Automatic Online Calibration of Cameras and Lasers

Jesse Levinson, Sebastian Thrun
Stanford Artificial Intelligence Laboratory
{jessel,thrun}@stanford.edu

***Abstract*—The combined use of 3D scanning lasers with 2D cameras has become increasingly popular in mobile robotics, as the sparse depth measurements of the former augment the dense color information of the latter. Sensor fusion requires precise 6-DOF transforms between the sensors, but hand-measuring these values is tedious and inaccurate. In addition, autonomous robots can be rendered inoperable if their sensors' calibrations change over time. Yet previously published camera-laser calibration algorithms are offline only, requiring significant amounts of data and/or specific calibration targets; they are thus unable to correct calibration errors that occur during live operation.**

In this paper, we introduce two new real-time techniques that enable camera-laser calibration online, automatically, and in arbitrary environments. The first is a probabilistic monitoring algorithm that can detect a sudden miscalibration in a fraction of a second. The second is a continuous calibration optimizer that adjusts transform offsets in real time, tracking gradual sensor drift as it occurs.

Although the calibration objective function is not globally convex and cannot be optimized in real time, in practice it is always locally convex around the global optimum, and almost everywhere else. Thus, the local shape of the objective function at the current parameters can be used to determine whether the sensors are calibrated, and allows the parameters to be adjusted gradually so as to maintain the global optimum.

In several online experiments on thousands of frames in real markerless scenes, our method automatically detects miscalibrations within one second of the error exceeding .25 deg or 10cm, with an accuracy of 100%. In addition, rotational sensor drift can be tracked in real-time with a mean error of just .10 deg. Together, these techniques allow significantly greater flexibility and adaptability of robots in unknown and potentially harsh environments.

I. Introduction

As robots are equipped with larger numbers and modalities of sensors, accurate extrinsic sensor calibration becomes increasingly important for the performance of perception systems. Traditional manual calibration techniques tend not to scale well to multi-sensor configurations, and suffer limited accuracy and flexibility. In addition, they are completely infeasible for situations in which a robot needs to dynamically and automatically calibrate its sensors online, due to unforeseen sensor movement.

In this paper, we discuss the automatic calibration of traditional 2D cameras to 3D scanning lasers, and extend our previously published offline methods [20] to offer online solutions. Ever since the birth of mobile robotics, 2D cameras have been among the most popular sensors for perceiving the environment. Over the past decade, lasers have seen increasing use, as their direct and accurate distance measurements provide information not obtainable from traditional cameras. However, lasers do not provide the high spatial resolution and valuable color information offered by 2D cameras. Consequently, much recent research makes use of the complimentary strengths of the two sensors jointly [7, 15, 22, 23].

Fig. 1. Our autonomous vehicle, with laser and camera mounted to the roof. Although our camera is usually mounted directly above our laser (left), our algorithms also successfully calibrate the sensors when they are far apart from each other (right).

On any mobile platform containing cameras and lasers, it is of paramount importance that the sensors be calibrated to each other. Our vehicle platform, with scanning laser and panoramic camera mounted on the roof, is shown in Fig.1. When the camera and laser are properly calibrated, laser measurements can be correctly projected into the camera image, and thus laser points can be associated with three-channel color information. Conversely, pixels in the camera image can be given depth values by querying the nearest laser returns.

Camera-laser calibration has been a difficult problem to tackle; as Le and Ng [5] explain, "It is very challenging to calibrate a camera with a laser range finder. This is because the camera gives 3D rays whereas a laser range finder gives 3D points."

Over the past several years, there has been a number of proposed solutions to the single camera / laser calibration problem. In general, they can be clustered into two bins on each of two axes: first, whether they require a known and fixed calibration target (e.g. a checkerboard) or work in arbitrary scenes, and second, whether they work automatically or require hand-labeling of correspondences between laser points and camera pixels. All of them have been offline algorithms, and are useless for detecting or correcting real-time calibration errors that occur during operation.

Several years ago, Zhang and Pless [1] proposed an algo-

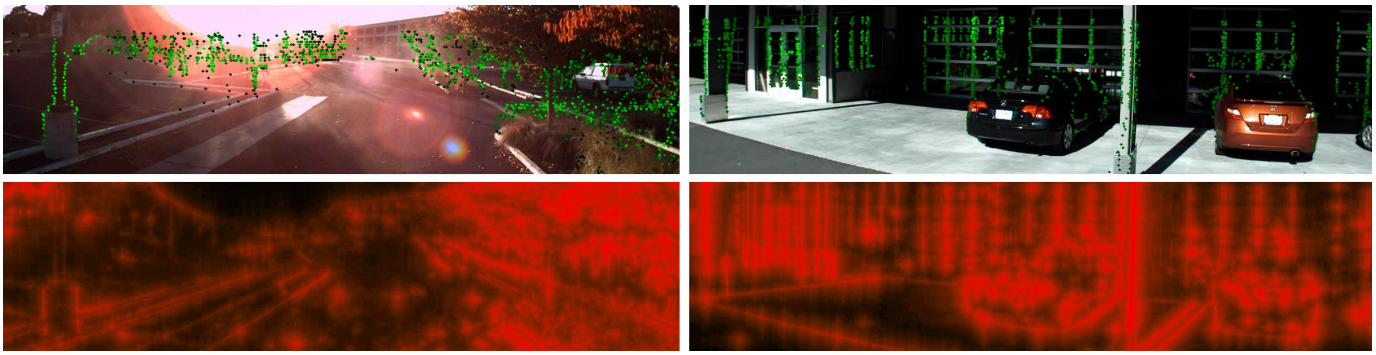

Fig. 2. Two frames, showing RGB camera images with laser points corresponding to depth discontinuities overlayed in green (Top) and processed camera images with inverse distance transform applied (Bottom). Across a sufficient number of frames, an accurate camera-laser calibration will cause the green points to coincide with the red edges more than an inaccurate calibration will.

rithm to calibrate a camera to a laser rangefinder using plane constraints in a scene with a fixed checkerboard calibration target. Shortly thereafter, Unnikrishnan et al. [2] published a toolkit for manual calibration of cameras and lasers, also assuming a dedicated calibration scene. More recently, Scaramuzza et al. [4] demonstrated a technique for improving visualizations of laser data, enabling a manual algorithm for calibrating a rotating 2D laser to a camera in arbitrary scenes. Theirs requires human labeling of correspondences between laser points and camera pixels, and then computes the best calibration using the perspective-from-n-points (PnP) algorithm. On the other side of the spectrum, Kassir and Peynot [12] provide a solution for automatic camera-laser calibration in the presence of a specific calibration target; in this case, corners of a large chessboard target are detected with their "chessboard extraction algorithm", and then automatically aligned with the laser points. Pandey et al. [10] calibrate the same sensors we use in this paper, also automatically, but with an offline method requiring checkerboard target patterns. Geiger et al. [17] calibrate multiple cameras to a multi-beam laser in a single shot, using multiple checkerboard patterns placed across a room. Nunez et al. [6] automatically align laser points with chessboard detections, but aggregate data over multiple frames for higher accuracy, with motion determined by an inertial measurement unit (IMU).

Recently, we described a solution to the most general problem: calibrating a camera to a laser automatically, with no hand-labeling, in an arbitrary scene with no known features [20]. Although more flexible and significantly more accurate than previous methods, it still requires offline processing of sensor data, and takes several minutes to solve the calibration problem. This limitation may not be a problem for robots that have rigidly mounted sensors and operate in calm environments, but it does prevent proper continuous operation of robots whose sensors may occasionally move unexpectedly for any number of reasons.

There exists some published literature on real-time camera-only calibration; Melo et al. [18] calibrate a medical endoscopy camera offline using a checkerboard pattern, and then adaptively updates its calibration parameters in real time using only image data. Petersen and Koch [19] calibrate a camera to an IMU on a mobile robot in real time, but requires a known marker in the scene with a custom pattern.

Due to the greater difficulty of laser-camera calibration, no online solution to the problem has yet been proposed. While solving calibrations with arbitrarily poor initializations remains infeasible in real time, we make two important contributions to online calibration in this paper. First, we show that it is possible to determine *whether* the sensor is calibrated, in real time and with extremely high reliability, even if the correct calibration itself remains elusive. This information can in fact be very valuable to a robot, as it may choose to stop, slow down, or otherwise alter its behavior upon discovering that its fused sensor data are unreliable.

Second, we show that when starting from a correct calibration, the translation and rotation transform parameters can be updated online in response to a slowly changing extrinsic pose of a sensor. Thus, in situations where a sensor is slowly drifting or rotating, the changes can be both detected and corrected in real time, without any interruption to the proper functioning of the robot's perception system.

As long as there exist visible 3D objects with edges in the scene (i.e. not only an expansive ground plane), our objective function can differentiate between calibrated and uncalibrated sensor transforms. Because we make no assumptions about the existence of any known features or landmarks in our scenes, our algorithms are eminently applicable to robots operating in a wide variety of dynamic real-life environments.

II. Sensor Processing

The goal of our algorithm is to take a series of corresponding camera images and laser scans, captured over time in an arbitrary environment, and to automatically determine the true 6-dimensional transform between the two sensors. Specifically, the six values are the x, y, and z translations, and the roll, pitch, and yaw Euler angle rotations between the two sensors. These six parameters uniquely determine the calibration. Whereas in previous work [20] we assumed the calibration to be rigid during the entire time spanned by the data, here we relax

that constraint, and instead approximate the calibration as rigid only over much shorter windows of time.

An underlying assumption, which is used in [16], and which [20] showed to be very robust, is that, everything else being equal, depth discontinuities in laser data should project onto edges in images more often when using accurate calibrations than when not. Across a series of frames, even a weak signal should overwhelm considerable noise.

We assume that the camera images are already calibrated for their intrinsic parameters (e.g. geometric distortion), either with manufacturer-supplied values or through another calibration procedure, e.g. [13]. The laser data is assumed to cover a field of view at least partially overlapping the camera, and to come from multiple rotating beams, or a single beam on a pan/tilt mount.[1]

The camera and laser data may either be captured simultaneously, or if they are not, then any time offsets between the two sensors should be accounted for. Temporal correction is easily achieved by projecting each laser point from the laser's origin at the time of detection. If a laser rotates while the robot is moving, an IMU (or other device) can be used to track the laser's position over time, which allows laser points to be projected into the camera image based on the laser's position at time the image was captured. In such a case, it is helpful to first calibrate the location of laser itself [9, 11, 8, 21]; we use the procedure described in [8], which also works automatically and in arbitrary scenes.

We now describe the algorithms used to process the image and laser data.

A. Image processing

Assume we are given a series of camera images $I^{1:n}$ and point clouds $P^{1:n}$ from n frames. Since the goal is to align laser depth discontinuities with image edges, we filter each camera frame I^i to give a metric of the "edginess" of each pixel. We use a two-step process. First, each image is converted to grayscale, and each pixel is set to the largest absolute value of the difference between it and any of its 8 neighbors.[2] We call this edge image E; an example is shown in Fig.3.

Next, we apply an inverse distance transform to each edge image $E^{1:n}$, in order to reward laser points which hit pixels near edges; effectively, this smooths out the objective function and thus helps to avoid local optima in the search procedure. We use the *L1* distance for ease of computation; in doing so, we can apply the transform in time linear in the number of pixels. Effectively, each edge has an exponential spilloff into its neighbors. Each pixel $D_{i,j}$ is computed as follows:

$$D_{i,j} = \alpha \cdot E_{i,j} + (1-\alpha) \cdot \max_{x,y} E_{x,y} \cdot \gamma^{max(|x-i|,|y-j|)} \quad (1)$$

[1]In principle, the algorithms described here apply equally to the case of a single-beam laser. However, due to the reduced information generated by a single beam, it is likely that more scans would be required for a single-beam laser than a multi-beam or tilting laser for equally robust results. Although not tested here, a dense flash-lidar sensor should be easily amenable to the algorithms described in this paper.

[2]The choice of edge filter is not particularly important.

Fig. 3. Example edge-processed image frame (Top) and after inverse distance transform (Bottom).

where i, j and x, y each correspond to row and column indices for pixels in the image. We use $\alpha = \frac{1}{3}$ and $\gamma = .98$, as in [20]. Examples of images after the inverse distance transform is applied are shown in Fig.2 (bottom), and a before/after comparison is shown in Fig.3.

B. Laser processing

For laser returns, we consider each beam independently, and look for points that are *closer* than at least one of their two neighbors; due to parallax and occlusion, points which are *farther* than their neighbors are less likely to coincide with an image edge. Specifically, we use each point cloud P^i to compute a new point cloud X^i, where each point p in X^i is assigned a value as follows:

$$X^i_p = max(P^i_{p-1}.r - P^i_p.r, P^i_{p+1}.r - P^i_p.r, 0)^\gamma \quad (2)$$

Here, the $.r$ suffix refers to the laser range measurement in meters corresponding to that point. We use $\gamma = .5$, and for efficiency, we filter out all points with a depth discontinuity of less than 30cm. The laser points in Fig.2 (Top) correspond to the points selected in X.

III. Miscalibration Detection

Given a calibration C, we can project all laser points in X^i onto the image D^i using basic geometry; we consider only those points which actually fall in the image. Unlike the offline case in which C would be scored across all n frames in the dataset, here we compute the objective function J_C over just the last w frames, where w is our window size:

$$J_C = \sum_{f=n-w}^{n} \sum_{p=1}^{|X^f|} X^f_p \cdot D^f_{i,j} \quad (3)$$

where f iterates over all frames, p iterates over all 3D points in X^f, and (i, j) refers to the coordinates in image space onto which point p projects. Put simply, this function sums up the depth discontinuities at each laser return in X times the "edginess" of D for some calibration C.

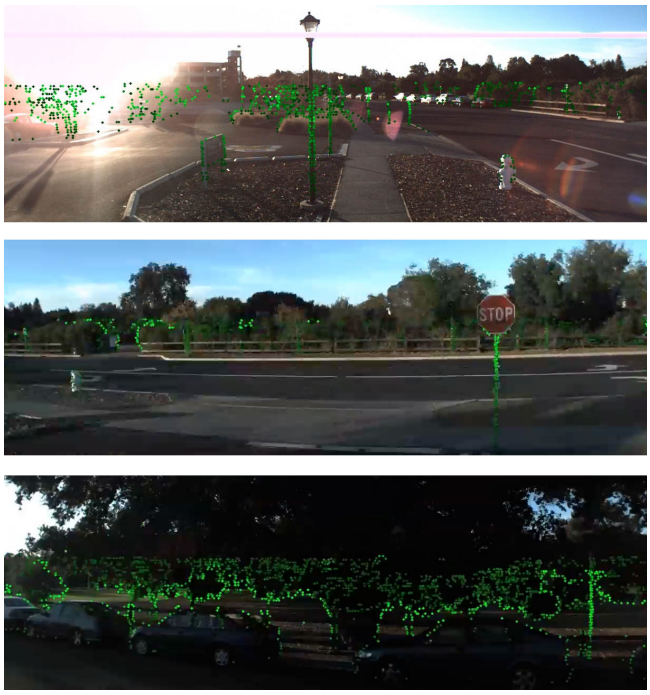

Fig. 4. We used realistic, real-world data to train and test our algorithms. Acquired in the late afternoon, many frames suffer from undesirable artifacts such as lens flare, ghosting, and blooming (Top), sparse 3D features (Middle), and underexposure (Bottom). Although calibration on frames such as these is more challenging than on well illuminated frames with plenty of 3D features, a sufficiently strong signal is still present.

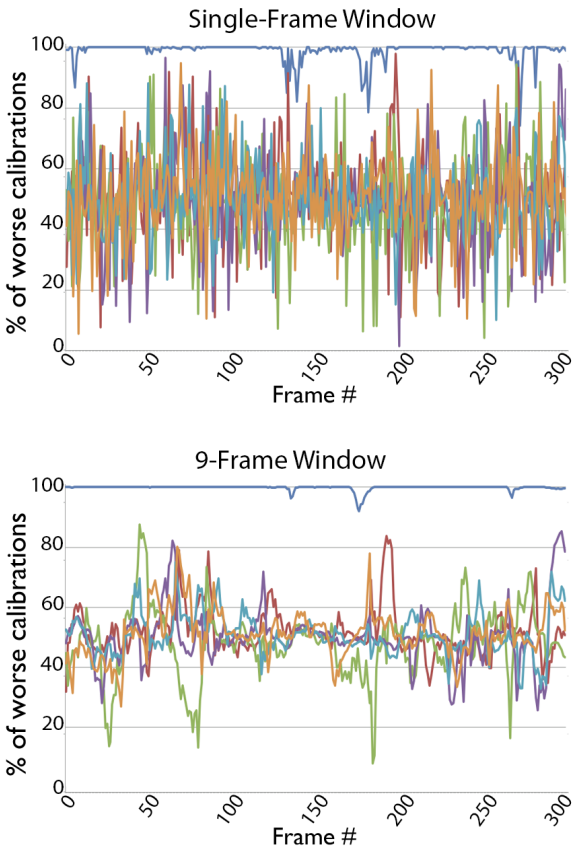

Fig. 5. Probability of worsening the objective function due to perturbation of calibration parameters. In both graphs, the top blue curve is the correct calibration, and the other curves represent incorrect calibrations. The top graph analyzes each frame individually, while the bottom considers a window of 9 frames at a time. Note that the larger window size engenders a significantly larger disparity between the correct and incorrect calibrations.

Ideally, J would be optimized by performing a global search over all possible calibrations C. However, that six-dimensional space cannot be effectively searched in real time, and the objective function is not convex. Therefore, an online search for the optimal value of J is infeasible.

However, despite that the global optimum of J is unobtainable in real time, it is possible to determine, with very high accuracy, whether a given C is correct to within a given threshold. For many classes of problem, it is often far simpler to discern whether a given solution to a problem is correct than it is to determine a correct solution from scratch. Indeed, the same holds here.

Just as a highly out-of-focus camera image is obviously not focused properly, even if the viewer cannot determine the precise distance that the camera should have been focused at, it is possible to determine that a proposed C is wrong even without knowing what the correct C is. In other words, whether or not J_C is a local optimum of J is a strong indication of whether C is correct. But why should that be, given that we know J not to be convex? The answer is that even though J is unlikely to have only one local optimum, the probability of any *given* wrong calibration C being one of those optima is extremely small.

Given a calibration C, we can compute J_C very quickly. If we perform a grid search with radius 1, centered around a given C, across all 6 dimensions, we will compute $3^6 = 729$ different values of J (one of which will be J_C itself; that is, the center of the grid).[3] Let F_C be the fraction of the 728 perturbed values of J that are worse than J_C. For example, if all 728 perturbations of C result in values of J worse than J_C, F_C would be 1. If half of the 728 perturbations of C result in values of J worse than J_C, F_C would be .5.

The key idea is that when C is correct, most perturbations of C should lower the objective function J; after all, a perturbation of a correct calibration must not also be correct, and therefore, if our objective function is effective, it should be worse for inaccurate calibrations. On the other hand, if C is incorrect, there should be a very low chance that J_C will be at a local maximum of J. We now show that this distinction is empirically true.

Fig.5 plots, over a series of frames, what fraction of the 729 perturbations of C result in values of the objective function

[3]The chosen size of the perturbations (in terms of angular rotation and linear translation) corresponds to the granularity by which miscalibrations are desired to be detected. The smaller of a miscalibration that needs to be detected, the smaller a perturbation that should be used; the tradeoff here, as we discuss later, is that smaller perturbations require more frames for reliable results.

J that are worse than J_C, for both the correct C and several incorrect choices of C. In other words, it plots F_C for each of six different Cs across a series of frames. Our hypothesis is confirmed: the F_C values for the correct calibration are significantly higher than those of the incorrect calibrations. If we only use a window size of 1 frame, i.e. $w = 1$, we see that the correct calibration (top blue curve) usually gives the best value of J out of all perturbations, and for all frames, at least 80% of perturbations result in a decrease of J. On the other hand, the other five curves are quite noisy, but on average approximately 50% of perturbations to those calibrations improve J.

Moving to a larger window size of 9 frames (which is just under one second of data at 10Hz), the two cases become significantly more disparate. Here, the correct calibration almost always gives the single best value of the objective function J, and no frame ever goes below F_C = .9. At the same time, the incorrect values of C are now much more concentrated around F_C = .5; both of these changes together make it even easier to disambiguate between the correct and incorrect case.

We can also plot these data in a histogram, making it even more clear how distinct the two distributions are. In Fig.6, we see the distributions for the wrong calibrations on the left, and the correct calibrations on the right. It is readily apparent how different the two distributions are. Furthermore, we see that the top graphs, using a window size of 1 frame, are more similar to each other than the bottom graphs, which use a window size of 9 frames; this is expected, as the signal-to-noise ratio dramatically increases with the benefit of multiple frames.

These observations suggest a natural algorithm for determining whether the sensor is calibrated. Consider the two separate distributions of F_C across a number of training frames, one for correct calibrations and the other for incorrect calibrations. We can fit a Gaussian to each of the two distributions, which then allows us to compute, for any value of F_C, the probability that it was sampled from one distribution versus the other.[4] Building the distributions from tens of incorrect calibrations (each generated by randomly perturbing the 6 calibration parameters) and one correct calibration on several hundred frames from each of several different logfiles, using a 9-frame window, we obtain a mean of $\mu_1 = 99.7\%$ and standard deviation of $\sigma_1 = 1.4$ for correct calibrations, and a mean of $\mu_2 = 50.5\%$ and a standard devation of $\sigma_2 = 14$ for incorrect calibrations.

Therefore, using the standard formula for a Gaussian distribution, we can compute:

$$P(calibrated) = \frac{e^{-.5(x-\mu_1)^2/\sigma_1^2}}{e^{-.5(x-\mu_1)^2/\sigma_1^2} + e^{-.5(x-\mu_2)^2/\sigma_2^2}} \quad (4)$$

where P represents the probability that a calibration C having $F_C = x$ is a correct calibration.[5] It is important to note that P is not a probability distribution over calibrations; rather, it is a statistical test which gives the probability of the observed data arising from a correctly calibrated sensor versus an incorrectly calibrated sensor.

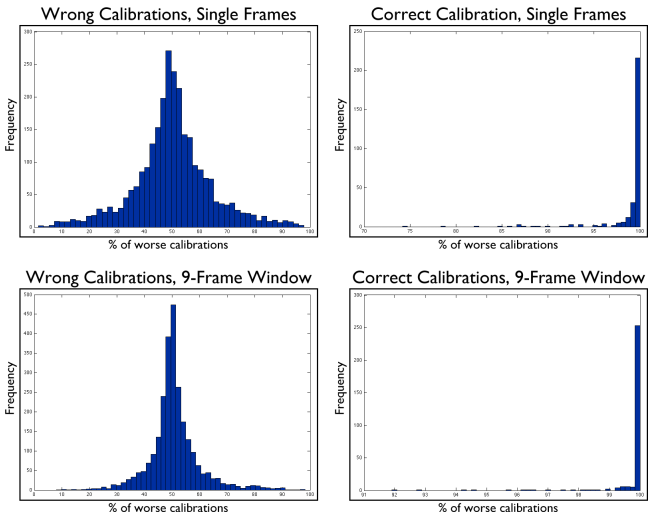

Fig. 6. Histogram plots illustrating the widely disparate probability distributions of the objective function behavior when considering correctly vs. incorrectly calibrated sensors. Note that the x-axis for the top-right graph goes from 70 to 100, and for the bottom-right graph goes from 90 to 100; thus, the correct and incorrect distributions are even more different than they initially appear.

To be clear, although mathematically there is only exactly one calibration C which is actually "correct", so that even an arbitrarily small deviation becomes "incorrect", that is not the criteria used in this classification. Instead, the formula simply answers whether the sensors being calibrated or uncalibrated best explain the observed data, assuming each is equally likely *a priori*. Importantly, the tightness of the definition of correctness can be adjusted by changing σ_1, while the tightness of the definition of incorrectness is controlled by σ_2.

Thus we have derived an easily computable formula that yields the probability that the robot's current sensor calibration is accurate. Depending on the context, if this value falls below a designated threshold, a robot may choose to alert a command center, suspend its operation until further notice, or simply pause its activities to perform a more comprehensive offline calibration before resuming its work.

IV. Automatic Calibration Tracking

In addition to determining whether a calibration is likely to be correct or not, we can also exploit the local convexity of the objective function J near the global optimum J_C to track small changes in C over time.

If we consider all 729 perturbations of C from the grid search described in the previous section, some of them should

[4]Empirically, the distributions appear closer to Laplacian than Gaussian, but in this case, the distinction is relatively unimportant. Since the histograms corresponding to the correct and incorrect cases are so different from each other, a precise model would be of little practical benefit.

[5]This formula is accurate as long as we assume *a priori* that a correct calibration is as likely as an incorrect calibration. However, a nonuniform prior probability of C being correct can be trivially incorporated into the equation, if desired.

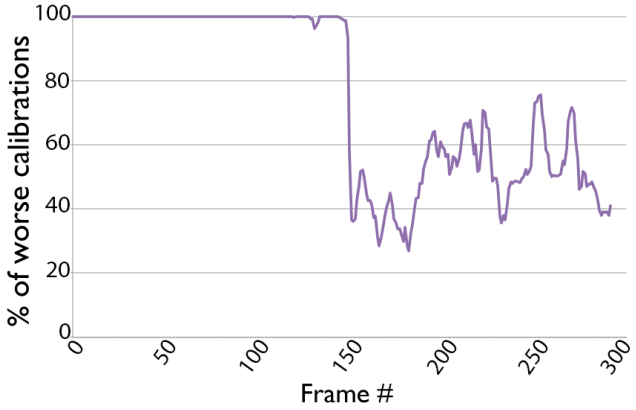

Fig. 7. The sensor becomes synthetically miscalibrated at the halfway point (Frame 150). Here we see that during the first half, when the sensor is correctly calibrated, almost all of the perturbations to C cause the objective J to decrease. However, when the sensor is shifted by a fraction of a degree in roll, pitch, and yaw, and several cm in x, y, and z, that ceases to be the case.

be improvements if C is slightly incorrect. Of course, if C is wildly incorrect, there is no guarantee that ascending J will lead us in the right direction. However, if C is close to the right answer, gradient ascent would be expected to work. Our implementation is straightforward: at each iteration, the calibration C remains constant if all perturbations of C cause J to decrease, or else we select a new calibration C' which is the calibration out of all 729 candidates that gives the best value of J.

Indeed, as we show in the results section, a slow and gradual adjustment of C in the direction of higher J does in fact allow us to track calibration changes over time. Unsurprisingly, the more frames that are used in the window, the better the signal-to-noise ratio is. It is not realistic to expect a single-frame analysis to work all of the time, particularly in poorly-lit areas, areas with fewer 3D objects, or under severe flare or other artifacts. Instead, we find that a 9-frame analysis window is significantly more robust.

As a consequence of a multi-frame window, we assume that the calibration changes within the window are negligible. For a window size of under a second, this should be a valid assumption in almost all scenarios.

V. Experimental Results

For our experiments, we used a Velodyne HDL-64E S2 LIDAR sensor with 64 beams oriented between -22 and +2 degrees vertically and rotating 360 degrees horizontally. The unit spins at 10Hz and provides around 100,000 points per spin. The camera was a Point Gray Ladybug3 Panoramic unit with 5 separate cameras, of which we used just one for this paper. We ran the camera at 10Hz and used the middle third of the 1200x1600 vertical image, since the vertical field of view of the camera far exceeds that of our laser. The sensors were mounted to a vehicle equipped with an Applanix LV-420 GPS/IMU system, which was only used to adjust for local motion of the laser during each frame; no global coordinate frame or GPS coordinates were used, as the algorithms presented here do not require a globally consistent trajectory of the vehicle.

Our algorithms were implemented in C++, and all of the results we describe below were obtained in real time on a laptop CPU.

A. Online Miscalibration Detection

We performed an extensive series of evaluations to determine how quickly and accurately our algorithms could detect sensor miscalibrations. Using the distributions from Fig.6 and the probabilities given by Eq. (4), we evaluated thousands of frames with correct and incorrect calibrations.

A typical example is shown in Fig.7. Here, for 150 frames, the correct calibration was used, and on every frame, the probability of correct calibration was 100.0%; again, the two distributions are so different, that in almost all cases, the algorithm will be essentially sure of one answer or the other. Within a few frames of the miscalibration (using a 9-frame window), the percentage of bad perturbations drops precipitously, and remains in a range well within the incorrect distribution; again, in all of these cases, the probability of incorrect calibration is 100.0%.

We note that in generating these distributions, we used random translational offsets of up to 20cm and random rotational offsets of up to 2 degrees. However, we also evaluated how accurately we could detect very small calibration errors. Of course, arbitrarily small errors will be undetectable, especially with only a 9-frame window.

In our experiments with a 9-frame window, over thousands of frames, we never incorrectly classified a calibration as being correct or not if the margin of correctness was at least .25 degrees or 10 cm. Testing rotational calibration errors as small as 0.1 degrees, we were able to discern the correct answer over 90% of the time. There is a tradeoff between window size and error detection, so if it is important to be able to reliably detect errors on the order of .1 degrees, a larger window could be used, with the downside that it would likely take over a second to detect the misalignment.

B. Online Miscalibration Correction

To evaluate the online calibration tracking performance of our algorithm, we applied additive random synthetic offsets to all three rotation parameters of our camera calibration transform at each frame in a 1500-frame logfile. Thus, the roll, pitch, and yaw of the sensor was artificially modified by Brownian motion in each parameter, and we attempted to track these (unknown to the algorithm) synthetic rotational offsets in real time. Each offset was either increased or decreased by .02 degrees on each frame, at random, which represents quite a high rate of drift for a sensor on a robot.

We believe that rotational drift is the far more common case for sensor movement on mobile robots, as it is common for a sensor to be relatively immobile but still slightly free to rotate about one or more axes.

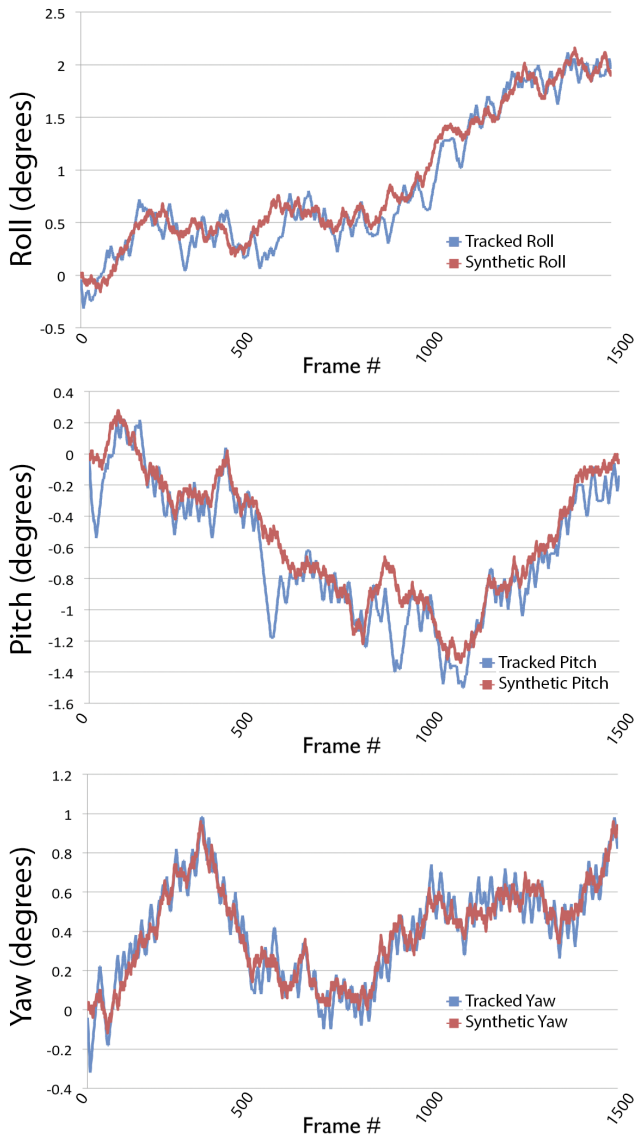

Fig. 8. Simultaneously tracking synthetic Brownian motion of roll, pitch, and yaw in real time. Our system does a good job at tracking random roll and pitch rotations, with an average mean error of .12 degrees, and an excellent job at tracking random yaw rotations, with an average mean error of only .06 degrees.

In our experiments, we were able to simultaneously track yaw with a mean average error of .06 degrees, and roll and pitch with a mean average of .12 degrees each. It is not surprising that yaw is the most precise, as the nature of the scanning laser (sweeping from left to right with high horizontal angular resolution) makes it considerably more sensitive in that dimension. In contrast, the laser's vertical resolution is several times lower than its horizontal resolution, and in most scenes, there are typically fewer 3D features in that direction as well, which explains the greater difficulty of tracking pitch. Finally, changes in roll induce smaller deviations in pixel space than yaw and pitch rotations, since points in the center of the frame are unaffected by roll. Overall, an average rotational

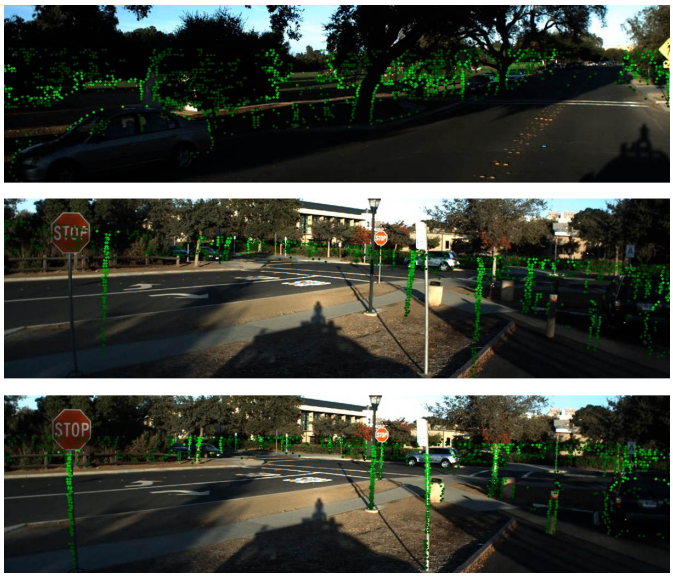

Fig. 9. Tracking calibration errors in real time. (Top) shows the first frame in a 400-frame sequence. Initially, the transform is correct. We then synthetically perturb two translation and two rotation parameters, over the course of the 400 frames, resulting in a badly miscalibrated sensor (Middle). However, with our online tracking algorithm, the transform updates itself in real time, adjusting to the perturbations, and ends up almost perfectly calculating the synthetic shift all the way through the end of the sequence (Bottom).

error of .10 degrees across all dimensions is quite low, and actually exceeds the precision of most offline calibration techniques [12, 4]. Fig.8 shows how accurately we are able to track random sensor drift in all three rotational dimensions at once.

A qualitative example of tracking is shown in Fig.9. Here, the algorithm starts out with the correct calibration, but then the calibration is synthetically worsened over time, in two translational and two rotational dimensions. After 400 frames, the synthetic calibration is wildly inaccurate, but the tracked result, which was done in real time, is very well aligned.

VI. Conclusion

As robots move away from laboratory and factory settings and into real-life, unpredictable, and long-term operations, it is essential that sensor miscalibrations do not render them inoperable. A robot should be able to detect and correct errors in its calibration in real-time during operation, so that it can continue operating safely and effectively.

In this paper, we have developed two new algorithms to assist robots equipped with cameras and lasers in the reliability of their perception systems over time. The constant background monitoring algorithm that detects sudden miscalibrations is an important tool for robots which need to know whether they can actually rely on the sensor data they're receiving. While this is but one of many important checks a robot ought to perform on its sensor data, we believe it is an important, and perhaps underappreciated one.

Additionally, we have shown that it is possible to track gradual drift of sensor pose over time, without performing

computationally intensive global optimizations over the entire search space. This technique is suitable to be run in a background process, consuming very little CPU time, but potentially significantly improving the accuracy of a perception pipeline that includes camera-laser fusion. As expected, there is a tradeoff between the sensitivity with which minor miscalibrations can be detected and the number of frames required to make the determination. Yet even using less than one second of data, our results are more accurate than state-of-the-art *offline* techniques which require a calibration target [12] or hand-labeling of camera-laser correspondences [4].

Further improvements in tracking accuracy and robustness should be possible by considering larger grid radii or by using a Monte Carlo sampling approach, such as a particle filter, rather than the greedy approach described here.

Although we focused on calibrating cameras to lasers in this paper, we hope that some of these insights will be useful to a variety of calibration tasks. In particular, the formation of a simple objective function that can discern the difference between a correctly and incorrectly calibrated sensor is often relatively straightforward, and from that starting point, many of the techniques discussed here should be readily applicable.

VII. Acknowledgments

We are grateful to Mike Sokolsky and Brice Rebsamen for maintaining our autonomous vehicle research platform, and to Alex Teichman for his implementation of laser-to-camera projection code.

References

[1] Q. Zhang and R. Pless. Extrinsic calibration of camera and laser range finder. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004.

[2] R. Unnikrishnan et al. Fast extrinsic calibration of a laser rangefinder to a camera. *CMU, Tech. Rep.*, 2005.

[3] S. Thrun, W. Burgard and D. Fox. Probabilistic Robotics. *MIT Press*, 2005.

[4] D. Scaramuzza, A. Harit, and R. Siegwart. Extrinsic Self Calibration of a Camera and a 3D Laser Range Finder from Natural Scenes. In *International Conference on Intelligent Robots and Systems (ICRA)*, 2007.

[5] Q. Le and A. Ng. Joint calibration of multiple sensors. In *International Conference on Intelligent Robots and and Systems (IROS)*, 2009.

[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. In *Camera*, 2009. pp. 31-36.

[7] B. Douillard, A. Brooks, and F. Ramos. A 3D Laser and Vision Based Classifier. In *International Conference in Intelligent Sensors, Sensor Networks and Information Processing*, 2009.

[8] J. Levinson and S. Thrun. Unsupervised Calibration for Multi-beam Lasers. In *International Symposium on Experimental Robotics*, 2010.

[9] G. Chao and J. Spletzer. On-Line Calibration of Multiple LIDARs on a Mobile Vehicle Platform. In *International Conference on Robotics and Automation (ICRA)*, 2010.

[10] G. Pandey, JR. McBride, S. Savarese, and RM. Eustice. Extrinsic Calibration of a 3D Laser Scanner and an Omnidirectional Camera. In *7th IFAC Symposium on Intelligent Autonomous Vehicles*, 2010.

[11] M. Sheehan, A. Harrison, and P. Newman. Automatic Self-Calibration of a Full Field-of-View 3D n-Laser Scanner. In *International Symposium on Experimental Robotics (ISER)*, 2010.

[12] A. Kassir and T Peynot. Reliable Automatic Camera-Laser Calibration. In *Australasian Conference on Robotics and Automation*, 2010.

[13] J.Y. Bouguet. Camera Calibration Toolbox for Perspective Cameras. Online at http://www.vision.caltech.edu/bouguetj/calib_doc/

[14] H. Men, B. Gebre, and K. Pochiraju. Color Point Cloud Registration with 4D ICP Algorithm. In *International Conference on Robotics and Automation (ICRA)*, 2011.

[15] M. Haselich, M. Arends, D. Lang and D. Paulus. Terrain Classification with Markov Random Fields on Fused Camera and 3D Laser Range Data. In *European Conference on Mobile Robots (ECMR)*, 2011.

[16] L. Baboud, M. Cadik, E. Eisemann and H. Seidel. Automatic Photo-to-Terrain Alignment for the Annotation of Mountain Pictures. In *Computer Vision and Pattern Recognition (CVPR)*, 2011.

[17] A. Geiger, F. Moosmann, C. Omer, and B. Schuster. Automatic Camera and Range Sensor Calibration Using a Single Shot. In *International Conference on Robotics and Automation (ICRA)*, 2012.

[18] R. Melo, JP. Barreto, and G. Falcao. A new solution for camera calibration and real-time image distortion correction in medical endoscopy-initial technical evaluation. In *IEEE Trans Biomed Eng, 2012 March. 59(3):634-44*.

[19] A. Petersen and R. Koch. Video-based realtime IMU-camera calibration for robot navigation. In *Proc. SPIE 8437, Real-Time Image and Video Processing 2012, 843706*, (June 1, 2012).

[20] J. Levinson and S. Thrun. Automatic Calibration of Cameras and Lasers in Arbitrary Scenes. In *International Symposium on Experimental Robitics (ISER)*, 2012.

[21] M. Sheehan, A. Harrison, and P. Newman. Self-calibration for a 3d laser. In *International Journal of Robotics Research (IJRR)*, 2012.

[22] Y. Bok, D. Choi and I. Kweon. Generalized Laser 3-Point Algorithm for Motion Estimation of Camera-Laser Fusion System. In *International Conference on Robotics and Automation (ICRA)*, 2013.

[23] D. Held, J. Levinson and S. Thrun. Precision Tracking with Sparse 3D and Dense Color 2D Data. In *International Conference on Robotics and Automation (ICRA)*, 2013.