Combining Inertial Navigation and ICP for Real-time 3D Surface Reconstruction

M. Nießner and A. Dai and M. Fisher

Stanford University

Abstract

We present a novel method to improve the robustness of real-time 3D surface reconstruction by incorporating inertial sensor data when determining inter-frame alignment. With commodity inertial sensors, we can significantly reduce the number of iterative closest point (ICP) iterations required per frame. Our system is also able to determine when ICP tracking becomes unreliable and use inertial navigation to correctly recover tracking, even after significant time has elapsed. This enables less experienced users to more quickly acquire 3D scans. We apply our framework to several different surface reconstruction tasks and demonstrate that enabling inertial navigation allows us to reconstruct scenes more quickly and recover from situations where reconstructing without IMU data produces very poor results.

1 Introduction and Related Work

With the advent of commodity real-time RGB-D sensors such as the Microsoft Kinect and the Asus Xtion, 3D reconstruction has gained new momentum in the computer graphics and vision community. In particular, online reconstruction approaches that involve real-time volumetric fusion have received significant attention [CL96, NIH*11, RV12, WJK*12, CBI13, NZIS13]. These methods require frame-to-frame tracking in order to align input scan data. However, achieving high-quality alignments is challenging. One approach is to use visual SLAM (simultaneous localization and mapping) [Dav03, KM07, KRD08, NLD11]. Computationally cheaper tracking can be realized by leveraging depth data provided by RGB-D cameras, typically using a variant of the iterative closest point algorithm (**ICP**) [BM92, CM92]. While there are depth tracking approaches beyond ICP (e.g., [HJS08]), ICP has been established in particular in the context of volumetric fusion. ICP works by projectively aligning adjacent frames to determine correspondences between depth values and solving for the corresponding affine transformation. While ICP variants provide suitable tracking results in some scenarios, they fail at scanning scenes lacking sufficient geometric detail and moderately large frame-to-frame motion. Thus, 3D scanning applications such as Kinect Fusion (available in the Kinect SDK) are very limited in practice and require significant user training.

In this work we focus on making real-time scanning accessible to non-expert users by improving the robustness of ICP tracking. We specifically consider scenarios where depth-based tracking fails, such as large inter-frame mo-

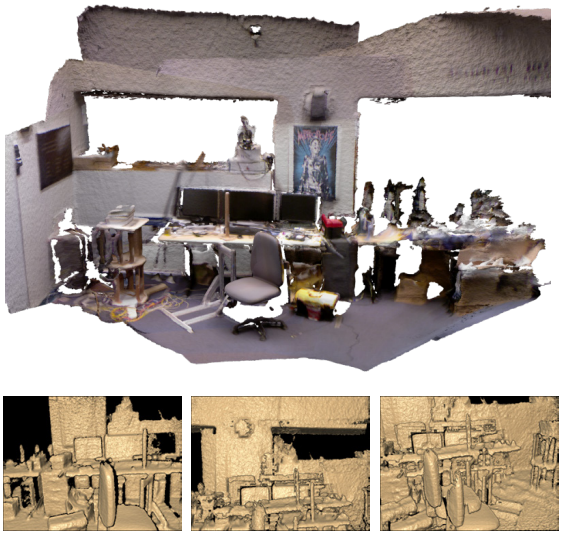


Figure 1: *Top: scene captured using a Kinect RGB-D sensor and reconstructed using inertial navigation for computing inter-frame alignment in real-time. Bottom: failure case for alignment with the same data in the absence of inertial navigation. When ICP computes an incorrect alignment, an invalid fusion step creates significant visual artifacts (middle, right) and the algorithm is not able to recover.*

© The Eurographics Association 2014.

tion and planar surfaces (e.g., walls and floors). Most existing online scanning methods cannot detect or recover from poor ICP tracking, resulting in significant visual artifacts like multiple overlapping copies of the scene [BGC13].

Our approach is to achieve high-quality alignments by combining the ICP algorithm with sensor readings from an inertial measurement unit (IMU). An IMU measures inertial forces, typically using gyroscopes to measure angular velocity and accelerometers to measure linear acceleration. More sophisticated IMUs may contain other sensors, such as a magnetometer to correct for orientation drift and a barometer to provide altitude. IMUs are now ubiquitous as they are integrated into many popular smartphones.

This paper presents our method for incorporating inertial navigation into a surface reconstruction framework:

- We introduce metrics to evaluate ICP tracking quality and determine when ICP tracking becomes unreliable.
- We show how to use IMU data to improve ICP quality and reduce the number of ICP iterations needed, in some cases requiring only a single ICP iteration per frame.
- We demonstrate that using an IMU for dead reckoning enables real-time 3D scanning to correctly recover when ICP tracking fails.

2 ICP Failure Analysis

In order to perform camera tracking for data fusion, we employ the rigid point-to-plane ICP variant [CM92]. The goal of ICP is to determine the cumulative frame-to-frame transform $\mathbf{M}$ which is composed of a rotation and translation $\mathbf{T}(t_x,t_y,t_z)\cdot\mathbf{R}(\alpha,\beta,\gamma)$. In this work, we further need to determine the quality of an ICP match so that we can use inertial navigation when ICP is unreliable. In the remainder of this section, we detail the specifics of our ICP quality metric.

We start by obtaining weighted (based on normal variation and distance) projective correspondences $\{\mathbf{s}_i,\mathbf{d}_i\}$ between the current and the last observed frame. These define the non-linear least squares minimization problem

$$\mathbf{M}_{opt} = \arg\min \sum_i (w_i(\mathbf{M}\cdot\mathbf{s}_i - \mathbf{d}_i)\cdot\mathbf{n}_i)^2$$

with $\mathbf{T}$ being a 3×4 translation matrix, and $\mathbf{R}$ a 3×4 rotation matrix with $\mathbf{R}(\alpha,\beta,\gamma) = \mathbf{R_z}(\gamma)\cdot\mathbf{R_y}(\beta)\cdot\mathbf{R_x}(\alpha)$.

Following Low [Low04], we linearize the rotations (assuming small rotation angles):

$$\mathbf{M} = \mathbf{T}(t_x,t_y,t_z)\cdot\mathbf{R}(\alpha,\beta,\gamma) \approx \begin{pmatrix} 1 & -\gamma & \beta & t_x \\ \gamma & 1 & -\alpha & t_y \\ -\beta & \alpha & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} = \hat{\mathbf{M}}$$

We now rearrange $(\mathbf{M}\cdot\mathbf{s}_i - \mathbf{d}_i)\cdot\mathbf{n}_i$ into a linear system $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{x} = (\alpha,\beta,\gamma,t_x,t_y,t_z)^T$. Next, we solve for $\mathbf{x}$ in the optimal least squares formulation $\mathbf{A^T Ax} = \mathbf{A^T b}$. We accomplish this using a parallel reduction on the GPU [HSO07] to build both $\mathbf{A^T A}$ (6×6), and $\mathbf{A^T b}$ (6×1) for all valid correspondence pairs $\{\mathbf{s}_i,\mathbf{d}_i\}$. We then solve the linear 6×6 system for $\mathbf{x}$ on the CPU using a singular value decomposition which allows us to compute $\mathbf{M}\cdot\mathbf{x}$. Since $\mathbf{x}$ approximates the solution of the non-linear least squares system, we iterate this process until convergence.

To determine ICP quality, we incorporate the residual $\mathbf{r} = ||\mathbf{Ax}-\mathbf{b}||_2^2$, the number of non-rejected correspondences n, and the summed confidence weight $\sum_i w_i$ (that are used to weight the rows of $\mathbf{A}$) into the GPU reduction at each ICP step. We further determine the condition of the system matrix $\mathbf{A}$ using the previously obtained singular values $\kappa(\mathbf{A}) = \frac{\sigma(\mathbf{A})_{max}}{\sigma(\mathbf{A})_{min}}$. This specifies the descriptiveness of the geometric features; e.g., if the algorithm aligns two planar shapes, $\sigma(\mathbf{A})_{min}$ will be close to zero, and in the limit $\kappa(\mathbf{A}) \to \infty$ which means that the system becomes underconstrained and $\mathbf{A}$ is ill-conditioned. This obtained ICP error allows us to determine ICP convergence, and when to stop iterating. We further employ empirically determined error thresholds to identify lost tracking states as required by our inertial navigation approach described in Section 3.

3 ICP Correction using IMU Data

IMUs are comprised of many different sensor types. For this work, we only assume that the IMU is able to estimate the rigid motion of the scanner over a time range. We represent this with a function $InertialEstimate(t_a,\ t_b)$, which returns a matrix representing the rigid transformation of the sensor's coordinate frame from time t_b to t_a. If the IMU has no error, this transform is sufficient to perform surface reconstruction and ICP is unnecessary. In practice, the IMU estimate accumulates error from many different sources such as sensor noise, global drift, and low sampling rates.

3.1 Improved ICP Initialization

Typically, the ICP algorithm for frame t is initialized with the transform computed for frame $t-1$. This approach can converge to a good result if the motion between frames is small, the previous frame's transform is accurate, and there are strong features in the frame. Otherwise convergence often becomes slow and a bad local minima is reached.

We use the rigid transform estimated by inertial navigation to provide a significantly better initial guess for the ICP algorithm. We compute Δ_{IMU}, the inertial estimate from frame $t-1$ to frame t. We use the previous frame's transform followed by Δ_{IMU} as our initial seed for ICP. When Δ_{IMU} is accurate, this allows ICP to converge to the correct transform in a very small number of iterations. It also improves robustness in cases when ICP produces many possible solutions, such as nearly planar regions. In Section 4, we show that typically only a single ICP iteration is needed when using the inertial estimate.

© The Eurographics Association 2014.

M = Identity()
loop
  Δ_{IMU} = *InertialEstimate*(t_i, t_{i-1})
  [rigidICP, tracking] = *ICP*(scan, **M** * Δ_{IMU})
  if (tracking)
    Fusion(scan, rigidICP)
    M = rigidICP
  else
    M = **M** * Δ_{IMU}

Figure 2: *Pseudocode for surface reconstruction using ICP augmented with inertial navigation. The inertial motion estimate is used both as an improved starting seed for the ICP algorithm, and for dead reckoning when ICP tracking becomes unreliable. This allows the method to eventually recover when scanning regions which are difficult for ICP to track.*

Figure 3: *Our experimental setup. Left: components from left to right are a smartphone, battery, Kinect, and laptop. Right: Kinect, phone, and battery fixed together for scanning.*

3.2 Tracking Recovery

We also use inertial navigation to estimate the motion of the scanner in situations where the ICP algorithm is not able to produce a reliable motion estimate. For frames in which we detect that ICP tracking is invalid using the method described in Section 2, we ignore the transform computed by ICP and accumulate the motion estimated using Δ_{IMU}. This greatly assists the system in recovering from areas where tracking with ICP is not possible. Once the scanner points to an unreliable area, the IMU takes over and continues to provide position estimates using dead reckoning until ICP is able to converge to a previously scanned region in the scene. With accurate IMU data, this can allow the scan to correctly recover even when the scanner has undergone significant displacement. In Figure 1 and the accompanying video, we show that the ability to detect and recover from lost tracking produces coherent reconstructions in situations where ICP without inertial navigation fails.

Pseudocode for both our ICP initialization and tracking recovery is shown in Figure 2.

4 Results

4.1 Experimental Setup

We apply our system to demonstrate the benefits of incorporating inertial navigation when scanning environments. For all the results in this paper, we use an XBox 360 Kinect as an RGB-D camera and a Samsung Galaxy S4 as an IMU. Both hardware items are easily available, and many other brands of cellphones contain similar inertial sensors. Our Kinect and phone were rigidly held together as shown in Figure 3 (no calibration required).

Our results use only the inertial sensor readings provided by the Android SDK, which does no significant sensor fusion or filtering of the raw sensor signal. After experimentation, we found the gyroscope sensor effective for measuring changes in orientation, but found the accelerometer produces very poor translation estimates. In order to determine translation from the accelerometer readings, gravitational acceleration must be factored out and the resulting signal must be integrated twice. In practice, most users have found the resulting error to be too large to be useful [Sac10]. While more specialized IMUs are able to produce translation estimates with better accuracy, we did not find this to be necessary to demonstrate the benefits of inertial navigation.

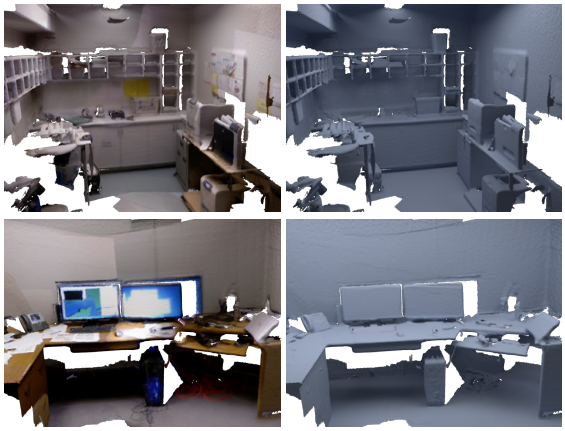


Figure 4: *Scenes reconstructed using the ICP algorithm augmented with inertial navigation from an Android Galaxy Nexus S4.*

4.2 Applying Inertial Navigation

In Figure 1, we show a comparison of a reconstruction on a building interior with and without IMU data. With inertial navigation the system was able to approximately reconstruct the environment. Without any inertial information the ICP algorithm resulted in incorrect inter-frame alignments. These repeated misalignments ultimately prevent the algorithm from producing a coherent mesh.

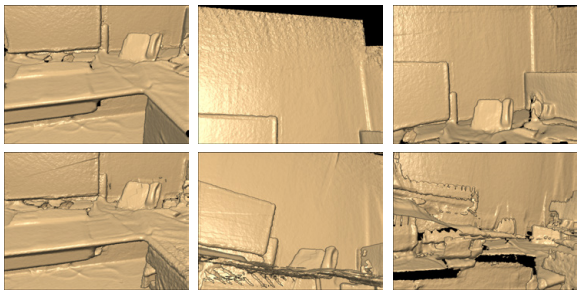
In Figure 4, we show two other scenes reconstructed using inertial navigation. For scenes with high detail such as the top scene in this figure, a single ICP iteration per frame combined with IMU data was sufficient to reconstruct the scene correctly. For scenes with featureless regions such as the bottom scene, in the absence of IMU data our ICP algorithm with 30 iterations per frame lost tracking and was not

© The Eurographics Association 2014.

able to correctly align the frames, resulting in a decoherent result.

4.3 Recovering Tracking with Dead Reckoning

As discussed in Section 3.2, we use inertial navigation to recover from regions where ICP tracking fails. We show an example of such a recovery in Figure 5. The algorithm lost tracking in the middle image, but with inertial dead reckoning was able to recover. Without IMU data, even with additional ICP iterations the algorithm was not able to converge correctly when encountering the featureless wall. Severe misalignment occurred when the scanner moved back down.



Frame N Frame N + 35 Frame N + 70

Figure 5: *Using inertial dead reckoning to recover from lost tracking. Top: three frames from a reconstruction using inertial navigation and 3 ICP iterations per frame. Bottom: three frames from a reconstruction without IMU data and 20 iterations per frame.*

5 Conclusion

We presented a practical approach to using inertial navigation to improve dense scene reconstruction tasks. Our technique improves robustness to regions that are difficult for ICP to track, achieves good results with significantly fewer ICP iterations, and works with commonplace hardware. There are many different variations of real-time scene reconstruction algorithms beyond the method explored in this paper, but we believe that our work can improve alignment for many existing approaches. However, our system still has many failure cases that can be built upon.

One problem with our system is that our inertial estimate relies upon integrating commodity accelerometer readings, which results in a very poor translation estimate. One approach that could produce a much more accurate translation estimate without expensive hardware is to compute the optical flow between adjacent frames in the RGB-D camera or the phone's camera. This optical flow can then be used to estimate linear velocity and could be further improved by factoring out the inter-frame rotation using the gyroscope. With a better translation estimate, the system becomes significantly more robust to failures in the ICP alignment.

Another significant problem with most real-time scene reconstruction systems is global drift, where the system does not correctly re-align with a previously encountered area, resulting in multiple, misaligned copies of the scene, which is a significant problem when scanning in a loop (loop closure). Many smartphones also contain global positioning units that provide a noisy estimate of the sensor's absolute position. By fusing this global position with our relative position computed via inter-frame alignment, it should be possible to mitigate the problem of global drift. Real-time loop closure with GPS is a challenging problem and an exciting area for further development.

References

[BGC13] BEN GLOCKER SHAHRAM IZADI J. S., CRIMINISI A.: Real-Time RGB-D Camera Relocalization. In *Proc. IEEE Int. Symp. Mixed and Augmented Reality* (2013), pp. –. 2

[BM92] BESL P., MCKAY N.: A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. and Mach. Intell. 14*, 2 (1992), 239–256. 1

[CBI13] CHEN J., BAUTEMBACH D., IZADI S.: Scalable real-time volumetric surface reconstruction. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 113. 1

[CL96] CURLESS B., LEVOY M.: A volumetric method for building complex models from range images. In *In Proc. Computer graphics and interactive techniques* (1996), ACM, pp. 303–312. 1

[CM92] CHEN Y., MEDIONI G.: Object modelling by registration of multiple range images. *Image and Vision Computing 10*, 3 (1992), 145–155. 1, 2

[Dav03] DAVISON A. J.: Real-time simultaneous localisation and mapping with a single camera. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on* (2003), IEEE, pp. 1403–1410. 1

[HJS08] HUHLE B., JENKE P., STRASSER W.: On-the-fly scene acquisition with a handy multi-sensor system. *International Journal of Intelligent Systems Technologies and Applications 5*, 3 (2008), 255–263. 1

[HSO07] HARRIS M., SENGUPTA S., OWENS J. D.: Parallel prefix sum (scan) with cuda. *GPU gems 3*, 39 (2007), 851–876. 2

[KM07] KLEIN G., MURRAY D.: Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on* (2007), IEEE, pp. 225–234. 1

[KRD08] KAESS M., RANGANATHAN A., DELLAERT F.: isam: Incremental smoothing and mapping. *Robotics, IEEE Transactions on 24*, 6 (2008), 1365–1378. 1

[Low04] LOW K.-L.: *Linear least-squares optimization for point-to-plane ICP surface registration*. Tech. rep., Chapel Hill, University of North Carolina, 2004. 2

[NIH*11] NEWCOMBE R. A., IZADI S., HILLIGES O., MOLYNEAUX D., KIM D., DAVISON A. J., KOHLI P., SHOTTON J., HODGES S., FITZGIBBON A.: KinectFusion: Real-time dense surface mapping and tracking. In *Proc. IEEE Int. Symp. Mixed and Augmented Reality* (2011), pp. 127–136. 1

[NLD11] NEWCOMBE R., LOVEGROVE S., DAVISON A.: DTAM: Dense tracking and mapping in real-time. In *Proc. IEEE Int. Conf. Comp. Vision* (2011), pp. 2320–2327. 1

[NZIS13] NIESSNER M., ZOLLHÖFER M., IZADI S., STAMMINGER M.: Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (TOG) 32*, 6 (2013), 169. 1

[RV12] ROTH H., VONA M.: Moving volume KinectFusion. In *British Machine Vision Conf.* (2012). 1

[Sac10] SACHS D.: Sensor fusion on android devices: A revolution in motion processing, Dec. 2010. URL: http://davidcrowley.me/?p=370. 3

[WJK*12] WHELAN T., JOHANNSSON H., KAESS M., LEONARD J., MCDONALD J.: *Robust Tracking for Real-Time Dense RGB-D Mapping with Kintinuous*. Tech. rep., 2012. Query date: 2012-10-25. 1

© The Eurographics Association 2014.