliminate two of the three remaining variables from the three remaining equations, and solve the resulting quartic equation in closed form for r_{33}.
4) Substitute the solutions into the three equations, and use the Sylvester resultant (see Chapter 3, §5 in [12]) of two of them to eliminate one of the two remaining variables, and solve the resulting quartic equation in closed form.
5) Test the solutions to obtain the one satisfying all equations.
6) Back-substitute for the variables in $\mathbf{r}_2$ and $\mathbf{t}$.

The symbolic templates generated by this process will remain the same across all instances of the problem, so they only need to be computed once, so that solving an instance of a problem can be accomplished by substitution of the data into the expressions for the solution.

We will now describe the solution in detail. Let us first eliminate $\mathbf{t}$ from the first three constraints of the form (2). These are linear equations, so we can express $\mathbf{t}$ as

$$\mathbf{t} = -\begin{bmatrix}\mathbf{n}_1^\top \\ \mathbf{n}_2^\top \\ \mathbf{n}_3^\top\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{n}_1^\top R\mathbf{x}_1 \\ \mathbf{n}_2^\top R\mathbf{x}_2 \\ \mathbf{n}_3^\top R\mathbf{x}_3\end{bmatrix}. \tag{7}$$

In the above system, the components of $\mathbf{t}$ are expressed in terms of the remaining variables $\mathbf{r}_2$ and $\mathbf{r}_3$. Let us call the coefficients of the remaining variables b_{ij}. The three equations in (7) then become

$$t_i = \begin{bmatrix}\mathbf{b}_i^\top & \mathbf{b}_i'^\top\end{bmatrix}\begin{bmatrix}\mathbf{r}_2 \\ \mathbf{r}_3\end{bmatrix}, \tag{8}$$

for $i = 1,\ldots,3$ which is linear in the the components of $\mathbf{r}_2$ and $\mathbf{r}_3$.

We now substitute for $\mathbf{t}$ in the three remaining constraints of the form (2) for $i = 4,\ldots,6$ to get three linear constraints in $\mathbf{r}_2$ and $\mathbf{r}_3$.

$$\begin{bmatrix}\mathbf{c}_4^\top \\ \mathbf{c}_5^\top \\ \mathbf{c}_6^\top\end{bmatrix}\mathbf{r}_2 + \begin{bmatrix}\mathbf{c}_4'^\top \\ \mathbf{c}_5'^\top \\ \mathbf{c}_6'^\top\end{bmatrix}\mathbf{r}_3 = 0, \tag{9}$$

where c_{ij} and c'_{ij} are the coefficients (expressed once again entirely in terms of problem data)

$$\begin{aligned}\mathbf{c}_i &= \mathbf{a}_i + \begin{bmatrix}\mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3\end{bmatrix}\mathbf{n}_i \\ \mathbf{c}'_i &= \mathbf{a}'_i + \begin{bmatrix}\mathbf{b}'_1 & \mathbf{b}'_2 & \mathbf{b}'_3\end{bmatrix}\mathbf{n}_i.\end{aligned} \tag{10}$$

Let C and C' be the 3×3 matrices of coefficients c_{ij} and c'_{ij}. Using the linear system (9), we can express $\mathbf{r}_2$ as

$$\mathbf{r}_2 = -C^{-1}C'\mathbf{r}_3. \tag{11}$$

We can now eliminate $\mathbf{r}_2$ from the system by substituting equation (11) into the three second order constraints (4), arriving (after full expansion) at the system

$$r_{13}^2 + r_{23}^2 + r_{33}^2 = 1 \tag{12}$$

$$\begin{aligned}e_{11}r_{13}^2 + e_{12}r_{13}r_{23} + e_{13}r_{23}^2 + e_{14}r_{13}r_{33} \\ + e_{15}r_{23}r_{33} + e_{16}r_{33}^2 = 1\end{aligned} \tag{13}$$

$$\begin{aligned}e_{21}r_{13}^2 + e_{22}r_{13}r_{23} + e_{23}r_{23}^2 + e_{24}r_{13}r_{33} \\ + e_{25}r_{23}r_{33} + e_{26}r_{33}^2 = 0,\end{aligned} \tag{14}$$

where the constants e_{ij} are computed in closed form in terms of entries of C^{-1} and C' after the substitution. These coefficients are given explicitly in the supplemental material[1]. We can observe that given a set of values that satisfy this system, the negative values will also satisfy it. Thus, we expect to solve this system via a quartic polynomial in squares of the variables instead of an eighth degree.

While we cannot directly solve for any of the variables in the system formed by the last three equations, nor can we reduce the number of variables further by rearranging the system, we can still obtain a univariate polynomial in r_{33} by using the Macaulay resultant of the three equations. This multivariate resultant is the quotient of the determinants of the numerator matrix (17) and the denominator matrix

$$\begin{bmatrix}e_{13} & 0 & e_{16} \\ 0 & e_{13} & e_{11}r_{13}^2 - 1 \\ 1 & 0 & 1\end{bmatrix} \tag{15}$$

When we set this resultant to 0, we obtain the following polynomial equation:

$$g_1 r_{33}^8 + g_2 r_{33}^6 + g_3 r_{33}^4 + g_4 r_{33}^2 + g_5 = 0. \tag{16}$$

The coefficients g_i can be computed in closed form using a symbolic mathematics program, such as Maple [13]. They only have seven thousand terms due to the sparsity of the matrix (17), and are explicitly given in the supplemental material.

The equation (16) can be solved in closed form as a quartic equation in r_{33}^2, thus giving us up to four real solutions and eight possibilities for r_{33}. We note that since both $[R|\mathbf{t}]$ and $[-R|-\mathbf{t}]$ are the solutions to the original system, we only need to back-substitute the positive, real solutions of (16), and there are up to four of them. Once we obtain the corresponding $[R|\mathbf{t}]$, simply choose the sign of the overall solution that makes the determinant of R positive.

While we know that each solution for r_{33} corresponds to a single solution for the rest of the variables in the original system, when we substitute the numeric values $\tilde{r}_{33}$ in the equations (13), (14), and (12), we are faced with a system which may have false solutions due to numerical error. While it is possible to substitute the closed-form solutions to the quartic, it is not practical due to the radicals in the quartic formula. Instead, we compute the Sylvester

[1]The MATLAB function, which includes all coefficients, can be obtained from www.cis.upenn.edu/~narodits/SolveLidarCalibrationOpt.m.

$$\begin{bmatrix}
e_{11} & 0 & 0 & 0 & 0 & 0 & e_{13} & 0 & 0 & 0 & 0 & e_{16}r_{33}^2-1 & e_{15}r_{33} & e_{14}r_{33} & e_{12} \\
0 & e_{11} & 0 & 0 & 0 & 0 & e_{15}r_{33} & e_{13} & 0 & e_{12} & 0 & 0 & e_{16}r_{33}^2-1 & 0 & e_{14}r_{33} \\
0 & 0 & e_{11} & 0 & 0 & 0 & e_{16}r_{33}^2-1 & e_{15}r_{33} & e_{13} & e_{14}r_{33} & e_{12} & 0 & 0 & 0 & 0 \\
e_{14}r_{33} & e_{12} & 0 & e_{11} & 0 & 0 & 0 & 0 & 0 & e_{13} & 0 & 0 & 0 & e_{16}r_{33}^2-1 & e_{15}r_{33} \\
0 & e_{14}r_{33} & e_{12} & 0 & e_{11} & 0 & 0 & 0 & 0 & e_{15}r_{33} & e_{13} & 0 & 0 & 0 & e_{16}r_{33}^2-1 \\
e_{16}r_{33}^2-1 & e_{15}r_{33} & e_{13} & e_{14}r_{33} & e_{12} & e_{11} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & r_{33}^2-1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & r_{33}^2-1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & r_{33}^2-1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & r_{33}^2-1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & r_{33}^2-1 \\
e_{31} & 0 & 0 & 0 & 0 & 0 & e_{33} & 0 & 0 & 0 & 0 & e_{36}r_{33}^2 & e_{35}r_{33} & e_{34}r_{33} & e_{32} \\
0 & e_{31} & 0 & 0 & 0 & 0 & e_{35}r_{33} & e_{33} & 0 & e_{32} & 0 & 0 & e_{36}r_{33}^2 & 0 & e_{34}r_{33} \\
e_{34}r_{33} & e_{32} & 0 & e_{31} & 0 & 0 & 0 & 0 & 0 & e_{33} & 0 & 0 & 0 & e_{36}r_{33}^2 & e_{35}r_{33} \\
0 & e_{34}r_{33} & e_{32} & 0 & e_{31} & 0 & 0 & 0 & 0 & e_{35}r_{33} & e_{33} & 0 & 0 & 0 & e_{36}r_{33}^2
\end{bmatrix} \tag{17}$$

$$\begin{bmatrix}
e_{13} & e_{12}r_{13}+e_{15}\tilde{r}_{33} & e_{11}r_{13}^2+e_{16}\tilde{r}_{33}^2-1+e_{14}r_{13}\tilde{r}_{33} & 0 \\
0 & e_{13} & e_{12}r_{13}+e_{15}\tilde{r}_{33} & e_{11}r_{13}^2+e_{16}r_{33}^2-1+e_{14}r_{13}\tilde{r}_{33} \\
1 & 0 & -1+r_{13}^2+\tilde{r}_{33}^2 & 0 \\
0 & 1 & 0 & -1+r_{13}^2+\tilde{r}_{33}^2
\end{bmatrix} \tag{18}$$

resultant of (13) and (12) with respect to r_{23}, which is the determinant of the Sylvester matrix (18). The resultant is thus a quartic polynomial in r_{13}, and its coefficients are computed symbolically and are given in the supplemental material.

We set the resultant to 0 and obtain the real solutions $\tilde{r}_{13}$ to the equation. We substitute the values into (14) and solve for the remaining variable r_{23}. It now becomes a simple matter of testing the solutions using equation (12) to see which solution indeed lies on the sphere. We repeat this process with the remaining real solutions to (16).

Having obtained up to four sets of solutions for $\mathbf{r}_3$, it is simple to recover solutions for $\mathbf{r}_2$ from (9), $\mathbf{r}_1$ from (3), and $\mathbf{t}$ from (7). As we mentioned before, we must choose the sign of each solution to make the determinant of R positive. We can find the unique solution to the problem by using a 7^{th} correspondence to disambiguate among the four solution. To do this, we substitute the $\mathbf{n}_7$ and $\mathbf{x}_7$ into the constraint equation (2) for each candidate R and $\mathbf{t}$ and choose the solution with the lowest absolute residual.

The steps described above produce a symbolic template for this problem. By this we mean that given any particular set of six correspondences, we can arrive at the solutions for R and $\mathbf{t}$ only by substituting the data into pre-computed expressions and solving univariate polynomials (in this case of degree 4).

VI. RESULTS

In this section we will first establish the correctness of our algorithm and explore its sensitivity to noise using simulated data, and then perform a real calibration of a LIDAR-camera rig mounted on a mobile robot for the purpose of coloring the 3D LIDAR points with pixels from the camera and show the results.

A. Simulations

Our first experiment establishes the correctness and numerical stability of our symbolic template. We generate, uniformly at random, roll, pitch and yaw of the LIDAR with respect to the camera in the range of $\pm 30^\circ$, and translation vectors with uniformly random components from 0 to 30cm. For each of these ground truth calibrations, we generate six noise-free correspondences.

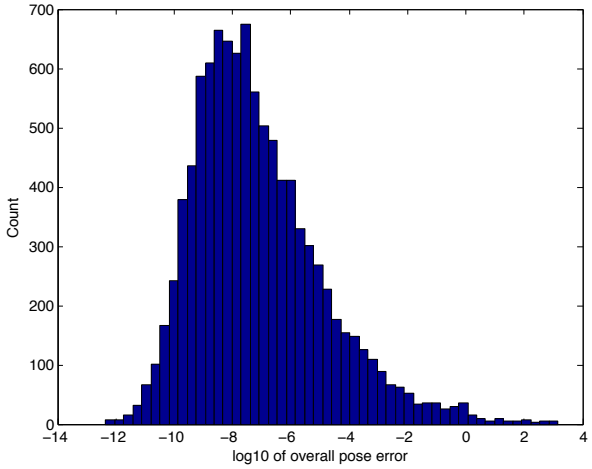

Fig. 4. The histogram of numerical errors for 10^5 random, noise-free instances of the problem. The error is defined as the $\log_{10} e$, where e of the Frobenius norm of the difference between the ground truth and the computed matrices (see (19)). Since the points were not checked for degeneracy (such as collinearity), some failures are observed. If we consider a failure to be $\log_{10} e > -1$, then the method fails 1.97% of the time.

We generate these correspondences by simulating a camera looking at a target. Specifically, we choose two random points in sampling volumes in front of the camera, one on the left and one on the right. This closely models what happens in the real system where the 3D line must intersect the LIDAR scan plane which is close to the vertical plane separating the left and right halves of the image. These two points define a line which is then intersected with the LIDAR plane to obtain the point $\mathbf{x}$. The points, along with the camera

center, define the plane and its normal $\mathbf{n}$.

The histogram of errors for 10^5 such configurations is shown in Figure 4. The error metric is the the following:

$$e = \min_i(\|[R_{gt}\,|\,\mathbf{t}_{gt}] - [R_i\,|\,\mathbf{t}_i]\|_{\mathscr{F}}), \tag{19}$$

for i from 1 to the number of solutions, R_{gt} and $\mathbf{t}_{gt}$ comprise the ground truth calibration and $\|.\|_{\mathscr{F}}$ is the Frobenius norm. The figure demonstrates that our solution correctly solves the calibration problem. The accuracy varies due to the random nature of correspondences and lack of degeneracy checking during sampling.

We now profile the algorithm with respect to noise in sensor data. We consider three sources of error: depth uncertainty in the LIDAR points and misestimation of position and rotation of the corresponding lines in the image, both of which lead to the 3D point being some distance off the plane through the line and center of the camera. For the LIDAR depth error, the Hokuyo UTM-30LX specifies the standard deviation of 30mm for ranges less than 1m. We will study the accuracy for noise standard deviations of 0mm to 30mm. The image processing accuracy will be in pixels with respect to a baseline calibrated camera, which we define as a 640×480 pixel sensor with a 60° field of view. While in the real data the errors in line extraction will depend on the length of the edge segments extracted, we will use the range of 0 to 4 pixel standard deviations in the simulated results. The results for different error levels are shown in Figure 5, and demonstrate a greater sensitivity to image noise than depth noise.

B. A Real Calibration

The sensor rig for the real-world calibration experiment consists of a calibrated 640×480 Flea2 camera with a 77° field of view and a Hokuyo UTM-30LX line scanner, with angular resolution of 0.25 deg (see Figure 1).

Images of the calibration target (see Figures 2 and 3) are captured synchronously by the two sensors at various positions. We detect the target as follows. For the LIDAR calibration target detection we use the line scan, including intensity, returned from the device in the region of interest for calibration. First, the derivatives of the intensity vector of the line scan are computed using a difference of gaussians filter. The peaks of this derivative signal are detected by non-maximal suppression. This detects both rising and falling edges in the intensity signal. We then improve the estimate of the edges by fitting a line to all the neighboring 3D points and projecting the intensity edge sample point onto that line to give us the final LIDAR feature point $\mathbf{x}_i$. The discrete sampling of the angle could cause an angular error, but since we can control the target's location during calibration, this error can be controlled. Even if the LIDAR and camera are further apart as on a larger robot, a long target can be used which could keep the target close to both sensors.

The first two steps in image line extraction are radial distortion removal and edge detection [14]. The edgels are then combined using the Hough transform to output line segments. We define "linescore" as is a measure of how well the gradient information in the image fits with the proposed line. We compute this on the line segments to orient them and prune out ones with poor support. In order to define linescore, first define Q_j as the set of pixels which lie within 1 pixel of the line segment j to score, and define $\mathbf{g}_i$ to be the gradient at pixel i, and $\mathbf{m}_j$ to be the normal to the line segment j which we are scoring. $linescore_j = \sum_{i \in Q_j} \mathbf{g}_i^\top \mathbf{m}_j$ measures the support in the gradient image for each line.

Next, the candidate edges are pruned more using the following heuristics, which use the fact that our target contains a single black stripe on a white background. The heuristics look for pairs of line segments which contain a white to black transition followed by a black to white transition. To do this, we look at the normal direction combined with the linescore to propose candidate pairings respecting this. For each candidate pairing we perform a number of tests:

1) Check that the candidate pair's normals are close to parallel while accounting for possible perspective distortions.
2) The lines are required to be close together because the target never fills up too much of the field of view.
3) Check that the ratio of width the height of the rectangle created by the pair of line segments is appropriate. This is a check that the target has the correct aspect ratio.
4) Overlap is computed and thresholded to rule out line segments which don't occur next to each other.

Once these criteria are passed we take the two endpoints of each line, and record the normal to the plane $\mathbf{n}_i$ passing through these points and the camera center. Thus the vector $\mathbf{n}_i$ corresponding to the LIDAR point $\mathbf{x}_i$ become the input for RANSAC process. We then compute a robust calibration solution using around 400 correspondences in the RANSAC framework. The process chooses the best calibration hypothesis based on six correspondences which we then iteratively refine using all of the inliers.

We tested the quality of our calibration. Our rig is equipped with stereo cameras that we calibrated using the Camera Calibration Toolbox [3]. We computed the error in the LIDAR-camera calibration by observing that we can compute the left-to-right camera calibration by combining the left camera and right camera to LIDAR calibrations. We define an error metric as follows. Let us denote a calibration transformation $P_q = \left[\begin{smallmatrix} R_q & \mathbf{t}_q \\ 000 & 1 \end{smallmatrix}\right]$. We can define the error matrix

$$P_{err} = P_{lr} P_{rh} P_{lh}^{-1},$$

where P_{lr} is the left-to-right calibration computed using the calibration toolbox, and P_{rh} and P_{lh} are the left camera and right camera to Hokuyo calibrations, respectively, computed with our method. If both of our calibrations were accurate, R_{err} would be close to identity and $\|\mathbf{t}_{err}\|_2$ would be close to 0. The real rotation error was 0.26°, and the translation error was 1.9mm (the distance between the LIDAR and each camera was approximately 140mm).

This calibration was used to color the LIDAR points with the corresponding pixels from the camera. Our system automatically acquired the images and registered LIDAR scans using visual odometry. The registered LIDAR point

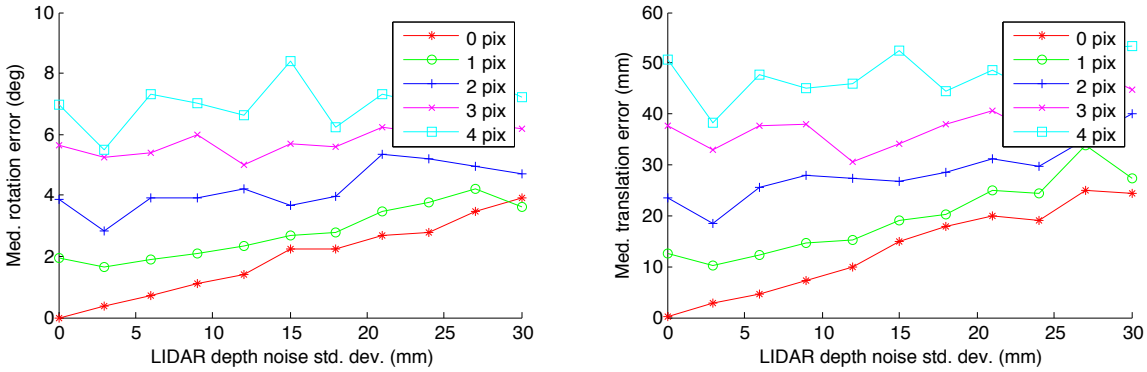

Fig. 5. Errors in rotation and translation estimation for a simulated rig with a 100mm distance between the camera and LIDAR. Each point shows median error for 200 random configurations of LIDAR-image correspondences. Each sequence corresponds to different levels of image noise plotted against LIDAR noise. The noise values are the standard deviations. The image errors are line translation error (pix) for the baseline camera described in Section VI-A.

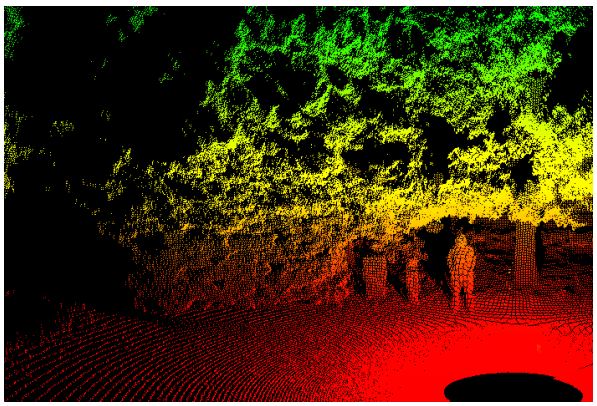

Fig. 6. A sample LIDAR scan acquired by the mobile robot colored by height.

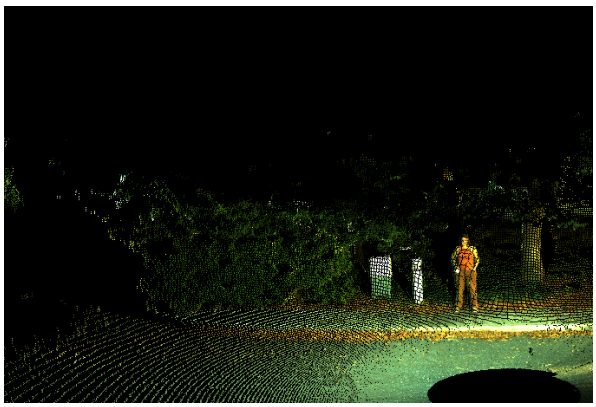

Fig. 7. A LIDAR scan colored using camera pixels.

cloud is shown in Figure 6. The same point cloud colored by the camera image pixels is shown in Figure 7.

VII. CONCLUSION

In this paper we overcome the difficulty of calibrating LIDAR-camera rigs by introducing a new algorithm based on the minimal solution to the calibration from line to 3D point correspondences, used in a robust framework and without initialization. Using automatic feature detection described, the algorithm can be used to automatically calibrate a variety of sensor platforms. Our experiments with simulated and real data indicate that this method is both correct and practical.

VIII. ACKNOWLEDGMENTS

The authors are grateful for support through the grants NSF-IIP-0742304, NSF-IIP- 0835714, ARL MAST-CTA W911NF-08-2-0004 and ARL RCTA W911NF-10-2-0016, and thank Raia Hadsell of SRI for creating Figures 6 and 7.

REFERENCES

[1] M. Fischler and R. Bolles, "Random sample consensus," *Communications of the ACM*, Jan 1981.

[2] Q. Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder," vol. 3, sep. 2004, pp. 2301 – 2306 vol.3.

[3] J.-Y. Bouguet, "Camera calibration toolbox for matlab," *www.vision.caltech.edu*, 2006.

[4] C. Mei and P. Rives, "Calibration between a central catadioptric camera and a laser range finder for robotic applications," may. 2006, pp. 532 –537.

[5] R. Unnikrishnan and M. Hebert, "Fast extrinsic calibration of a laser rangefinder to a camera," *Tech. Rep. CMU Robotics Institute*, p. 339, 2005.

[6] P. Nunez, P. Drews, R. Rocha, and J. Dias, "Data fusion calibration for a 3d laser range finder and a camera using inertial data," *European Conference on Mobile Robots '09*, p. 9, 2009.

[7] R. Horaud and F. Dornaika, "Hand-eye calibration," *Intl. J. Robot. Res.*, 1995.

[8] D. Scaramuzza, A. Harati, and R. Siegwart, "Extrinsic self calibration of a camera and a 3D laser range finder from natural scenes," in *IROS 2007*, 2007, pp. 4164–4169.

[9] D. Nister, "An efficient solution to the five-point relative pose problem," *IEEE PAMI*, Jan 2004.

[10] M. Bujnak, Z. Kukelova, and T. Pajdla, "A general solution to the p4p problem for camera with unknown focal length," *CVPR 2008*, Jan 2008.

[11] Z. Kukelova and T. Pajdla, "Two minimal problems for cameras with radial distortion," *OMNIVIS 2007*, Jan 2007.

[12] D. Cox, J. Little, and D. O'Shea, "Ideals, varieties, and algorithms," *Springer*, Jan 1997.

[13] M. Minimair, "Mr: Macaulay resultant package for maple," *SIGPLAN Not.*, vol. 39, no. 4, pp. 26–29, 2004.

[14] J. Canny, "A computational approach to edge detection," *IEEE PAMI*, vol. 8, no. 6, pp. 679–698, 1986.