Department of Computer Science
George Mason University Technical Reports

4400 University Drive MS#4A5
Fairfax, VA 22030-4444 USA
http://cs.gmu.edu/ 703-993-1530

Dynamic RGB-D Mapping

Michael Paton mpaton@gmu.edu
Jana Košecká kosecka@cs.gmu.edu

Technical Report GMU-CS-TR-2012-1

Abstract

Localization and mapping has been an area of great importance and interest to the robotics and computer vision community. It has traditionally been accomplished with range sensors such as lasers and sonars. Recent improvements in processing power coupled with advancements in image matching and motion estimation has allowed development of vision based localization techniques. Despite much progress, there are disadvantages to both range sensing and vision techniques making localization and mapping that is inexpensive and robust hard to attain. With the advent of RGB-D cameras which provide synchronized range and video data, localization and mapping is now able to exploit both range data as well as RGB features. This thesis exploits the strengths of vision and range sensing localization and mapping strategies and proposes novel algorithms using RGB-D cameras. We show how to combine existing strategies and present through evaluation of the resulting algorithms against a dataset of RGB-D benchmarks. Lastly we demonstrate the proposed algorithm on a challenging indoor dataset and demonstrate improvements where either pure range sensing or vision techniques perform poorly.

1 Introduction

The ability for autonomus robots to localize in novel environments and generate accurate 3D maps has been of great interest, and heavily researched by the robotics community. Traditional mapping was performed with sensors such as lasers and sonars. While successful solutions have been attained [1] , they either required expensive laser range finders or they were demonstrated with less expensive sonars in very simplistic environments [2]. The increase in the processing speed as well as advancements in vision based motion estimation and invariant feature based matching has fueled the progress in vision based localization. Vision based mapping takes advantage of visual features, matching them frame by frame to compute accurate relative poses between them. While vision based localization has made great strides [3], with the exception of few [4] the focus on the large scale 3D reconstruction has been limited and the models obtained are often characterized by a sparse set of features.

More recently, efforts have been made to combine range finding sensors with vision sensors to overcome the weaknesses of both. Research in this area has been revived by availability of RGB-D cameras, which provide video (RGB) data just the same as a normal camera, along with per-pixel depth information.

The goal of this thesis is to exploit the strengths of previous strategies which use range and vision sensing and to provide a robust solution for localization and mapping with an RGB-D sensor. This will be accomplished by combining mapping algorithms using range information and visual features to obtain a robust and accurate system, capable of coping with environments where either pure visual or range sensors fail.

Overview Related work is detailed in Chapter II. Preliminaries for RGB-d mapping are discussed in Chapter III. Chapter IV analyzes the proposed RGB-D mapping algorithms. Testing and experimental process with the RGB-D data set as well as results of the experiments are reviewed in Chapter V. Chapter VI provides closure on the results as well as future work.

2 Related Work

Existing simultaneous localization and mapping (SLAM) research has differed in sensor modality, the types of maps they strive to create, and their use. Common systems use modalities such as lasers, camera vision or both to acquire metric, topological or hybrid maps. Metric mapping algorithms have ranged from on-line recursive update strategies, pairwise non-linear motion estimation strategies, loop closure [5], and global optimization methods [1].

In this work we focus on metric SLAM using RGB-D cameras, this problem can be broken down into three sub-problems: data association and motion estimation from two views, loop closure detection, and globally consistent motion estimation.

While the latter portion is often similar regardless of modality, data association and motion estimation differ. An overview of the techniques for globally consistent motion estimation can be found in [1]. Considering that RGB-D sensors have characteristics of range finding devices and camera based vision systems, techniques for motion estimation between two frames have been adopted from previous research on both modalities, as described in the following sections.

Laser Mapping Until recently laser range finders were the sensor of choice for localization and mapping. 3D laser range finders are able to provide accurate and fast 3D depth information. Using registration techniques such as [6], robotic systems are able to quickly compute transformations between corresponding laser scans. Laser based mapping work done by [7] is thought to be the original work on globally consistent mapping with laser range finders. This work involves collecting local pose information between laser scans, either by scan matching or odometry. The poses and scans are kept in memory and are globally aligned using a maximum likelihood criterion. These methods benefit from loop constraints, which can greatly improve the local pose estimates.

More recently [8] uses scan matching and Rao-Blackwellized particle filtering. There has been a large number of works published on problems of data associations in the context of range sensing; ranging from data associations of raw scans to various feature based methods. Some discussion of the topic can be found in [9]. One of the key components of the scan matching stage is the so called Iterative Closest Point (ICP) algorithm originally proposed by [10]. Our Proposed method will be based on a variation of an ICP algorithm and will be described more in detail in Chapter III.

While laser range finders can create highly accurate metric maps, there are some drawbacks. With laser range finders accuracy often decreases with the price of the sensor. Less expensive sensors will often provide ambiguous motion estimates when displacements are large. Another drawback is the lack of visual information in the outputted 3D map, which is often not informative to the user.

Vision Mapping The need to robustly differentiate between scenes with similar geometry and obtain richer models has spurred research in vision based mapping. With the advances in scale invariant image feature matching such as [11] and [12], research in vision based frame matching, loop closure detection, and 3D mapping led to several successful solutions to the vision based localization problem. Vision mapping research has seen use of a large variety of methods. The use of invariant features coupled with increases in computational power has allowed efficient searches in large databases as well as accurate mapping of features between frames. Using a single perspective camera, [13] creates a topological representation online by adding images to a database ad creating a link graph. An image matching scheme then allows for mapping and localization. Due to the compact representation, this method can maintain a graph containing millions of images in real time. A SLAM technique developed by [14] utilizes a stereo vision system to achive local pose estimation. This method relys solely on input images from the stereo camera. In this method visual correspondences are calculated using scale invariant features, and motion estimation is calculated via the RANSAC algorithm.

Along with the vision sensing modality comes a list of unique drawbacks. Maps generated by vision based modalities are sparser then those generated by range finders. This is attributed to a lower data rate, and the use of correspondences by means of visual features which can be sparse in certain environments. Perhaps the most critical weakness of vision based mapping is also its greatest strength, visual features. There are certain texture-less environments where there are a distinct lack of the crucial visual features that are needed to generate correspondences. An example of this is the common office hallway, white walls and often featureless carpets. In this situation it is very difficult for a vision system to match features between frames, especially in the precense of large motions.

Multi Sensor Mapping The common drawbacks of both range finder and vision based modalities, spurred research into techniques that utilize both laser and vision sensors. A recent example of this is [15], who uses both a laser scanner and a camera installed on a mobile robot to incrementally build a 3D metric map of the environment. The map is built from laser generated point clouds. Images taken from the camera are used for loop closure detection. This method of loop closure is tolerant of repetitive visual structure and takes advantage of the strengths of both laser and vision data. The combination of both sensors allows for highly accurate odometry estimation from the laser data coupled with robust and quick loop estimation from the camera, effectively making the fusion approach more effective then either sensor alone.

These techniques may overcome some of the obstacles of range finding and vision sensors, but they still rely on expensive laser sensors, additionally it can be difficult to associate 3D range data with the collected vision data. Previous techniques which use both laser and vision required calibration of the two sensors [16], in order to

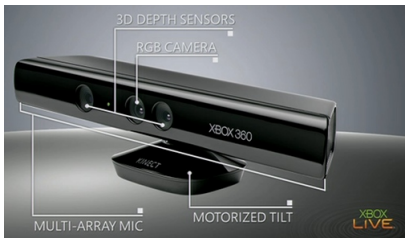

Figure 1: Microsoft Kinect for XBOX-360

Figure 2: Example RGB-D Frames

accurately register the two sensing modalities and hence making the combined sensors less flexible.

2.1 RGB-D Mapping

RGB-D sensors solve the problem of data correspondence found in multi sensor mapping by providing RGB images synchronized with per-pixel depth information. RGB-D sensors are able to provide both visual and 3D depth information, which allow us to exploit the direct correspondence of range data to visual features. This allows for direct correspondence of 3D range data, by means of extracting and matching visual features between frames. Being a relatively new sensor, research in RGB-D mapping is in its infancy. Current examples of RGB-D mapping techniques are [17] and [18]. Details of both of these algorithms can be found in Section 3.4.

2.1.1 RGB-D sensor details

For our experiments we used the Microsoft Kinect©sensor, it is an RGB plus depth sensor capable of providing both RGB images per-pixel depth information at a 30Hz rate. Due to its relatively low price vs the quality of its output it is ideal for robotics research. A disadvantage of the sensor is its limited field of view of 57° horizontally and 43° vertically. This is in contrast to current laser range scanners which often provide scans of 180° or 360° scans. The Kinect also has a motor on its vertical axis capable of tilting the sensor with a range of motion of 54°.

The RGB camera The RGB camera uses 8-bit VGA 640x480 resolution, along with a Bayer color filter to provide 30 frames per second.

The Depth sensor The Depth sensor is capable of supplying 11-bit per-pixel depth information corresponding with the video stream. This is achieve by an infrared projector along with a CMOS (complimentary metal-oxide semiconductor). A major benefit of this approach is the kinect's ability to calculate depth data regardless of the lighting conditions. The depth sensor has a range of 1.2-3.9m.

3 Preliminaries

In this chapter we will describe the component algorithms needed to perform data association and motion estimation between two views. This chapter acts as a primer, describing in detail the algorithms and techniques needed for RGB-D SLAM. The individual components of the data association and motion estimation techniques are as follows:

1. Feature extraction and matching.
2. Coordinate conversion
3. Outlier rejection, correspondence registration.
4. 3D Point cloud Registration.

There are several alternatives of how to carry out the data association and motion estimation using two consecutive frames. In purely visual based approaches the two view motion estimation is typically proceeded by extracting and matching features in the consecutive views, followed by robust motion estimation based on epipolar geometry [19]. This technique enables estimation of the relative pose $T = (R, t)$ between two views, using closed form strategy followed by non-linear refinement. Due to the absence of 3D information the translation component can only be estimated up to a scale. The first stage of the methods is the process of extraction of robust scale invariant features (SIFT) and their associated descriptors. Towards this end we choose commonly used SIFT features.

The SIFT algorithm [11] developed by David Lowe, extracts and matches visual features and their associated descriptors between frames, it is described in section 3.1. The putative matches are then converted from 2D image coordinates to 3D world coordinates. The matching 3D point clouds are then refined by the robust RAndom SAmple Consensus (RANSAC) algorithm. The RANSAC algorithm estimates an initial transformation as well as outlier rejection based on initial feature matches obtained by SIFT matching, it is described in Section 3.2. Depending on the environment, the RANSAC transformation may contain a transformation with a great amount of error. If the RANSAC error is too high the transformation is corrected by the generalized ICP algorithm, which is described in section 3.3.1.

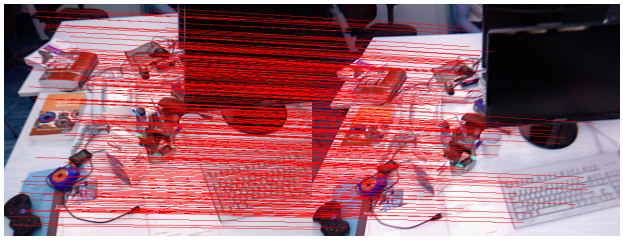

Figure 3: SIFT Matching

3.1 Feature Extraction and Matching

SIFT is an algorithm used to extract point features and their associated descriptors from images. Features extracted from SIFT are invariant to changes in the image, such as scaling, rotation, and translation. Once features are extracted from an image, they can be easily matched up in subsequent frames. Once features are extracted from an image, they can easily be matched in subsequent frames.

A common problem with matching frames with SIFT is a lack of features between frames. Another problem is features that are mismatched. These problems depend to a large extent on the type of environment, office hallways and areas with low lighting are common examples of environments with limited features. In order to successfully estimate motion between frames based off of visual features, mismatches must be ignored. In the case of having a small amount of features, an entirely different method might be needed.

3.2 RANSAC

RANdom SAmpling and Consensus (RANSAC) [20] is an iterative method for estimating mathematical models in the presence of data which contains outliers. The general RANSAC algorithm is detailed in figure 4. The inputs to RANSAC are a mathematical model and data which the model is to be fitted to. The method works by iteratively selecting random samples of the dataset and fitting the model to that subset of data. Fitting the model generates a hypothesis model, which is then tested against the entire dataset. RANSAC is often used to extract inliers from associated range scans, as well as generate an initial motion estimate between the two scans. For our case we use RANSAC to estimate motion between frames based off of corresponding 3D point clouds. The corresponding clouds are obtained by matching SIFT features and converting the 2D matches from image coordinates to 3D world coordinates with the Kinect's depth data. RANSAC is able to estimate a motion between the two point clouds while simultaneously removing likely mismatches.

For our purposes we use RANSAC to estimate motion corresponding 3D point clouds, as well as eliminating outliers from the correspondences. The corresponding

RANSAC[data,Model]

for $itr = 0 \rightarrow maxIterations$ **do**
  $sample \leftarrow extractSamples(model)$
  for $points \in data \notin sample$ **do**
    $\hat{p} \leftarrow$ {point in data transformed by T_0}
    $p \leftarrow$ {corresponding point in model}
    $err \leftarrow ||\hat{p} - p||^2$
    if $err \leq thresh$ **then**
      {Add p to inliers}
    end if
  end for
  {If there are enough candidate inliers, then this is a valid model}
  if $sizeof(candInliers) \geq minSize$ **then**
    $T \leftarrow estimateModel(inliers)$
    $(\hat{p}, p) \leftarrow$ {inlier elements in Model and Data}
    $estimateErr \leftarrow \sum_{i=0}^{m} |T \cdot \hat{p}_i - p_i|^2$
    if $globalErr \leq bestErr$ **then**
      $BestModel \leftarrow proposedModel$
      $BestErr \leftarrow globalErr$
    end if
  end if
end for

Figure 4: Algorithm: RANSAC

clouds are obtained by matching SIFT features and converting the 2D matches from pixel coordinates (i, j) to 3D world coordinates (x, y, z), using the following formula:

$$z \leftarrow depth(i, j)$$
$$y \leftarrow (j - c_y) \cdot (z / f_x)$$
$$x \leftarrow (i - c_x) \cdot (z / f_y)$$

The RGB-D sensor provides per-pixel depth information, therefore given a set of pixel coordinates (i, j) the z coordinate is simply extracted from the depth data. The (x,y) coordinates can then be calculated based off of the camera's intrinsic matrix, with (c_x, c_y) being the optical center of the camera, and (f_x, f_y) being the focal length of the camera.

3.3 Iterative Closest Point

Iterative closest point (ICP) is a technique used to register point clouds, it was first developed by [10]. The ICP algorithm can be partitioned into the following steps...

1. given point clouds $\{a_i\}$ and $\{b_i\}$ from two scans, find the corresponding points between the scans.
2. Compute a rigid body transformation between $\{a_i\}$ and $\{b_i\}$.
3. Transform all b_i's with the estimated transformation to align the scans.

The existing techniques again differ in the means of finding the corresponding points, in the choice of the objective function and the optimization techniques for computing the estimates. Most commonly the corresponding points are found using the nearest neighbor methods in 3D space, which works well when the displacements between the frames are small. In the optimization stage one often chooses first order approximation of the transform, which can be estimated using linear techniques and updates the correspondences in an iterative manner [21]. Hence repeating these steps will either converge to the target transformation, or fall when the maximum amount of iterations is reached. Alternatively one could use closed form pose estimation [22] which works well providing the set of initial correspondences is accurate.

There are a few variants of ICP, all containing the same basic structure but differing in the objective function. The earliest variant is the so called point-to-point ICP method, named after the simple Euclidian distance objective function.

$$T \leftarrow \underset{T}{\operatorname{argmin}} \sum_{i=0}^{m} \parallel T \cdot b_i - a_i \parallel^2$$

Where $T = (R, t)$ and a_i is a point in cloud $A = \{a_i\}$, and b_i is a point in point cloud $B = \{b_i\}$. Improving upon the point to point method, point to plane ICP was developed by [23]. This method takes advantage of surface normal information of point clouds and is achieved by simply changing the objective function.

$$T \leftarrow \underset{T}{\operatorname{argmin}} \sum_{i=0}^{m} \parallel \eta_i \left(T \cdot b_i - a_i\right) \parallel^2$$

where η_i is the surface normal projection at point a_i which is the nearest neighbor of b_i, thus computing the error between one point and the projection of a point onto it's surface normal. In order for this method to work, it must make the assumption that the point clouds being compared are not just arbitrary points in space, but a collection of data from a geometrically known environment. Which is always the case for range sensor data.

The advantages of any ICP variant is the simplicity of the algorithm, and quick performance when the nearest neighbor calculations are optimized. This is typically accomplished with KD trees. A drawback of the ICP algorithm include the assumption that there is a full overlap between the two point clouds. This almost certainly never true when attempting to register range scans. luckily this problem can be resolved by implementing a max distance variable, which states that correspondences greater than the max distance are not considered. Another drawback the assumption that the the data is that of a known geometric surface, this problem in part is solved by point-to-plane ICP. Yet point-to-plane only takes surface normal information from one scan and knows nothing of the second. The generalized ICP method is able to model a plane-to-plane method, which considers the surface information of both scans. It can also be used to model the previous ICP variants.

3.3.1 Gen-ICP

Generalized ICP developed by [24], replaces the error metric of point-to-point and point-to-plane with a probabilistic model. This approach can not only model both previous algorithms but also a plane-to-plane approach as well. It achieves this by assuming that both point clouds have uncertainty governed by an underlying Gaussian distribution. A set of points $\hat{A} = \{a_i\}$ and $\hat{B} = \{b_i\}$ generate $\hat{A}$ and $\hat{B}$ given the following model.

$$a_i \sim \mathcal{N}(\hat{a_i}, {C_i}^A) \text{ and } b_i \sim \mathcal{N}(\hat{b_i}, {C_i}^B)$$

with ${C_i}^A$ and ${C_i}^B$ being covariance matrices associated with measured points in each respective cloud. Assuming perfect correspondences, one can assume that a correct transformation T^* will yield $\hat{b}_i = \mathbf{T}^* \hat{a}_i$. Given an arbitrary transformation $\mathbf{T}$, let ${d_i}^{(\mathbf{T})} = b_i - \mathbf{T} \cdot a_i$. The distribution from which ${d_i}^{(\mathbf{T}^*)}$ is draw, can then be defined as:

$$d_i^{(\mathbf{T}^*)} = \mathcal{N}(0, {C_i}^B + (\mathbf{T}^*){C_i}^A(\mathbf{T}^*)^T)$$

Gen-ICP then uses Maximum likelihood estimation (MLE) to iteratively compute a transformation T, yielding the following minimization problem.

$$\mathbf{T} = \underset{\mathbf{T}}{\operatorname{argmin}} \sum_{i=0}^{n} {d_i}^{(\mathbf{T})^T} ({C_i}^B + (\mathbf{T}^*){C_i}^A(\mathbf{T}^*)^T)^{-1} {d_i}^{(\mathbf{T})}$$

This formula replaces the Euclidian error metric of the previous ICP algorithms. Different values of ${C_i}^A$ and ${C_i}^B$ will transform the algorithm from point-to-point, point-to-plane, and plane-to-plane.

To model point-to-point ICP one may set ${C_i}^B = I$, and ${C_i}^A = 0$. This gives the formula $T = \underset{T}{\operatorname{argmin}} \sum_{i=0}^{n} {d_i^{(\mathbf{T})}}^T {d_i}^{(\mathbf{T})}$, which is exactly the sum of the errors of the Euclidian distance between the points. Similarly, point-to-plane ICP can be modeled by setting ${C_i}^B = {P_i}^{-1}$ and ${C_i}^A = 0$, with P_i being the orthogonal projection matrix, projecting onto the span of the surface normal at b_i. This yields the error metric $\mathbf{T} = \underset{\mathbf{T}}{\operatorname{argmin}} \sum_{i=0}^{n} {d_i^{(\mathbf{T})}}^T {P_i}^{-1} {d_i}^{(\mathbf{T})}$. This is exactly the error between a point in A, and the projected point in B.

While point-to-plane ICP only considers the surface normals of one point cloud, plane-to-plane looks at both, assuming that the scans are made up of real world surfaces, and not just random points in the environment. To take advantage of this, points have high covariance along their surface planes and low covariance along their surface normals. This effect is achieved by setting the covariances to:

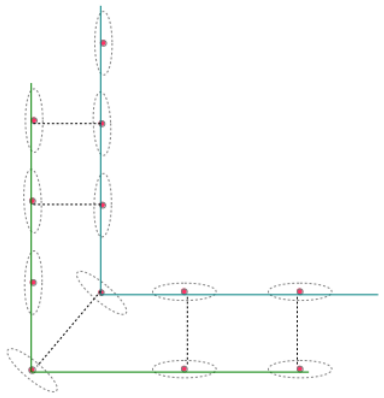

Figure 5: Plane To Plane

$$C = \begin{pmatrix} \epsilon & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Where ϵ is a very small constant representing the covariance along the surface normal. The Covariances of each model are then calculated by:

$$C_i{}^A = R_{\mu_i} \cdot C \cdot R_{\mu_i}{}^T \text{ and } C_i{}^B = R_{\nu i} \cdot C \cdot R_{\nu i}{}^T$$

Where μ and ν are the normal vectors of points a_i and b_i respectively, and R_μ being the rotation that transforms $\mu_i \leftarrow a_i$. This gives plane-to-plane a distinct advantage over point-to-plane, since both surface are taken into account.

Generalized ICP

1: $T \leftarrow T_0$
2: $\{a_i \in A, b_i \in B\}$
3: $C_i{}^A \leftarrow$ {Covariance matrix associated with measured points in cloud A}
4: $C_i{}^B \leftarrow$ {Covariance matrix associated with measured points in cloud B}
5: **for** $i = 0 \rightarrow MaxIterations$ **do**
6: $\quad \mathbf{T} \leftarrow \underset{T}{\text{argmin}} \sum_{i=0}^{m} d_i^{\mathbf{T}^T} (C_i{}^B + \mathbf{T} \cdot C_i{}^A \cdot \mathbf{T}^T)^{-1} \cdot d_i{}^T$
7: **end for**

Figure 6: Algorithm: Generalized ICP

ICP algorithms are often used to reconstruct 2D and 3D models and localize robots by computing motion between scans. There are however shortcomings depending on the sensor modality. Range finders typically have trouble with data association. When displacements in the scans are large, computing nearest neighbors can be difficult. If the displacement is too large, ICP may not converge. Vision based cameras can perform data association easily with visual features. There are times however when visual features are sparse. If data association does not provide a sufficient amount of correspondences, ICP may not converge. The other downside to motion estimation with cameras, is that they are just not as accurate as range finders.

The RGB-D sensor has a distinct advantage over range finders and cameras when estimating motion with ICP. In areas where range finders have difficulty associating data, when displacements in the scans are too large, the RGB-D camera can rely on visual feature matching. In feature limited environments, where vision based systems would fail, RGB-D sensors can rely on pure depth data. The later example can be seen in our GMU hallway experiments.

3.4 RGB-D ICP

The RGB-D ICP method, developed by [18] is one of the first algorithms created for RGB-D mapping. RGB-D images are first matched using SIFT features and a motion between them is estimated with RANSAC. The RANSAC estimate is then used to initialize point-to-plane ICP. The transformations from both RANSAC and point-to-plane ICP are both considered in the objective function where a user parameter (α) gives weight to two sets of correspondences. This allows one to bias the sensor towards a more visual or depth based estimation. The Algorithm is coined RGBD-ICP and can be seen in Figure 3.4.

RGBD-ICP[A,B]

1: $F_A \leftarrow ExtractFeatures(A)$
2: $F_B \leftarrow ExtractFeatures(B)$
3: $[\mathbf{t}^*, A_f] \leftarrow RANSAC(F_A, F_B)$
4: **while** $iterations \leq maxIterations \vee Converged == true$ **do**
5: $\quad A_d \leftarrow ComputeClosestPoints(A, B, \mathbf{t}^*)$
6: $\quad \mathbf{t}^* \leftarrow \underset{t}{\text{argmin}}\, \alpha(\frac{1}{|A_f|} \sum_{i \in A_f} w_i \|\mathbf{t} \cdot b_i - a_i\|^2 + (1 - \alpha)(\frac{1}{|A_d|} \sum_{j \in A_d} w_j \|\eta_j(\mathbf{t} \cdot b_j - a_j)\|^2$
7: **end while**

Figure 7: Algorithm: RGBD-ICP

When α is set to zero, only the ICP features are considered and when α is set to one only RANSAC transformations are taken into account. As transformations are calculated, loop closure and global optimization is also being performed to keep an accurate map.

3.5 RGB-D SLAM

The RGB-D SLAM method is very similar to [18], it can be summarized in the following steps:

1. Input: RGB-D Images;
2. feature extraction and matching (SURF);
3. motion estimation (RANSAC);
4. motion refinement (ICP);
5. motion optimization (HOGMAN);

The first step is simply the acquisition of RGB-D frames from the sensor. Features are extracted using the SURF method instead of SIFT. An initial motion is then estimated and outliers are removed using the RANSAC method and the estimated motion is then refined by generalized ICP. The final step is global optimization performed by the HOGMAN method. In contrast to [18], features are extracted with the SURF method instead of SIFT, and motion is refined using generalized ICP instead of point-to-plane ICP.

A downside of both [17] and [18] is the mandatory use of RANSAC. There are situations where the initial motion estimated by RANSAC is wrong to a degree where initializing ICP with this motion will cause the algorithm to not converge. In this situation it is far better to ignore the RANSAC result and start generalized ICP from scratch. This can be performed using the user given weights in [18], but there must be a method to switch between RANSAC and ICP in order to successfully traverse dynamic environments. The main contribution of our paper is the use of generalized plane-to-plane ICP, as well as a novel approach to using both RANSAC and GICP together to solve for these situations that are difficult for RGB-D sensors to overcome.

4 Dynamic RGB-D Mapping

In this section our approach to motion estimation between two views is described. Our method combines matching and motion estimation strategies found in both visual and range sensing modalities. The general outline of a common RGB-D mapping technique consists of the following steps:

1. Feature detection and matching;
2. Initial rigid body motion estimation with RANSAC;
3. Estimation refinement with ICP registration;

This method however is not robust in situations when RANSAC fails. This can happen for two reasons: there are not enough visual features in the environment, or there are not enough common features between the two frames. The former case can be seen in common indoor environments such as office hallways. The latter case can be attributed to rapid movement, small overlap between consecutive views or possible data loss due to communication and threading issues. An example of this can be seen in Figure 8, the only overlap between the two frames is the corner wall which is essentially featureless. Range sensor scans however would be able to easily associate the data points between the two frames. In all of these scenarios the motion estimate derived from RANSAC may not be representative of the true motion and initializing ICP with this motion can cause ICP to not converge. We propose a method to overcome these weaknesses by using generalized ICP not as a motion refinement tool but as a fallback method if RANSAC provides an unsatisfactory result.

Figure 8: Limited matching features.

4.1 Dynamic RGB-D Mapping

To overcome the problems of a feature limited environment, we have developed an algorithm that is capable of dynamically alternating between motion estimation techniques. A general flow of our algorithm is detailed in Figure 4.1.

There are three possible paths in our algorithm. The first two start with feature extraction and matching with SIFT features, motion estimation and outlier rejection based off of the visual correspondences is then performed by RANSAC. After this step the performance of RANSAC is analyzed, based off of the error returned from the estimation, as well as the amount of visual correspondences that were found to be inliers. If the result is satisfactory then the motion estimate is accepted. In the case of the estimate's error being high generalized ICP is used to refine the estimation. The third case estimates a motion solely from generalized ICP and a random subset of the RGB-D depth data. This is triggered when any of the following happens:

1. **There are not enough visual correspondences to compute an estimate with RANSAC.**
2. **The motion estimate generated by RANSAC has errors high enough to hinder ICP refinement.**
3. **RANSAC failed to find a motion estimate.**

In any of these cases a motion estimate generated by RANSAC will only serve to hinder ICP from converging to the correct solution, therefore it is far better for generalized ICP to search for a motion estimate without the initialization of RANSAC. This path also throws away the correspondences from the feature detection and matching stage in place of a random subset of the

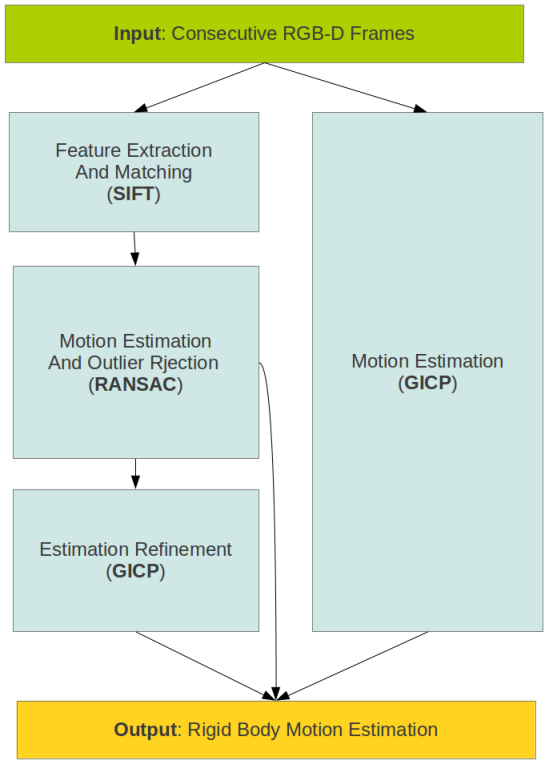

Figure 9: plane-to-plane algorithm flow

entire 3D range data. Generalized ICP does not require correspondences to converge to a correct motion estimation. This is in part due to the max distance value, a variable which ignores corresponding points that are farther away in distance than the max value. This allows for correct registration, even when there is a lack of full overlap of point clouds. Full details of the algorithm are in 4.1.

The algorithm begins by extracting SIFT features from two consecutive RGB-D frames. The extracted features are then used generate a set of correspondences between the two images. With the help of the per pixel depth information, the matched point clouds are converted from 2D image coordinates to 3D world coordinates. An estimate of the motion between the frames, as well as correspondence outlier rejection is then obtained via the RANSAC algorithm. It is here at line six that the code branches into three separate paths. Given the user provided thresholds τ_1, τ_2, and μ The decision of whether to run generalized ICP is based off of the error and number of inliers returned from the RANSAC motion estimation step. If the error of the estimated motion generated from the RANSAC algorithm is less then or equal τ_1 and the number of inliers is less then or equal to μ, then the RANSAC estimate is deemed successful and is used as the final answer, otherwise generalized ICP is

Dynamic RGB-D mapping$[RGBD_A, RGBD_B]$

1: $[\{F_A\}, \{F_B\}] \leftarrow ExtractSIFT(RGBD_A, RGBD_B)$
2: $[2DP_A, 2DP_B] \leftarrow MatchFeatures(F_A, F_B)$
3: $[3DP_A, 3DP_B] \leftarrow ConvertTo3DWorld(2DP_A, 2DP_B)$
4: $T_0 \leftarrow I$
5: $[T, inliers, err] \leftarrow RANSAC(3DP_A, 3DP_B, T_0)$
6: **if** $(sizeof(inliers) \leq \mu) \vee (err \geq \tau_1)$ **then**
7: **if** $err \geq \tau_2$ **then**
8: $[3DP_A, 3DP_B] \leftarrow GetCloudSubset(3DP_A, 3DP_B)$
9: $T \leftarrow GICP(3DP_A, 3DP_B, I)$
10: **else**
11: $T \leftarrow GICP(3DP_A, 3DP_B, T)$
12: **end if**
13: **end if**

Figure 10: Algorithm: Dynamic RGB-D mapping

used. If the RANSAC error is greater then the user provided threshold τ_2, than the motion estimate provided by RANSAC is deemed to0 unstable and generalized ICP starts from scratch. In line 8 a randomized subset of the full 3D range data provided from the RGB-D depth frames is extracted. In line 9 generalized ICP registers the new point clouds using only the identity matrix as an initialization. If the error is less than τ_2 but greater than τ_1, generalized ICP uses the motion estimate and inliers obtained by RANSAC to enhance the transformation **T**.

5 Experiments

In order to test the validity of our algorithm, we ran two separate experiments. The goal of the first experiment was to compare our results to similar RGB-D mapping algorithms of the robotics community, specifically [18] and [17]. This was accomplished by utilizing the public RGBD mapping benchmarks provided by Technische Universität München. While the benchmark datasets provided sequences of various lengths and purposes, all of them contain frames rich with visual features. So the second experiment was set up in order to achieve the goal of testing our algorithm in a feature limited environment. This was accomplished by collecting our own data to experiment with.

5.1 RGB-D Dataset and Benchmarks

The RGB-D dataset is provided by Sturm's group [25], with the intent to provide the computer vision and robotics community with a set of benchmarks to evaluate RGB-D SLAM systems and 3D object reconstruction.

Figure 11: **XYZ:** This benchmark is the smallest benchmark in the group. It is composed of simple translatory motion along the principle axes with little to no rotations. The general purpose of this sequence is for debugging camera calibration issues.

The RGB-D data was taken with a Microsoft Kinect sensor, providing 640 x 480 RGB and depth frames at a 30Hz rate. They have also provided corresponding ground truth trajectories for evaluation purposes. The ground truth motion data was taken with a highly accurate motion capture system, composed of eight 100Hz cameras.

For our experiments we chose five of the benchmarks from the dataset to focus our study on. The chosen benchmarks vary in length, difficulty, presence of loops, and even purpose. We chose these five specific benchmarks because they span a range of difficulty. For example the XYZ benchmark shown in figure 11 is rich with visual features, is of short duration and has contains an extraordinarily large amount of loops allowing for almost constant global optimization. This benchmark is almost trivial to solve and serves mainly as a tool to debug camera calibration errors. In contrast the SLAM3 benchmark, shown in Figure 15 is nearly three times the length, and contains no loops. All benchmarks are described in Figures 11 to 15 and are ordered by their difficulty.

The main goal of these experiments was to provide an empirical comparison of our algorithm vs current state of the art RGBD mapping methods. This was achieved by comparing the benchmark results of our Dynamic RGB-D mapping method (described in Chapter 4) to the RGBD-SLAM [17] algorithm (described in section 3.5). We also compare results against our pure RANSAC method (section 3.2) in order to validate that generalized ICP does improve motion estimation at times.

Figure 12: **Floor:** With a simple sweep over a wooden floor, this benchmark contains a vast amount of easy to track visual features making it an Ideal environment for RANSAC and visual based tracking systems. The floor benchmarks is almost entirely on a single planar surface.

Figure 13: **Room:** This benchmark is a trajectory along an entire office room. The trajectory ends at the exact place it starts from, thus closing a single loop. This benchmark is well suited to debug drift errors and global optimization corrections.

Name	Duration (sec)	trajectory Length (m)
XYZ	30.09	7.112
Floor	49.87	12.569
Room	48.90	15.989
Pioneer	155.72	21.73
Slam3	111.9	18.135

Table 1: rgbd benchmarks

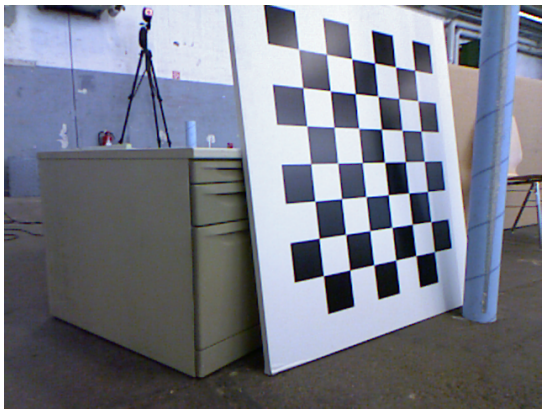

Figure 14: **Pioneer:** At 155.72 seconds in length, the pioneer trajectory is by far the longest tested in these experiments, as well as the first benchmark to have data collected from a pioneer robot. The pioneer was joysticked through a maze of tables, containers, and other walls. There are many loops and opportunities for loop closure in this benchmark, well suited to debug full SLAM systems.

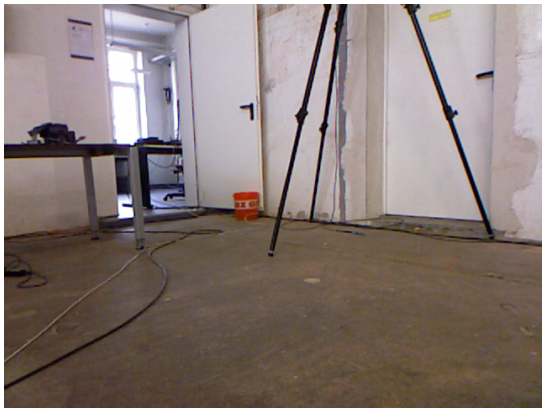

Figure 15: **SLAM3:** The Slam3 is similar to the pioneer trajectory both in length and means of data capture. A pioneer robot was joysticked through a large hall. The trajectory contains no loops, proving to be a difficult data set for algorithms that rely heavily on global optimization.

5.2 Pioneer Hallway Data

In order to test RGB-D mapping in feature limited environments, we took data of the hallway of our University's engineering building. This data consists mainly of white walls and grey carpets, with only doorways and light fixtures providing reliable features. There are sections of the trajectory that involve tight corners with large rotations between frames. This causes some frames to have little to no features in common. These benchmarks causes most algorithms to fail, and must not rely on visual features to compute a transformation. This data was taken with a Microsoft Kinect attached to a Pioneer robot.

5.2.1 Adjustable Parameters

Both the RANSAC and generalized ICP algorithms have a variety of adjustable parameters that affect their performance. The values of these parameters can greatly affect the performance of the mapping system. RANSAC parameters are detailed in Table 3 and the GICP parameters are detailed in Table 4.

6 Results

Presented in this section are the results from both the RGB-D benchmark experiments and the GMU hallway sequence experiment. While both experiments are identical in terms of how the algorithms are run and the format and nature of data collected, the results from both are presented in very different formats. The reason being is that the TUM benchmarks provide highly accurate ground truth data collected from a third party sensor. Due to time constraints we were unable to collected similar data for our hallway collection. As a result benchmark data is presented as relative errors with respect to truth data, while the hallway data is merely comparison of end result plots between algorithms.

6.1 Benchmark Results

Results of the benchmark experiments were calculated using the CVPR provided evaluation tools. Means for evaluation coupled with datasets allows for algorithms developed by the community to be tested under a common process, ensuring accurate comparison results. There are two means of evaluation: absolute trajectory error and relative pose error.

The absolute trajectory error (ATE) evaluation method directly compares the difference between poses in the ground truth and measured trajectory. Measuring the absolute position between poses, this error evaluation is especially useful for evaluating the performance of visual SLAM systems. The end result of the ATE evaluation method is the root mean squared error (rmse) of the

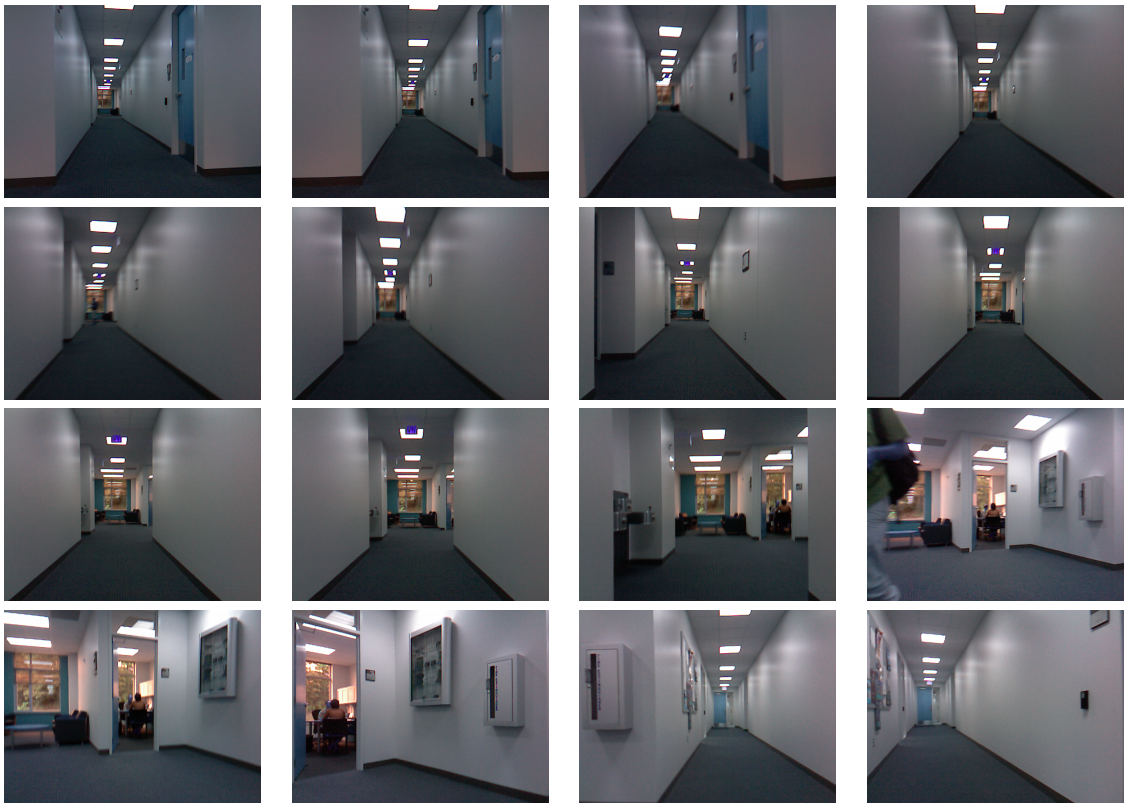

Table 2: GMU Hallway Sequence

Name	**Description**
sample size	The amount of samples used per iteration to compute a transformation
max iterations	The Maximum amount of iterations.
ε	Error threshold for acceptable inlier
Min. Inlier Size	Minimum amount of inliers for an acceptable transformation

Table 3: RANSAC Parameters

Name	**Description**
max distance	The maximum translation allowed before nearest neighbor rejection.
depth cloud size	The size of the 3D cloud created if generalized ICP creates a new motion estimate.
τ_1	Generalized ICP is used to refine the motion estimate, if the error from RANSAC is greater than this threshold.
τ_2	Generalized ICP is used to estimate a new motion with depth data independent of visual correspondences, If the error from RANSAC is greater than this threshold.
μ	The minimum amount of inliers before additional points are added.

Table 4: GICP Parameters

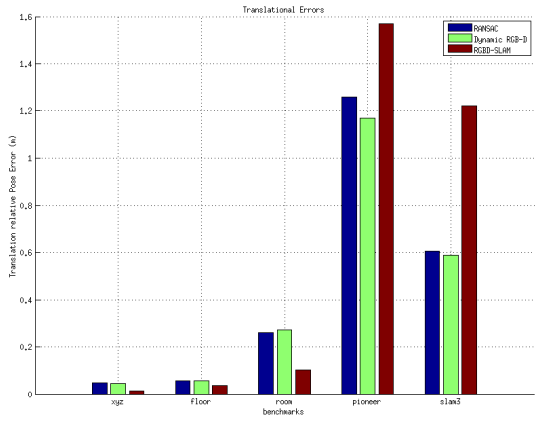

Figure 16: Benchmark evaluation Absolute Trajectory Error

per pose errors summed over the entire trajectory, it is calculated in the following way:

$$ATE_{rmse} = \sqrt{\frac{1}{\eta}(\sum_{i=0}^{\eta} \parallel \hat{t}_i - t_i \parallel^2)}$$
$$\hat{t}_i = \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{bmatrix} t_i = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Where $\hat{t}_i$ is the measured pose at position i, and t_i is the truth pose at position i. Results of the ATE evaluation method on selected benchmarks can be seen in Figure 16.

The relative pose error (RPE) evaluation method directly compares the difference error in relative motion between all timestamps in the trajectory. This method is very useful for evaluating the effect of accumulated drift. For example the evaluation process can only consider timestamps of a fixed delta apart, essentially allowing the user to calculate the drift after a given amount of time. Results from this method are separated into translational errors and rotational errors, this can be seen in Figure 17 which shows the RPE results of a 1.0 second drift on all benchmarks for every algorithm.

Considering that measured data and truth data may lie in different orientations and positions, the evaluation method must first rotate and translate the entire measured trajectory to fit with the truth data. After the two trajectories are aligned, the method simply calculates the motion between the given timestamps. As with the ATE method, results from the REP method will be presented in terms of the root mean squared error over the entire trajectory. The RPE method calculates both translational and rotational errors in the following way:

$$Trans_{RPE} \sqrt{\frac{1}{\eta}(\sum_{i=0}^{\eta} \parallel \hat{t}_i - t_i \parallel^2)}$$
$$Rot_{RPE} = \sqrt{\frac{1}{\eta}(\sum_{i=0}^{\eta} \parallel R_i{}^T \hat{R}_i \parallel^2)}$$

Where $(\hat{t}_i, \hat{R}_i)$ and (t_i, R_i) are the relative translations and rotations between frames of a specified time apart, for measured and truth data respectively.

This section will detail the results of comparing the dynamic RGB-D mapping method against various algorithms for five separate benchmarks. Results have been calculated with both the ATE and RPE methods.

6.1.1 Short Duration Benchmarks

The first set of benchmarks to be tested were the so called short duration benchmarks. These are the 3 benchmarks under fifty seconds long: xyz, floor and room. Overall results are detailed in Figures 16 and 17, for absolute trajectory errors and relative pose errors respectively.

In these short duration benchmarks, both RANSAC alone and the dynamic RGB-D mapping method produced nearly identical results, while Freiburg's RGBD-SLAM algorithm performed slightly better then either. The error gap between RGBD slam and our algorithms could be perhaps attributed to the amount of loops in these bench marks and our lack of loop closure and global optimization in our algorithm.

As stated in Chapter 5, there are a few configurable parameters for both the RANSAC and generalized ICP algorithm, parameter choices for these short duration benchmarks are detailed in Table **??** and **??**. It can be seen that all parameters remain a constant value for every benchmark, with the exception of GICP's error threshold. This is the maximum RANSAC error before generalized ICP used. These values differ between benchmarks, due to RANSAC performing differently between benchmarks. The general strategy for the hybrid mapping method was to use GICP when RANSAC fails, to do this we must set this error threshold at a level above the normal RANSAC errors. A look at RANSAC errors in Figure 18, shows that these error values correspond to a point when only error spikes are above the threshold.

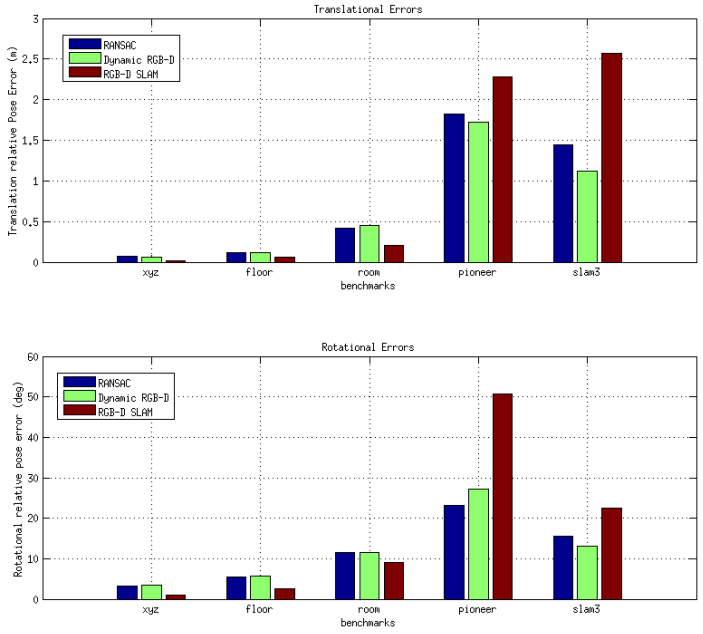

Figure 17: Benchmark Evaluation relative pose error

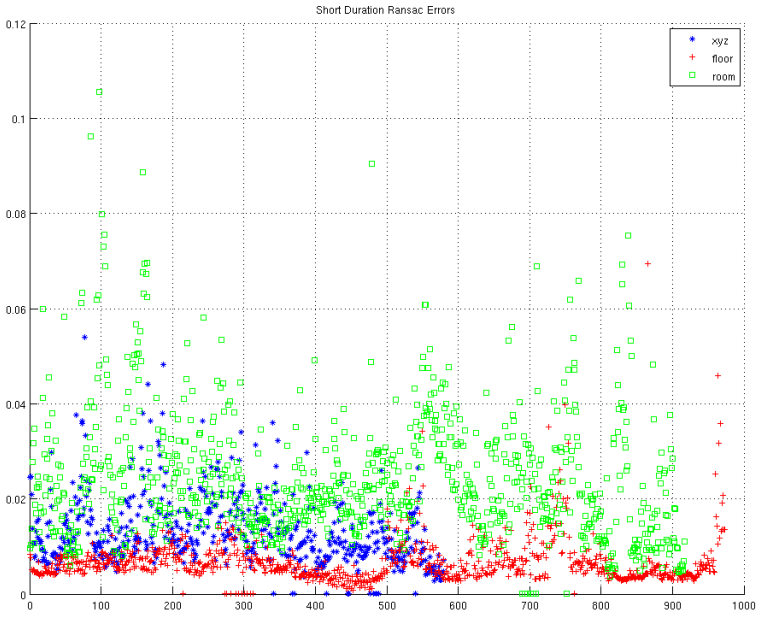

Figure 18: short distance RANSAC errors

XYZ The XYZ benchmark being the most simple and shortest benchmarks expectedly produced the smallest error values for every case. A comparison of realitive pose errors with a 1.0 second drift (Figure 19) shows that RGBD-SLAM consistently outperforms both our RANSAC and dynamic RGB-D method in the first 15 seconds. All algorithms perform nearly identical for the remainder of the benchmark but for one exception: there are occasional error spikes seen in both the RANSAC and Dynamic RGB-D methods. An explanation for the spikes perhaps, is our lack of global optimization.

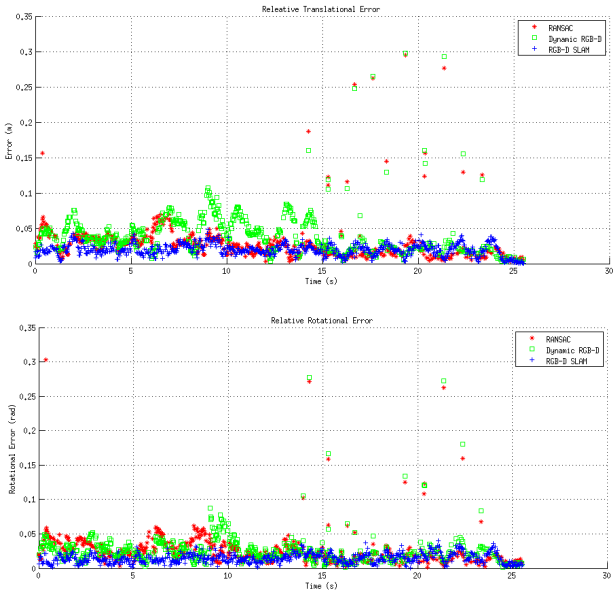

Figure 19: XYZ RPE 1.0 second drift

Along with the error spikes, there is little to no effect on the results when compared with the pure RANSAC algorithm. This can be explained by how well RANSAC performs in this benchmark, it can be seen in Figure 18 that the xyz benchmark has RANSAC errors consistently less than .04 meters.

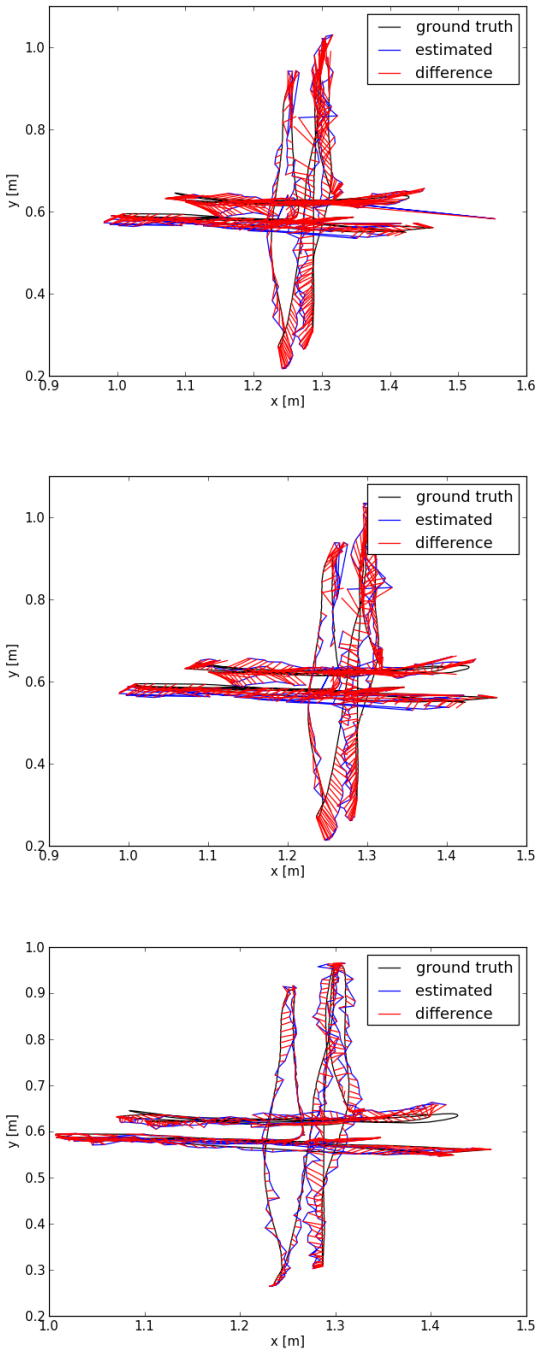

Figure 20: XYZ Absolute Trajectory Plots (RANSAC,Dynamic RGB-D,RGB-D SLAM)

Floor With its environment rich with visual features, computing visual correspondences and generating a motion estimate with RANSAC was expected to be highly successful. Figure 18 shows that the floor benchmark had the lowest overall RANSAC errors. For the majority of the run all of the algorithms performed much the same.

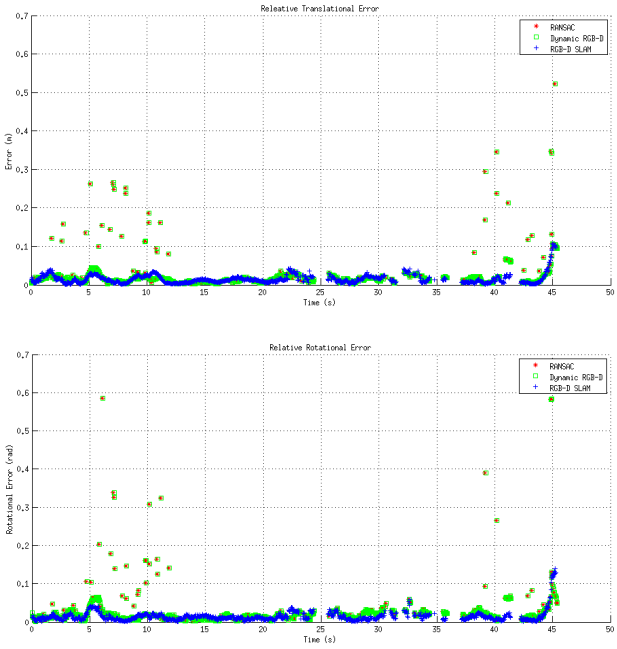

Figure 21: Floor RPE 1.0 second drift

Occasional errors are seen in the beginning and end of the trajectory for both our RANSAC and dynamic RGB-D method as well as a drift in translational error from the RGBD-SLAM implementation in the beginning of the trajectory. These might be caused by our lack of global optimization as there are a number of opportunities for loop closure. Figure 22 displays the absolute trajectory error plots of all three algorithms, it is interesting to note that while the summed errors in Figure 16 are slightly lower for RGB-D SLAM there are points in the trajectory where the dynamic method has smaller errors, especially in the top right corner of the plot.

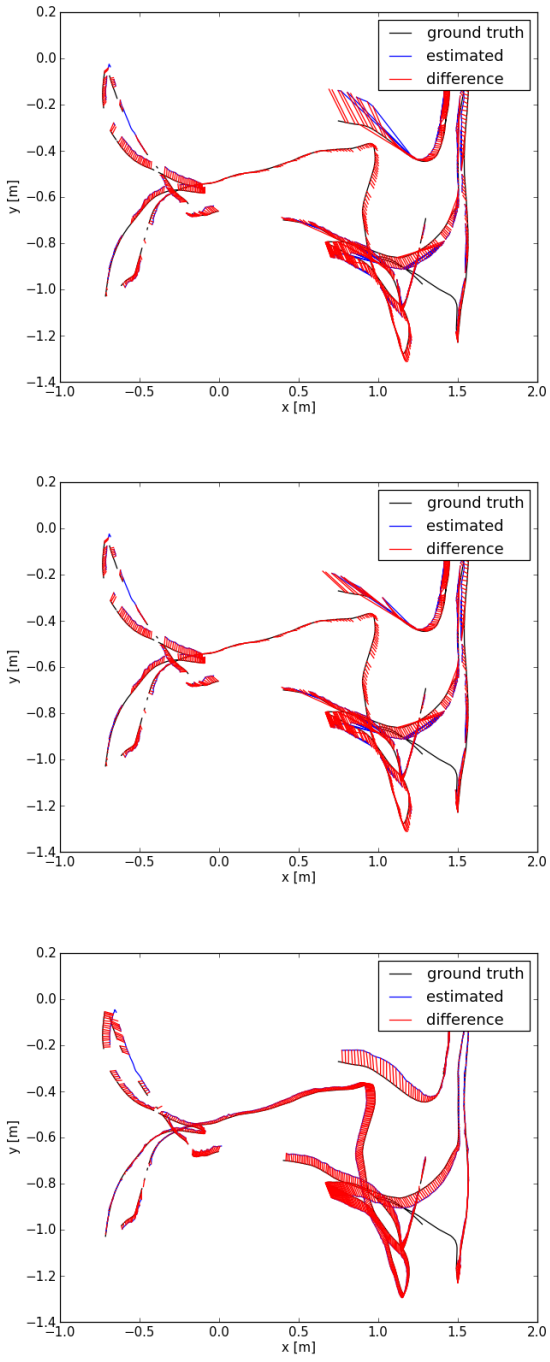

Figure 22: Floor Absolute Trajectory Plots (RANSAC,Dynamic RGB-D,RGB-D SLAM)

Room The room benchmark is a sweep through an office ending at the same point as the beginning, causing the trajectory to be one large loop. Relative pose errors with a 1.0 second drift are posted in Figure 23. All three algorithms have nearly identical rotational errors with the exception of a couple of spikes from each algorithm in different locations.

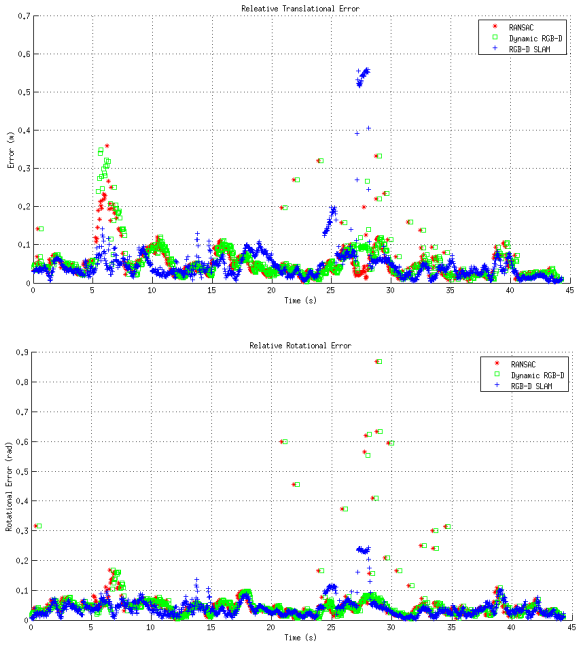

Figure 23: Room RPE 1.0 second drift

Our RANSAC and dynamic RGB-D methods have relatively high translational errors near the beginning of the trajectory, near the 30 second mark however, all algorithms nearly the same translational error. There is one very large error spike near the 25 second mark from RGB-D SLAM that our methods managed to avoid.

As with the other two short duration benchmarks, the dynamic RGB-D method did no better than standalone RANSAC. It is possible that the RANSAC errors are so low, being on average approximately 0.5 centimeters or less, that generalized ICP could not improve upon them.

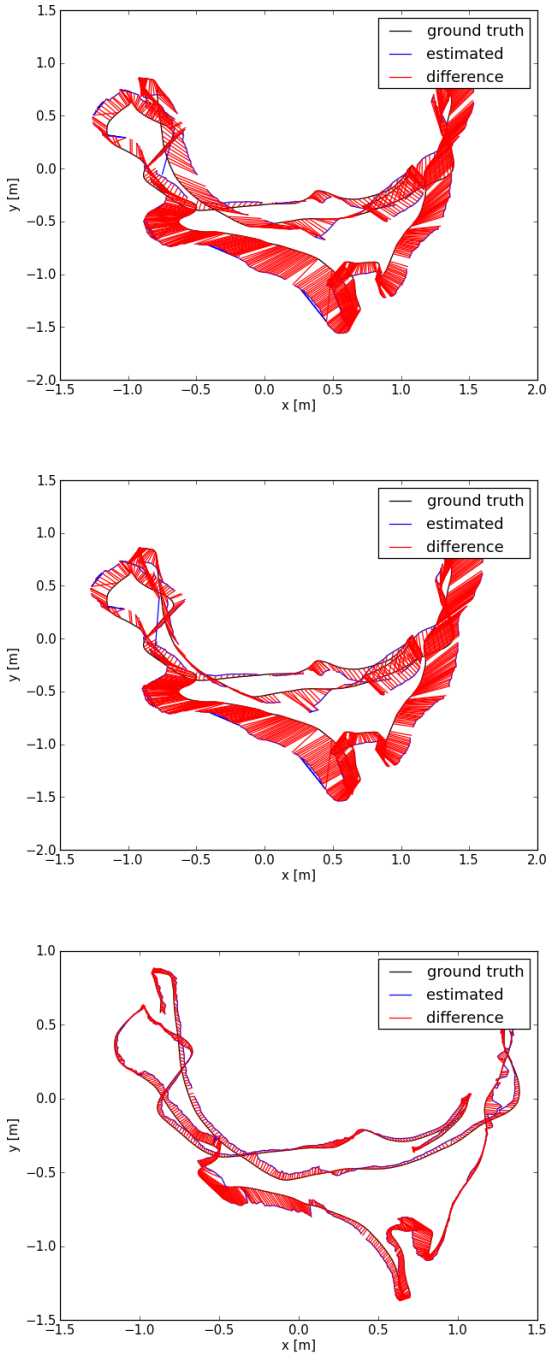

Figure 24: Room Absolute Trajectory Plots (RANSAC,Dynamic RGB-D,RGB-D SLAM)

6.1.2 Medium Distance Benchmarks

In addition to the short duration benchmarks we also tested our algorithms on two longer trajectories, between 100 and 160 seconds in length. Overall results are detailed in Figures 16 and 17, for absolute trajectory errors and relative pose errors respectively. Per frame RANSAC errors for our algorithm can be found in figure **??**. The purpose of testing against these benchmarks was to get an idea of how our algorithms are performing with respect to error due to accumulated drift.

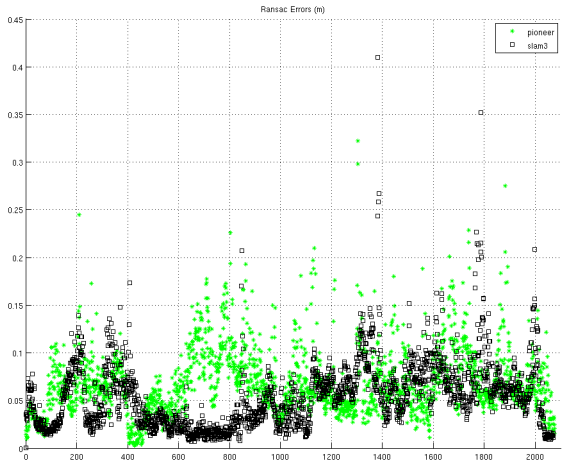

Figure 25: Medium distance RANSAC errors

With the absence of loop closure and global optimization we expected to perform better in short duration benchmarks and worse in longer trajectories, but just the opposite was found to be true. The results of these longer trajectories are strikingly different compared to the short distance benchmarks. In both the pioneer and slam3 trajectories our standalone RANSAC algorithm had lower errors than RGBD-SLAM, and our dynamic RGB-D method had lower errors than our standalone RANSAC algorithm (Figures 16 and 17).

Pioneer The pioneer benchmark was our first long distance test. Relative pose errors with a 1.0 second drift are detailed for every algorithm on this benchmark in Figure 26. Both the RANSAC and dynamic RGB-D mapping algorithms show consistently lower errors throughout the trajectory. There are frequently areas where the dynamic RGB-D method corrected errors made in the RANSAC algorithm, examples of this can be seen in translational error at 50, 120, and 150 seconds.

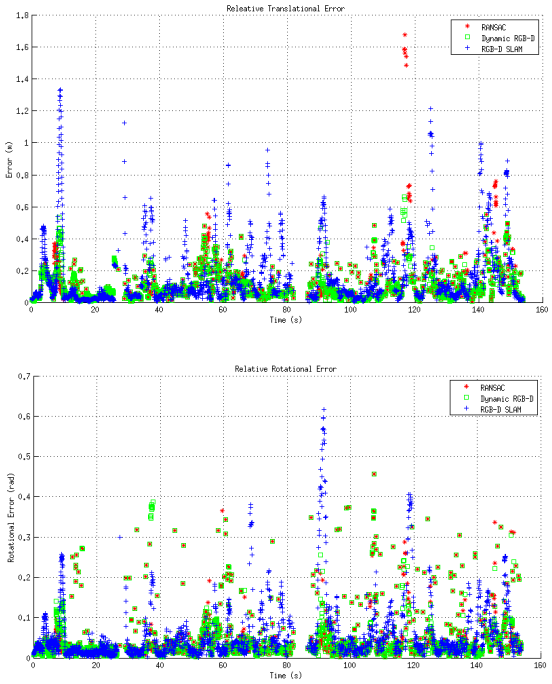

Figure 26: Pioneer RPE 1.0 second drift

These results are very curious and unexpected. The Pioneer SLAM benchmark contains many loops, which should be a great candidate for loop closure / global optimization improvements yet our algorithm outperforms RGBD-SLAM which implements a global optimization method. A possible answer to this might be a possible introduction of false loops into their loop closure algorithms. It is also possible that their algorithm suffered from overuse of generalized ICP and/or the initialization of generalized ICP with an inferior RANSAC estimate caused generalized ICP not to converge. The absolute trajectory plots (Figure 27) affirms that the dynamic RGB-D method's trajectory is the closest to ground truth.

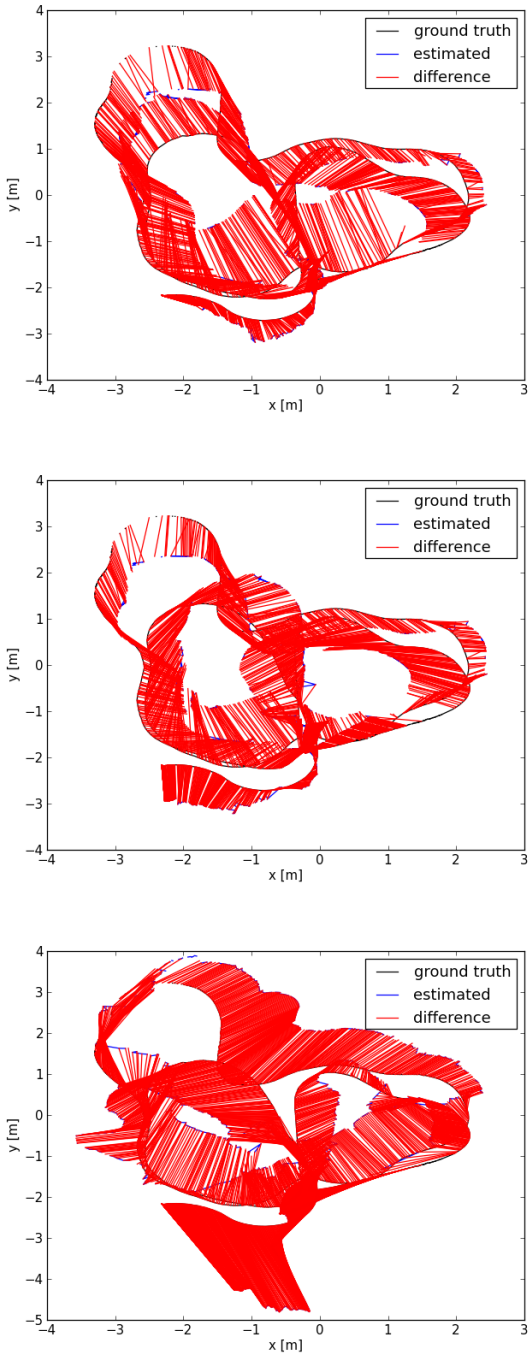

Figure 27: Pioneer Absolute Trajectory Plots (RANSAC,Dynamic RGB-D,RGB-D SLAM)

Slam3 Slam3 results are similar to the pioneer benchmark. Details of the relative pose errors are in Figure 28. It shows that there are times when the RGB-D SLAM's translational error is larger by a full meter. This can be seen near frame 50, frame 65, and frame 98. There are also areas where the dynamic-RGB-D method has lower errors than RANSAC, proving that generalized ICP correcting areas where RANSAC is returning poor estimations.

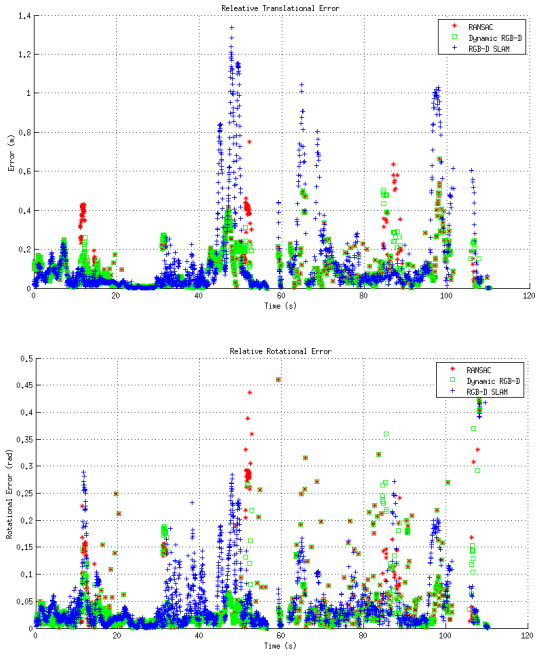

Figure 28: SLAM3 RPE 1.0 second drift

In both medium distance benchmarks, the dynamic RGB-D mapping algorithm produced the smallest relative pose errors and absolute trajectory errors. Our algorithm not only outperforms a global optimization algorithm in a trajectory with many loops, but it also does so in a no loop situation as well. There are two main differences between our dynamic RGB-D mapping method and RGB-D SLAM:

1. RGB-D SLAM contains global pose estimation, dynamic RGB-D mapping does not.

2. RGB-D SLAM utilizes generalized ICP to refine a motion estimate based off of RANSAC, dynamic RGB-D chooses which algorithm to use depending on the circumstances.

These facts point to a few conclusions; either global optimization is hindering the result by false loops or generalized ICP should not be used in all circumstances, and when it is it should not be initialized by a motion estimate with high errors.

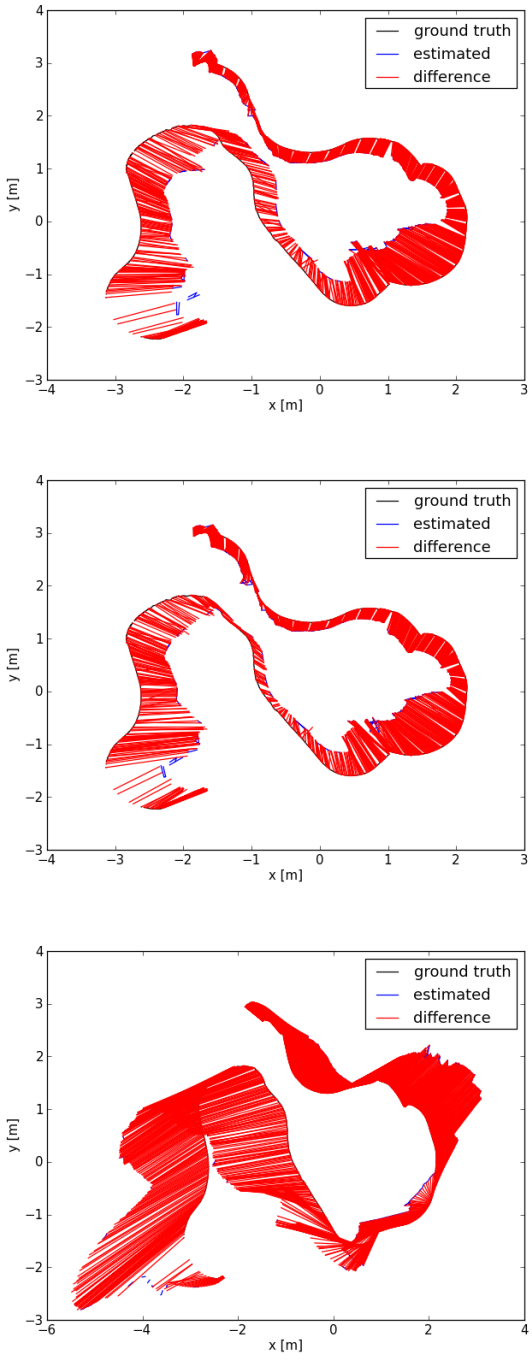

Figure 29: SLAM3 Absolute Trajectory Plots (RANSAC, Dynamic RGB-D, RGB-D SLAM)

6.2 Hallway Results

The intent of the hallway experiments was to see how the dynamic RGB-D method performed in a feature limited environment. This was accomplished by collecting data of some simple trajectories in the hallways of our University's engineering building. A summary of the trajectory is shown in Figure **??**. The sequence is approximately 600 frames and features blank grey walls and solid carpets. it features a straight trajectory down a hallway, one 90° turn to the right and a small distance down the next hallway.

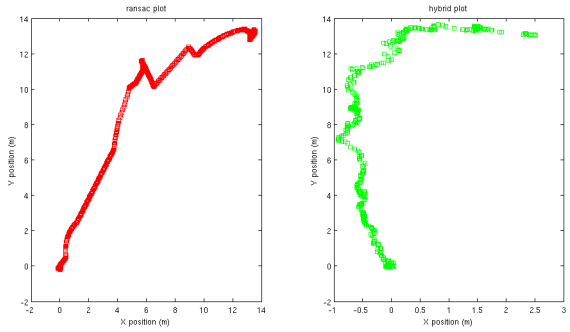

Figure 30: Hallway Comparison

The format of our results for this experiment are quite different from the benchmark data. We did not have the resources or time to collect detailed truth data as was provided with the RGBD benchmarks. Therefore the results provided are comparison plots of the calculated trajectories as can be seen in Figure 6.2. The Figure shows the pure RANSAC algorithm on the left in red compared with the Dynamic RGB-D method on the right in green. Comparing these trajectories to Table 2, it is evident that the pure RANSAC algorithm does not perform very well. The first straight away starting from point (0,0) veers to the right by at least five meters, while the 90° is all but ignored. The Dynaic RGB-D trajectory shows an accurate approximation of the trajectory. While more work needs to be done to quantify the errors between these algorithms, it is self evident that the plane-to-plane method can cope with these environments where the RANSAC only algorithms cannot.

7 Conclusions

The primary goals of our experiments have been met. We were able to benchmark our dynamic RGB-D algorithm against against a current state of the art method with positive results as well as prove that dynamic RGB-D ICP can cope with scenarios that is difficult for RGB-D sensors.

While we only tested against a limited number of benchmarks we can hypothesize on a couple of theories. The first is that in the face of many loops, global optimization can improve local motion estimates. The observations made in the short duration benchmark results (Section 6.1) showed that the RGB-D SLAM algorithm consistently outperformed the dynamic RGB-D mapping in simple trajectories that are rich in visual features and contain many loops. Since very low RANSAC errors in all of the short duration benchmarks leads us to believe that generalized ICP would do little to refine these trajectories, leaving only one real difference between RGB-D SLAM and dynamic RGB-D, global optimization.

The second hypothesis is that as trajectories grow longer in length and become increasingly difficult the need for a method of dynamically alternating between both vision and depth based algorithms increases. This was observed when comparing our dynamic RGB-D method with RGB-D SLAM in our two medium distance benchmark experiments. Despite our algorithm lacking loop closure we were able to drastically outperform RGB-D SLAM in both a trajectory that contained many loops and one that contained none. Disregarding global optimization the main difference between our algorithm and RGB-D SLAM is the role Generalized ICP plays. In the RGB-D SLAM method, generalized ICP acted as a mandatory refinement step, picking up where the RANSAC algorithm leaves off. In the dynamic RGB-D method, generalized ICP and RANSAC are used according to observed errors from the initial RANSAC motion estimate. This would allows to hypothesize that there are certain conditions when the RANSAC motion estimate might be is off to a degree where it will hinder generalized ICP from converging, and may even cause further deviation from the correct transformation. This makes it necessary at times to abandon the RANSAC estimate and have generalized ICP start from scratch. Our RGB-D mapping algorithm does this and has proven to outperform a state of the art RGB-D mapping algorithm, even in absence of global optimization.

The benchmark results proved that our dynamic RGB-D mapping method was able to successfully approximate our hallway trajectory dataset, compared to the standalone RANSAC algorithm which did not. The initial results show that dynamic RGB-D mapping has potential to solve the current problems of mapping with RGB-D sensors.

7.1 Future Work

Despite the successes of both the RGB-D benchmark experiments as well as the feature limited hallway, there is still a great deal of work and improvement that can be done. This section acts as a launch pad of ideas, and future goals for the dynamic RGB-D mapping algorithm as well as the RGB-D mapping problem in general.

Global Optimization In Section 6.1 our results showed that the dynamic RGB-D mapping algorithm

could outperform RGBD-SLAM in situations that highly favor algorithms with loop closure and global optimization, as well as situations where loop closure had no effect. This would point to the fact that the addition of loop closure and global optimization to our algorithm could possibly improve are results even more. We present both a loop closure algorithm as well as a global optimization method that could be added to our algorithm.

Loop Closure Given two images it is simple to calculate the likelihood that they are images taken of the same space. The problem is that given a map with thousands of images, it is too time consuming to compare a new image, with each image in the map. Even if you disregard the past X images, the time of loop closure grows with the size of the map. This problem has been solved by [5]. Given a database of 50,000 images, they are able to perform a loop query in 25ms. This is achieved by first building a hierarchical k-means tree, consisting of images taken in the given environment. As new images are taken, their features are extracted and pushed down the tree. Each feature will have a unique path down the tree, this can be thought of as a features identification. An inverted file index can be found at every leaf of the tree. If an image's feature ID ends at the leaf, its image index will be added to the inverted file index. For a given image, it can query the existing database by computing IDs for each feature, and comparing it to the inverted file indices. The hierarchical nature of the descriptor tree makes this query run very quickly.

Graph SLAM Graph SLAM (Simultaneous localization and mapping) is the the solution to the SLAM problem using a graph structure with a probabilistic framework. The SLAM problem is that there will always be errors in a robot's odometry and sensors. There is only so much the scan matching portion of the code can do about it. If the errors build up too much it will cause the map to become inaccurate and fail. The Graph SLAM problem can help to cope with some of the errors by incorporating loop constraints into the global optimization framework.

Further Experiments While the hallway experiments and results proved that using generalized ICP along with RANSAC is a viable method for coping with feature limited environments, More experiments with a better means of evaluating results needs to be performed. We propose collecting data and performing experiments much in the same vein as the TUM RGB-D dataset, with a focus on feature limited environments and difficult trajectories where current RGB-D methods will fail. This will serve as a way to test the robustness of RGB-D mapping algorithms and perhaps assist in the development of new algorithms to overcome these difficult situations. In this thesis we tested our algorithm against five benchmarks comparing to two separate algorithms. In order to fully grasp the performance of our algorithm, we need to evaluate against trajectory in the benchmark dataset.

References

[1] G. Grisetti, R. Kuemmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based slam." in *IEEE Transactions on Intelligent Transportation Systems Magazine.*, vol. 2(4), 2010, pp. 31–43.

[2] K. O. Arras, J. A. Castellanos, and R. Siegwart, "Feature-based multi-hypothesis localization and tracking for mobile robots using geometric constraints," in *Proc. IEEE International Conference on Robotics and Automation (ICRA'02)*, Washington DC, USA, 2002.

[3] K. Konolige and M. Agrawal, "Frame-frame matching for realtime consistent visual mapping," in *Proc. International Conference on Robotics and Automation (ICRA)*, 2007.

[4] D. Gallup, J.-M. Frahm, and M. Pollefeys, "Piecewise planar and non-planar stereo for urban scene reconstruction," in *CVPR 2010 (IEEE Int. Conf. on Comp. Vision and Pattern Recognition)*, 2010.

[5] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, ser. CVPR '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 2161–2168. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2006.264

[6] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, pp. 239–256, February 1992. [Online]. Available: http://dl.acm.org/citation.cfm?id=132013.132022

[7] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *AUTONOMOUS ROBOTS*, vol. 4, pp. 333–349, 1997.

[8] D. H ahnel, W. Burgard, D. Fox, and S. Thrun, "An efficient fastslam algorithm for generating maps of large-scale cyclic environments from raw laser range measurements," in *IN PROC. OF THE IEEE/RSJ INT. CONF. ON INTELLIGENT ROBOTS AND SYSTEMS (IROS*, 2003, pp. 206–211.

[9] R. Siegwart, I. Nourbakhsh, and D. Scaramuzza, *Autonomous Mobile Robotics*, 2nd ed. MIT Press, 2011.

[10] Z. Zhang, "Iterative point matching for registration of free-form curves," 1992.

[11] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004, http://www.cs.ubc.ca/ lowe/keypoints/.

[12] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Comput. Vis. Image Underst.*, vol. 110, pp. 346–359, June 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1370312.1370556

[13] F. Fraundorfer, C. Engels, and D. Nistér, "Topological mapping, localization and navigation using image collections," in *Proc. of IEEE/RSJ IROS*, 2007, pp. 3872–3877.

[14] K. Konolige and J. Bowman, "Towards lifelong visual maps," in *IROS - best paper finalist*, 2009, pp. 1156–1163.

[15] P. Newman, D. Cole, and K. Ho, "Outdoor slam using visual appearance and laser ranging," in *Proc. of IEEE ICRA*, 2006, pp. 1180–1187.

[16] J. Strom, A. Richardson, and E. Olson, "Graph-based segmentation for colored 3D laser point clouds," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2010.

[17] N. Engelhard, F. Endres, J. Hess, J. Sturm, and W. Burgard, "Real-time 3d visual slam with a hand-held rgb-d camera," in *Proc. of the RGB-D Workshop on 3D Perception in Robotics at the European Robotics Forum*, Vasteras, Sweden, April 2011.

[18] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments." Intel Labs Seattle, Seattle, WA: International Symposium on Experimental Robotics, 2010.

[19] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry, *An Invitation to 3D vision: From Images to geometric models.* Springer Verlag, 2000.

[20] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography." Communications of the ACM, June 1981.

[21] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," in *1992 IEEE Intl. Conf. on Robotics and Automation*, 1991, pp. 2724–2729.

[22] B. K. P. Horn, H. Hilden, and S. Negahdaripour, "Closed-form solution of absolute orientation using orthonormal matrices," *JOURNAL OF THE OPTICAL SOCIETY AMERICA*, vol. 5, no. 7, pp. 1127–1135, 1988.

[23] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images." Proc. of the 1992 IEEE Intl. Conf. on Robotics and Automation, 1991.

[24] A. V. Segal, D. Haehnel, and S. Thurn, "Generalized-icp." In Robotics: Science and Systems, 2009.

[25] J. Sturm, S. Magnenat, N. Engelhard, F. Pomerleau, F. Colas, W. Burgard, D. Cremers, and R. Siegwart, "Towards a benchmark for rgb-d slam evaluation," in *Proc. of the RGB-D Workshop on Advanced Reasoning with Depth Cameras at Robotics: Science and Systems Conf. (RSS)*, Los Angeles, USA, June 2011.