Unsupervised extrinsic calibration of depth sensors in dynamic scenes

Stephen Miller, Alex Teichman, and Sebastian Thrun

Abstract— **While inexpensive depth sensors are becoming increasingly ubiquitous, field of view and self-occlusion constraints limit the information a single sensor can provide. For many applications one may instead require a network of depth sensors, registered to a common world frame and synchronized in time. Historically such a setup has required a tedious manual calibration procedure, making it infeasible to deploy these networks in the wild, where spatial and temporal drift are common. In this work, we propose an entirely unsupervised procedure for calibrating the relative pose and time offsets of a pair of depth sensors. So doing, we make no use of an explicit calibration target, or any intentional activity on the part of a user. Rather, we use the unstructured motion of objects in the scene to find potential correspondences between the sensor pair. This yields a rough transform which is then refined with an occlusion-aware energy minimization. We compare our results against the standard checkerboard technique, and provide qualitative examples for scenes in which such a technique would be impossible.**

I. INTRODUCTION

With the advent of inexpensive depth sensors such as the Microsoft Kinect or Asus Xtion Pro, virtually anyone can now collect 3D pointclouds for the cost of a point-and-shoot camera. Rather than discerning information from a rectangular grid of pixels, algorithms can instead reason in the intuitive world of Euclidean space. This has simplified many perceptual tasks—be it mapping, reconstruction, object detection, or scene understanding.

However, these sensors are confined to a rather limited field of view. As such, algorithms must reason about non-intuitive, self-occluding hulls rather than full, 3D shapes. One can imagine instead a room with a network of mounted depth sensors, each with a novel view of the scene. With proper extrinsic calibration—knowledge of the translation, rotation, and time offset of each sensor with respect to a world frame—full pointclouds of moving objects could be constructed in realtime, allowing algorithms to analyze dynamic 3D scenes.

Such extrinsics are often difficult to obtain, since they typically require ground-truth correspondences by means of a precise calibration pattern. If any camera is bumped or is prone to drift, the entire routine must be redone. Even in the research setting, this is frustrating and time-consuming. When deployed in large-scale environments where there is no dedicated engineer to monitor the setup, it becomes a very serious stumbling block.

The task of automating this procedure has been well-studied in the 2D domain. Sensor registration is typically done by matching 2D keypoints—SIFT [1] and ORB [2] being two common examples. Typical indoor scenes, however, are often fairly untextured. These environments, when seen from wildly different viewing angles and exposures, do not lend themselves to feature-based approaches. This is particularly true for commodity depth sensors, whose RGB cameras—-if they exist at all—-are often low resolution and imperfectly registered to the depth image.

In this paper, we aim to make the extrinsic calibration task as painless and off-the-shelf as the sensors themselves, requiring no intentional human effort or distinctive texture cues. Rather than relying on structured calibration patterns, we will use the dynamic scene itself to calibrate. As objects move within the scene, their positions at each time frame provide candidate correspondences from which crude extrinsics can be inferred. This initial hypothesis is then refined by densely aligning the dynamic geometry of the scene, optimizing both the spatial and temporal offsets of the two sensors. We make no assumptions about the shape of the objects or the way they move, nor do we assume anything about the relative poses of the cameras beyond the basic requirement that *something* in the scene must be visible to both simultaneously.

In Section II we consider related work in the 2D and 3D domains. We then briefly introduce the problem in Section III, and present our unsupervised solution in Section IV. In Section V we compare our results against the standard checkerboard calibration approach, and give a number of qualitative examples.

II. RELATED WORK

Camera calibration

The problem of registering two sensors together has been extremely well studied in Computer Vision. By far the most common technique is the approach of Zhang [3], as widely popularized by Bouget's MATLAB toolbox [4] and the calibration tools present in the OpenCV library [5]. This approach uses a planar calibration target to establish camera-to-world correspondences, which can then be used to estimate intrinsic and extrinsic parameters. Although a checkerboard is the most common source of correspondences, others have used circular [6] and point [7] targets for similar purposes. Target-free techniques also exist for autocalibrating multiple cameras, particularly in the stereo domain. Demirdjian et al. [8] solve for a stereo baseline by means of a moving, textured plane. Houraud et al. [9] use sensor egomotion to solve for the stereo baseline in unstructured environments. Similar to our intuitions, Stauffer and Tieu [10] solve for the topology of a large sensor network by tracking the motion

Stephen Miller, Alex Teichman, and Sebastian Thrun are with the Department of Computer Science at Stanford University.
E-mail: {stephen, teichman, thrun}@cs.stanford.edu

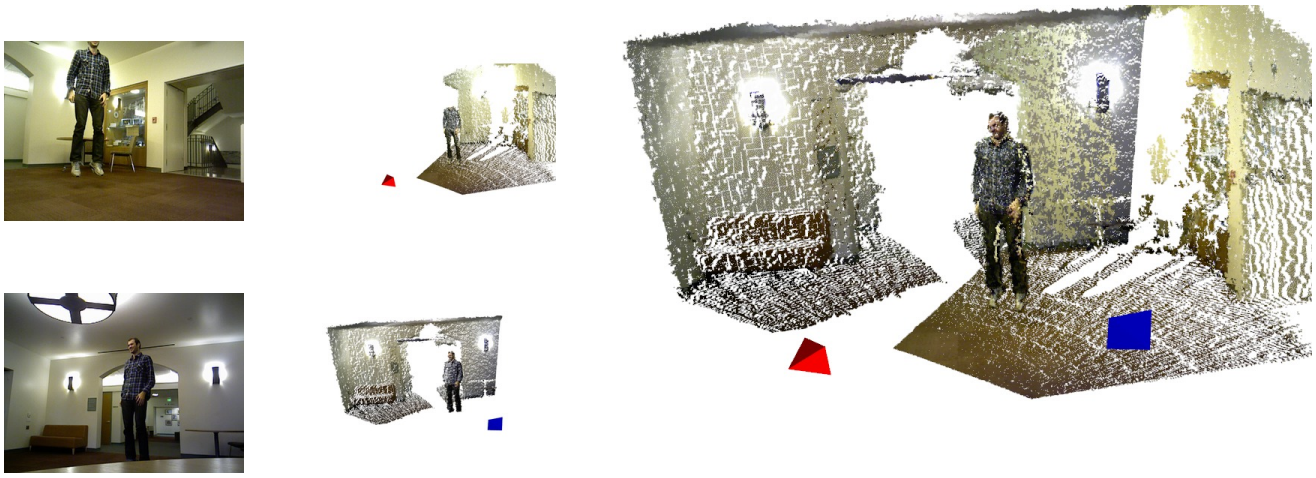

Fig. 1. An example calibration task. **Left**: the image and associated pointcloud for both sensors. **Right**: The combined cloud returned by our calibration routine.

of multiple arbitrary objects which are assumed to be planar. This assumption works well for their surveillance use case, for which distant cars are well approximated by planes.

Structure from motion, bundle adjustment

The goal of sensor alignment is not unique to extrinsic calibration. There is a large body of work in the Computer Vision community which aims to reconstruct the 3D structure of static scenes via multiple discrete camera views. In this task the sensor pose is often estimated as an inner loop of an alternating optimization algorithm, and thus these methods could be adapted for extrinsic calibration purposes. Seminal work by Pollefeys et al. [11], Agarwal et al. [12], Bao et al. [13], and numerous others have shown that sensor pose and scene geometry can be inferred from multiple, often uncalibrated sensors. An excellent survey of these methods is given by Triggs et al. [14], for those interested. Recent industrial advances such as Autodesk's 123D[1] have also shown impressive performance on a small scale, reconstructing shape from unordered cellphone images. And while state of the art reconstruction techniques such as DTAM [15] require a continuous trajectory rather than disparate viewpoints, the task of finding loop closures is quite similar to our own.

The above methods are clearly infeasible for calibrating sensors with only a depth channel. For those which provide registered RGB information, however, they could easily be adapted to the task at hand. Unfortunately, we've found that camera quality and depth registration error tend to limit their effectiveness. Even with high quality sensors, these approaches naturally favor highly textured scenes with maximally redundant fields of view. This is not unreasonable when calibrating a low-baseline stereo rig, reconstructing objects of interest, or inferring sensor egomotion in cluttered environments. When the scene itself is relatively featureless and sensors are placed to maximize coverage, however, these assumptions no longer hold.

Depth sensor calibration

Calibration techniques have also been designed explicitly for RGB-D sensors. Zhang and Zhang [16] use a checkerboard to track correspondences between the RGB and IR sensors. Others have developed a glass checkerboard[2], which has the advantage of being directly visible in the depth image. While these approaches attempt to register a depth sensor with an RGB camera, the task of finding correspondences is the same in the multi-sensor scenario.

In recent years, a body of work has emerged which attempts to perform extrinsic calibration in an unsupervised fashion, by optimizing the geometry of the reconstructed scene. Levinson and Thrun [17] propose a method for calibrating 2D and 3D LIDAR sensors mounted on a moving platform, using the observation that, on a small scale, local regions tend to be planar. Maddern et al. [18] and Pandey et al. [19] propose similar techniques, replacing the planarity assumption with entropy and mutual-information terms, respectively.

While the task of aligning a sensor to a vehicle reference frame is distinct from our own, the spirit of these approaches is quite similar. Both approaches, however, require an initial estimate of the sensor pose. This is reasonable for a specialized vehicle but less convenient when mounting sensors in new environments. We also note that depth discretization error is far more extreme in the Primesense-style sensors we wish to calibrate than it is in Velodyne data, and it is doubtful that a point-to-plane or entropy term alone would suffice.

Perhaps most similar to ours in spirit, if not domain, is the work of Kodagoda et al. [20], which tracks the motion of objects in a scene to align the 2D poses of coplanar LIDARs. This uses a fairly sophisticated motion model to directly align

[1]http://123dapp.com

[2]No known citation; see, for example: http://doc-ok.org/?p=289

Fig. 2. (A) Foreground pixels (shown in red) are extracted from the background. (B) Connected components in the foreground image are used to find candidate objects for each frame. (C) The centroids of these objects, aggregated over all time, provide candidate correspondences which RANSAC can use to estimate an initial pose. (D) This initialization is refined with an alternating grid search on foreground objects across all frames simultaneously.

the seen trajectories; while we too exploit motion, we are largely agnostic to the *way* in which things move. It is unclear whether the trajectory-based approach would be robust or scalable in the 6DOF domain.

Pointcloud registration

Finally, the problem of data association in pointclouds has been explored for a wide range of applications. As is the case in 2D, this is often done for the purposes of reconstruction from a continuously moving sensor. Recent work by Endres et al. [21], Henry et al. [22], and Whelan et al. [23] leverage both RGB and depth channels found in commodity sensors, while Newcombe et al. [24] and Bylow et al. [25] use the depth map alone. These systems, which depend on continuous egomotion, are not applicable to the task at hand. However, the loop-closure and registration techniques they rely on solve a similar problem to ours. While 2D feature matching is the most common, depth features such as the Fast Point Feature Histogram [26] have also been proposed. Other methods, like that of Makadia et al. [27], register clouds by globally aligning histograms of surface normals. Both approaches, by relying on accurate surface normals, do not scale well to large, noisy environments. Once crude initializations are given, local techniques such as point-to-plane [28] or generalized [29] ICP are often used to refine these registrations. While these could easily be adapted to our approach, we found that a dense grid search is better at overcoming poor initializations.

III. NOTATION

We consider a pair of sensors S_0 and S_1, each of which records timestamped depth maps and, optionally, color images. Given proper camera intrinsics, each depth reading can be converted into a 3D point measurement. This yields a stream of 3D pointclouds $\{\mathbf{C}_0^{(t)}\}, \{\mathbf{C}_1^{(t)}\}$, where $p \equiv \mathbf{C}(u,v)$ is the 3D point measurement at pixel (u,v). Here our coordinate system follows standard pinhole camera conventions, where the image plane lies in XY and $+Z$ denotes forward distance from the sensor.

While our timestamps are continuous, depth readings are only given at discrete points in time. For notational simplicity, then, we let $\mathbf{C}^{(t)}$ denote the cloud whose timestamp is *nearest* to t.

We wish to bring these clouds into a common reference frame. Without loss of generality, we assume S_0 lies at the origin of the world frame with accurate timestamps. The goal of this work, then, is to find the 6DOF transform $\mathscr{T}$ and time offset Δt such that $\mathscr{T}\mathbf{C}_1^{(t+\Delta t)}$ is aligned with $\mathbf{C}_0^{(t)}$.

IV. METHOD

As noted in Section I, this work hinges on the premise that the unstructured scene alone contains enough information to align two sensors. We will frame this as an optimization task, formulating an energy term $\mathbf{E}[\mathscr{T}, \Delta t]$ such that its minimum will be attained when the two sensors are properly aligned.

However, the search space is large and the objective non-convex, rendering a good initialization essential. To get a rough estimate of pose, we observe that even under drastically different viewing conditions one cue is fairly stable: the motion of foreground objects. These objects—or more accurately, their centroids—provide a sparse set of candidate correspondences, much like the corners of a checkerboard or results of a keypoint detector. Aggregated across all timesteps, these correspondences are used to predict an initial transform $\mathscr{T}_0$ via RANSAC. This prediction is then refined by an alternating minimization of $\mathbf{E}[\mathscr{T}, \Delta t]$. This approach is demonstrated visually in Figure 2, and outlined in Algorithm 1.

Algorithm 1: Calibration Pipeline

Data: Two sequences, $\{\mathbf{C}_0^{(t)}\}$, $\{\mathbf{C}_1^{(t)}\}$
Result: Transform $\mathscr{T}$ and time offset Δt
$\{O_{k,i}^{(t)}\} \leftarrow$ ExtractObjects($\{\mathbf{C}_k^{(t)}\}$);
$\mathscr{T}_0 \leftarrow$ CentroidRANSAC($\{O_{0,i}^{(t)}\}, \{O_{1,i}^{(t)}\}$);
$\mathscr{T} \leftarrow \mathscr{T}_0$;
$\Delta t \leftarrow 0$;
while $||\Delta\mathscr{T} - I||_F > \varepsilon$ **do**
  $\mathscr{T} \leftarrow \underset{\tilde{\mathscr{T}}}{\operatorname{argmin}} \mathbf{E}[\tilde{\mathscr{T}}, \Delta t]$;
  $\Delta t \leftarrow \underset{\tilde{\Delta t}}{\operatorname{argmin}} \mathbf{E}[\mathscr{T}, \tilde{\Delta t}]$;
end

A. Object extraction

We first segment each frame into foreground and background pixels. To do so, we sweep through the entire sequence and build up a per-pixel depth histogram

$$H_k(u,v,\hat{z}) \propto \sum_t \mathbf{1}\{b[C_k^{(t)}(u,v)_z] = \hat{z}\}$$

Where $b[\mathbf{z}]$ maps depth values to histogram bins. A foreground mask is then computed for each frame by finding pixels whose current depth values do not frequently occur in the sequence. Candidate objects $O_{k,i}^{(t)}$ are then extracted by finding large connected components in the foreground mask.

B. Initial alignment via Centroid RANSAC

We now wish to use these foreground objects to infer an initial transform. To do so, we follow the standard RANSAC approach: sample 3 pointwise correspondences, use them to estimate a 6DOF transform, and repeat for a set number of iterations. The transform which yielded the most inliers is chosen, and subsequently refined.

In the sampling step, a foreground object $O_{0,i}^{(t)}$ is chosen at random, among all clouds given by S_0. A corresponding object $O_{1,j}^{(t)}$ is sampled at random from $\mathbf{C}_1^{(t)}$. We repeat this three times, then use Levenberg Marquardt to solve for the 6DOF transform $\mathscr{T}_{guess}$ which minimizes the sum of squared distance to the corresponding objects' centroids.

Note that this sampling step is done jointly across all frames; thus, the need for three point correspondences does not mean that three objects must move. In fact, a single moving object, aggregated across multiple frames, provides ample correspondences.

This procedure is detailed in Algorithm 2.

Algorithm 2: Centroid RANSAC

Data: Two roughly synchronized sets of foreground objects, $\{O_{0,i}^{(t)}\}, \{O_{1,i}^{(t)}\}$
Result: Rough guess of transform, $\mathscr{T}_0$

```
BestInliers ← {};
for K iterations do
    Corr ← {};
    for 3 correspondences do
        t' ← RandomSelect({t});
        O_{0,i}^{(t')} ← RandomSelect({O_{0,i}^{(t')}});
        X_0 ← CentroidOf(O_{0,i}^{(t')});
        O_{1,j}^{(t')} ← RandomSelect({O_{1,j}^{(t')}});
        X_1 ← CentroidOf(O_{1,j}^{(t')});
        Corr = Corr ∪{(X_0 ↔ X_1)};
    end
    T_guess ← EstimateTransform(Corr);
    Inliers ← ComputeInliers(T_guess, d_inlier);
    if |Inliers| > |BestInliers| then
        T ← T_guess;
        BestInliers ← Inliers;
    end
end
T ← EstimateTransform(BestInliers);
```

C. Pose and synchronization Refinement

With a rough transform $\mathscr{T}_0$ given by Centroid RANSAC and an initial guess of $\Delta t = 0$, we can now perform our optimization. To do so, we formulate an energy function $\mathbf{E}[\mathscr{T}, \Delta t]$ which encodes the intuition that points which are visible in both scenes should align, while being robust to the largely non-overlapping portions of the scene.

It is tempting to use a simple Nearest Neighbor penalty across all frames:

$$\mathbf{E}[\mathscr{T}, \Delta t] = \sum_t \sum_{p_1 \in \mathbf{C}_1^{(t)}} \phi^{(t-\Delta t)}[\mathscr{T} p_1] \quad (1)$$

$$\phi^{(t)}[\hat{p}_1] = h(\|\hat{p}_1 - NN(\hat{p}_1, C_0^{(t)})\|) \quad (2)$$

$$h(\mathbf{x}) = \min(\mathbf{x}, d_{\max}^+) \quad (3)$$

Where $NN(p, \mathbf{C})$ denotes the nearest neighbor of point p in cloud $\mathbf{C}$, and the hinge loss $h(\mathbf{x})$ minimizes the effect of distant outliers.

However, as can be seen in Figure 3, this penalty can have adverse effects when the area of overlap between the sensors is small. When an object is viewed from the left and right sides, the penalty serves to compress the object, favoring pointwise overlap at the cost of plausibility. To correct for this, we add a free space violation term:

$$\phi^{(t)}[\hat{p}_1] = h(\|\hat{p}_1 - NN(\hat{p}_1, \mathbf{C}_0^{(t)})\|_{xyz^-}) + FSV \quad (4)$$

$$\|\mathbf{x}\|_{xyz^-} = \begin{cases} \|\mathbf{x}\|_1, & \mathbf{x}\cdot\mathbf{u} > -d_{\max}^- \\ \|\mathbf{x} - (\mathbf{x}\cdot\mathbf{u} + d_{\max}^-)\mathbf{u}\|_1 & \text{otherwise} \end{cases} \quad (5)$$

$$FSV = \begin{cases} |p_{0,z}^{\text{proj}} - \hat{p}_{1,z}|, & p_{0,z}^{\text{proj}} - \hat{p}_{1,z} > -d_{\max}^- \\ d_{\max}^-, & \text{otherwise} \end{cases} \quad (6)$$

Where $p^{\text{proj}} = C_0^{(t)}(u_{\hat{p}}, v_{\hat{p}})$ is the point in $\mathbf{C}_0^{(t)}$ which projects to the same pixel as $\hat{p}_1$, and $\|\mathbf{x}\|_{xyz^-}$ an anisotropic L1 distance which does not penalize for large positive displacement along the ray from S_1 to p_1, denoted $\mathbf{u}$. Here, the penetration limit $d_{\max}^-$ signifies the minimum allowable object thickness, below which any disparity should be attributed to noise. These combine to encode the intuition that if a point in one sensor is mapped to space which was observed to be free in the other, modulo sensor noise, it should incur a penalty. Otherwise it may be plausibly explained by occlusion, and should incur no additional cost.

We optimize this objective with an alternating minimization scheme. Holding Δt fixed, we solve for $\mathscr{T}$ by with a dense 6DOF grid search. We then hold $\mathscr{T}$ fixed and solve for Δt via a 1DOF grid search on the same objective. This is repeated until convergence.

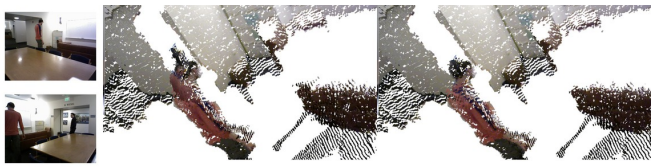

Fig. 3. **Left:** When a naive nearest-neighbor metric is applied, two sides of an object will be drawn toward each other, often at the expense of violating free space. **Right:** Our occlusion-aware energy term is able to recover from many of these situations.

Parameter	Description	Value
$b[\mathbf{z}]$	Depth binning function	$\lceil \mathbf{z}/10cm \rceil$
$d^{+}_{\max}$	Correspondence limit	10cm
$d^{-}_{\max}$	Penetration limit	3cm
ε	Minimum grid search step	0.01
K	Number of RANSAC iterations	1000
d_{inlier}	RANSAC inlier threshold	10cm

Fig. 4. A list of free parameters used in our approach, and the values used for our experiments.

D. Implementation Details

Our objective $\mathbf{E}[\mathscr{T}, \Delta t]$ requires a nearest-neighbor lookup over all points in all frames. To make this problem tractable, we randomly sample 5-frame fragments among those which exhibit motion. We have found that 50 such frames often suffice, putting the run time at an average of 10 minutes and 37 seconds. Also, when evaluating $\mathbf{E}[\mathscr{T}, \Delta t]$, to avoid double-counting background points, we align only foreground objects $\{O^{(t)}_{1,i}\}$ with the full cloud $\mathbf{C}^{(t)}_0$.

It is also well known that these sensors suffer from fairly high distortion at a distance, as observed by Herrera et al. [30]. To counter this effect, we preprocess our initial depth maps via a learned undistortion model, following the unsupervised calibration method proposed by Teichman et al. [31].

V. EXPERIMENTAL RESULTS

Our experiments were done with a pair of ASUS Xtion Pro Live sensors, plugged into separate machines. Shortly before recording each sequence, an NTP update was done to synchronize the frames. As mentioned previously, while this synchronizes the machines fairly well, a small time offset is typically incurred throughout the duration of the sequence.

Perfect ground truth is difficult to attain for this task, as 6DOF transforms and millisecond-level time offsets are too subtle for precise human labeling. For lack of a perfect method, then, we look to the *de facto* standard: checkerboards. For all configurations in which the sensors shared a wide enough field of view to make checkerboard calibration possible, we recorded a second calibration sequence with the sensors in the same pose. This employed a simple technique:

- Carry a checkerboard through the scene, as shown in Figure 5.
- Detect corners of the checkerboard in the registered RGB images of both sensors
- Project these corners into 3D space, and use RANSAC to estimate a transform.

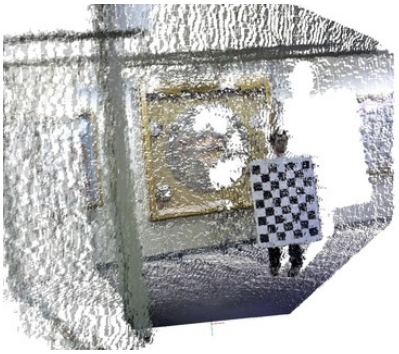

Fig. 5. Ground truth evaluation was provided by using a checkerboard to align the two sensors for each configuration. Quantitative results using this method can be seen in Figure 6.

Seq	Trans (cm)	Rot (°)	PtP (cm)	ChStd (cm/°)
01	3.43	0.70	3.86	±1.02 / ±0.35
02	2.75	0.49	2.49	±0.18 / ±0.05
03	4.67	1.23	4.67	±0.25 / ±0.08
04	10.49	1.54	6.57	±2.07 / ±0.33
05	4.21	1.30	4.56	±1.25 / ±0.25
06	5.11	0.19	4.62	±6.68 / ±1.20
Avg	5.11	0.91	4.46	
Std	±2.53	±0.48	±1.21	

Fig. 6. The results of our experiments on 6 sequences. **Trans** is the translation error. **Rot** is the minimal angle needed to rotate $\mathscr{T}_{\text{unsupervised}}$ into $\mathscr{T}_{\text{ground}}$. **P**oint **t**o **P**oint error refers to the average distance between corresponding points in the estimated and checkerboard-predicted frames, thresholded at 3m—beyond which readings are known to be noisy. **ChStd** is the approximate standard deviation of the checkerboard approach, using the Bootstrap method described in [32]. Note that this metric still does not capture the inherent *bias* of the checkerboard technique, and these numbers should not be taken as ground truth error; we urge those interested to visit `http://stanford.edu/~sdmiller/iros2013` and compare the pointclouds directly.

To estimate the noise inherent in this approach, we employed the Bootstrap method as described in [32], substituting the final transformation given by RANSAC for the mean.

We compared our unsupervised results with those given by a checkerboard in six sequences, as shown in Figure 6. As can be seen, they tend to agree with one another within 5cm and 1 degree. Note, however, that the bias inherent in the checkerboard approach makes it difficult to treat this comparison as a true error metric—the poorest scoring results are visually quite compelling. To illustrate, in Figure 8 we show the best and worst scoring results on the dataset.

We also perform a comparison between limited versions of our system, as shown in Figure 7. RANSAC yields the worst performance, showing that the refinement phase is quite crucial. Replacing the robust occlusion-aware energy term with the strict Nearest Neighbor metric of Equation 1 also leads to drastic performance reductions. Removing time optimization resulted in a significant error in one of the sequences, but otherwise the differences were well within

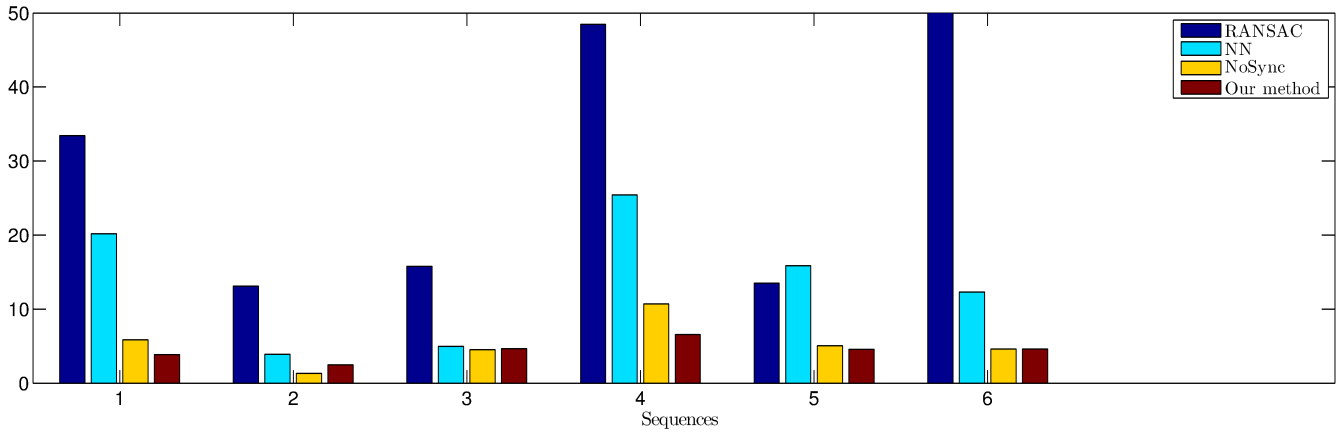

Fig. 7. Performance with different subcomponents removed. **RANSAC** does no refinement; the standard L2 Nearest Neighbor (**NN**) metric does not reason about occlusions and is prone to outliers, occasionally doing even worse than the unrefined estimate; when no synchronization (**NOSYNC**) is performed the results are fairly precise, likely because the synchronization error is fairly low to begin with.

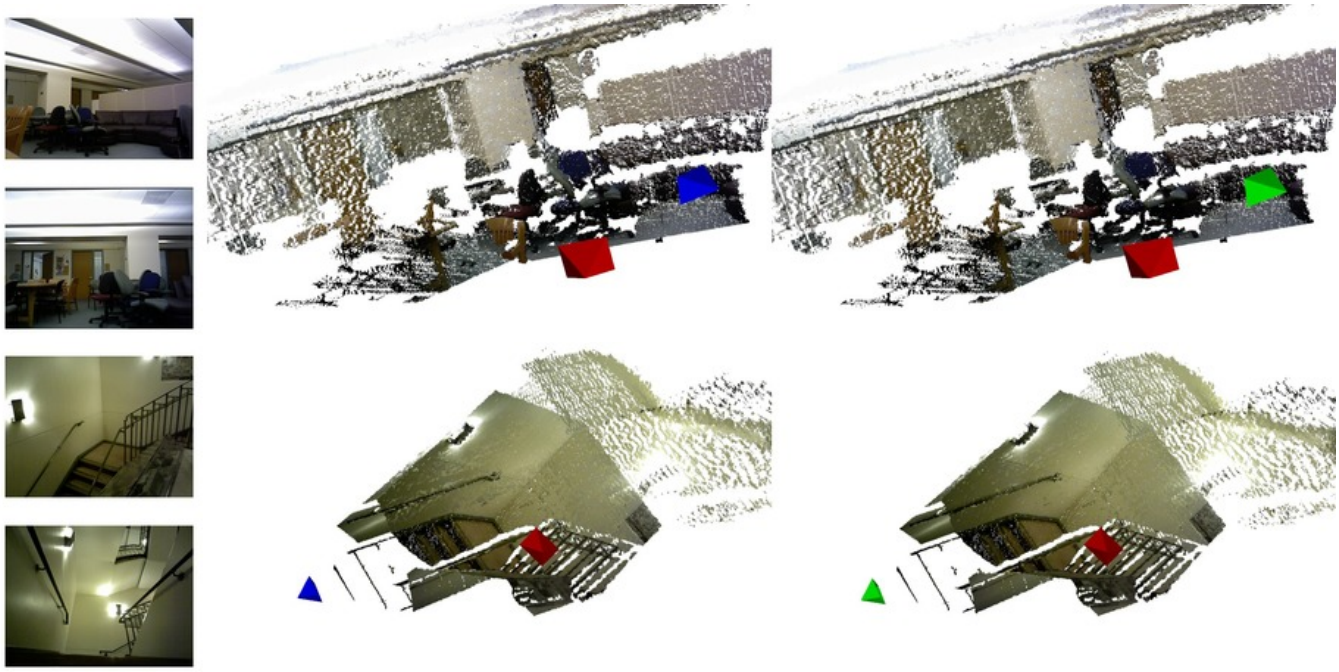

Fig. 8. The best (**top**) and worst (**bottom**) scoring calibration runs in our experiments. Sensor 0 is given in red, our unsupervised estimate of sensor 1 is given in blue, and the checkerboard estimate is given in green.

the noise of the calibration routine itself, suggesting that synchronization was not a serious issue in our dataset.

Figure 9 shows a number of results on our dataset. However, it is difficult to observe the reconstruction quality from single images. To better understand the performance of both approaches, we encourage those interested to visit `http://stanford.edu/~sdmiller/iros2013` and examine the 3D clouds directly.

One advantage of an unsupervised approach is that we are able to handle situations where, due to field-of-view constraints, checkerboard calibration is infeasible. See Figure 11 for an example. We note, however, that even an occlusion-aware energy suffers from "squishing" artifacts under such extreme field of view changes, as shown in Figure 10.

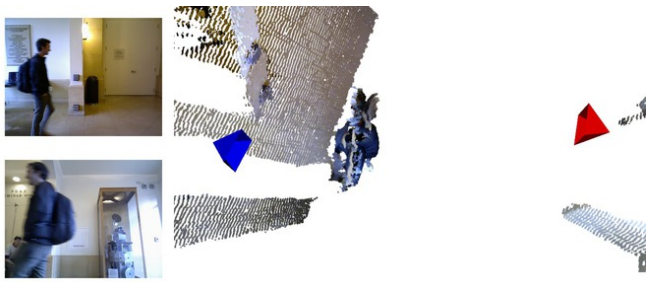

Fig. 10. Failure mode: when the angle between the cameras is extremely drastic and no thin objects are visible, our energy minimization approach still has a tendency to pull objects too tightly toward each other.

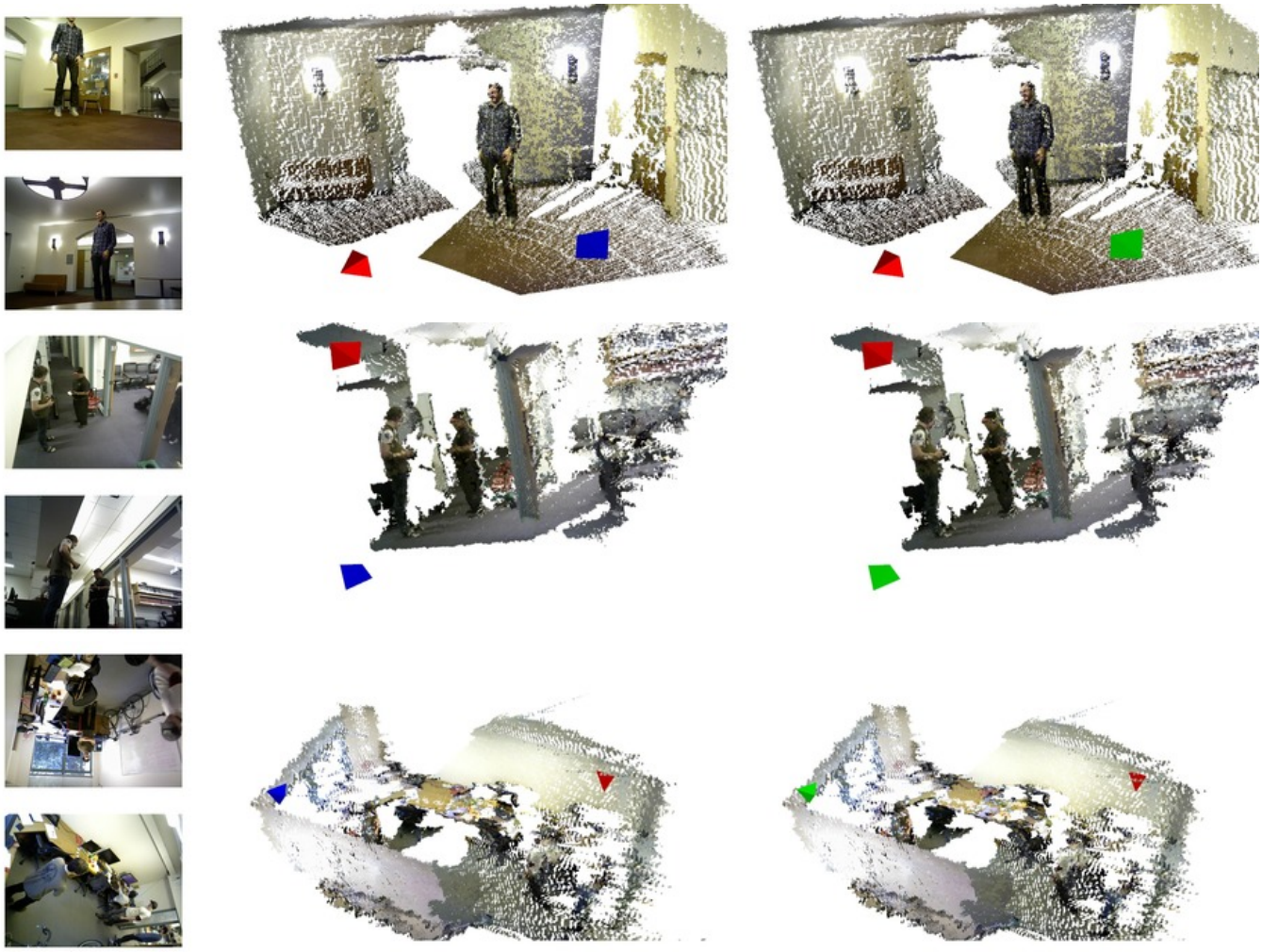

Fig. 9. Example results as compared to the checkerboard approach. **Left:** our unsupervised approach, **Right:** Results using a checkerboard. See website for details.

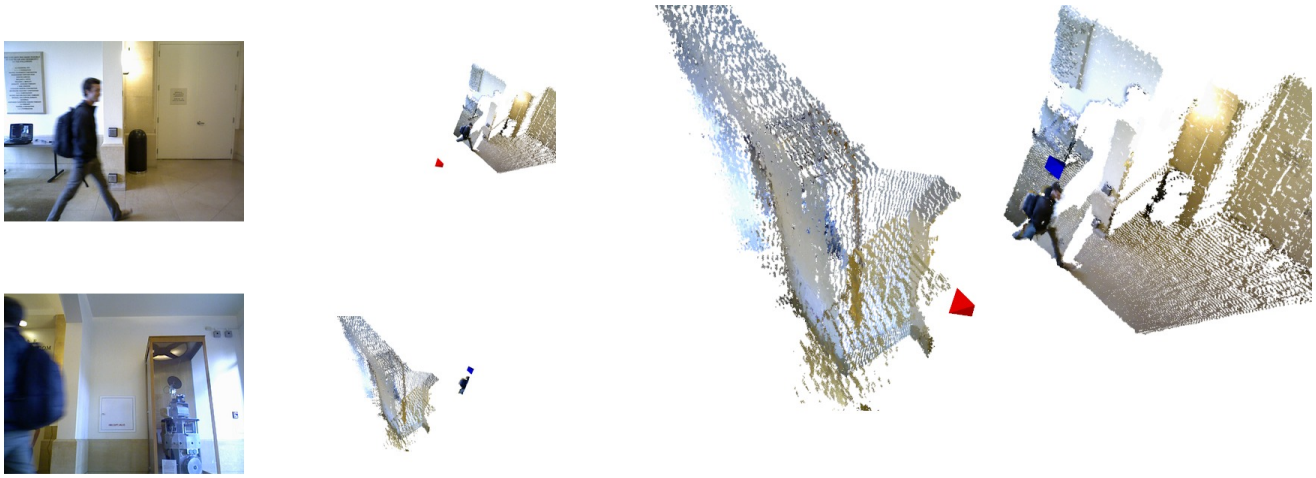

Fig. 11. An example result where few, if any, surfaces are visible to both sensors at the same time. Note that while this recovers the camera poses fairly well, it is not perfect: see Figure 10

VI. CONCLUSIONS AND FUTURE WORK

We propose a fully unsupervised technique for registering depth sensors in space and time, using the unstructured motion of foreground objects in a scene as our only cue. So doing, we eliminate the need to rely on distinctive calibration objects, textured environments and RGB registration, or indeed any intentional action at all. While this is meant to be used in environments where a manual routine would be impractical, we have shown it to perform competitively in those scenarios where both are feasible.

While this work dealt specifically with a pair of sensors, we recognize that it can be extended to—and made more reliable by—a higher number of sensors, via graph optimization techniques such as those in G2O [33]. In the future we hope to explore this extension.

VII. ACKNOWLEDGEMENTS

Stephen Miller is supported by the Stanford Graduate Fellowship and Google Hertz Foundation Fellowship.

REFERENCES

[1] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[2] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: an efficient alternative to sift or surf," in *International Conference on Computer Vision*, 2011.

[3] Z. Zhang, "A flexible new technique for camera calibration," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 11, pp. 1330–1334, 2000.

[4] J.-Y. Bouguet, "Camera calibration toolbox for matlab," 2004.

[5] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[6] G. G. Mateos, "A camera calibration technique using targets of circular features."

[7] T. Svoboda, D. Martinec, and T. Pajdla, "A convenient multicamera self-calibration for virtual environments," *Presence: Teleoperators & Virtual Environments*, vol. 14, no. 4, pp. 407–422, 2005.

[8] D. Demirdjian, A. Zisserman, and R. Horaud, "Stereo autocalibration from one plane," in *Computer VisionECCV 2000*. Springer, 2000, pp. 625–639.

[9] R. Horaud, G. Csurka, and D. Demirdjian, "Stereo calibration from rigid motions," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 12, pp. 1446–1452, 2000.

[10] C. Stauffer and K. Tieu, "Automated multi-camera planar tracking correspondence modeling," in *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, vol. 1. IEEE, 2003, pp. I–259.

[11] M. Pollefeys, R. Koch, and L. V. Gool, "Self-calibration and metric reconstruction inspite of varying and unknown intrinsic camera parameters," *International Journal of Computer Vision*, vol. 32, no. 1, pp. 7–25, 1999.

[12] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski, "Building rome in a day," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 72–79.

[13] S. Y. Bao, M. Bagra, Y.-W. Chao, and S. Savarese, "Semantic structure from motion with points, regions, and objects," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2012.

[14] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustmenta modern synthesis," in *Vision algorithms: theory and practice*. Springer, 2000, pp. 298–372.

[15] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "Dtam: Dense tracking and mapping in real-time," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2320–2327.

[16] C. Zhang and Z. Zhang, "Calibration between depth and color sensors for commodity depth cameras," in *International Workshop on Hot Topics in 3D*, 2011.

[17] J. Levinson and S. Thrun, "Unsupervised calibration for multi-beam lasers," in *ISER*, 2010.

[18] W. Maddern, A. Harrison, and P. Newman, "Lost in translation (and rotation): Rapid extrinsic calibration for 2D and 3D LIDARs," in *ICRA*, 2012.

[19] G. Pandey, J. R. McBride, S. Savarese, and R. Eustice, "Automatic targetless extrinsic calibration of a 3d lidar and camera by maximizing mutual information." in *AAAI*, 2012.

[20] K. Kodagoda, A. Alempijevic, J. Underwood, S. Kumar, and G. Dissanayake, "Sensor registration and calibration using moving targets," in *Control, Automation, Robotics and Vision, 2006. ICARCV'06. 9th International Conference on*. IEEE, 2006, pp. 1–6.

[21] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard, "An evaluation of the rgb-d slam system," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 1691–1696.

[22] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments," in *the 12th International Symposium on Experimental Robotics (ISER)*, vol. 20, 2010, pp. 22–25.

[23] T. Whelan, M. Kaess, M. Fallon, H. Johannsson, J. Leonard, and J. McDonald, "Kintinuous: Spatially extended kinectfusion," in *RGB-D Workshop at Robotics: Science and Systems (RSS)*, 2012.

[24] R. A. Newcombe, A. J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*. IEEE, 2011, pp. 127–136.

[25] E. Bylow, J. Sturm, C. Kerl, F. Kahl, and D. Cremers, "Real-time camera tracking and 3d reconstruction using signed distance functions," in *Robotics: Science and Systems (RSS)*, June 2013.

[26] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 3212–3217.

[27] A. Makadia, A. Patterson, and K. Daniilidis, "Fully automatic registration of 3d point clouds," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 1. IEEE, 2006, pp. 1297–1304.

[28] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*. IEEE, 2001, pp. 145–152.

[29] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp."

[30] D. Herrera C., J. Kannala, and J. Heikkilä, "Accurate and practical calibration of a depth and color camera pair," in *Computer Analysis of Images and Patterns*, 2011, pp. 437–445.

[31] A. Teichman, S. Miller, and S. Thrun, "Unsupervised intrinsic calibration of depth sensors via slam," in *Robotics: Science and Systems (RSS)*, 2013.

[32] B. Efron and R. Tibshirani, "Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy," *Statistical Science*, vol. 1, no. 1, pp. 54–75, 1986.

[33] R. Kuemmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *ICRA*, 2011.