4-2013

Fusion of Ranging Data From Robot Teams Operating in Confined Areas

Damian M. Lyons
Fordham University

Karma Shrestha
Fordham University

Tsung-Ming Liu
Fordham University

Recommended Citation

Lyons, Damian M.; Shrestha, Karma; and Liu, Tsung-Ming, "Fusion of Ranging Data From Robot Teams Operating in Confined Areas" (2013). *Faculty Publications*. 12.
https://fordham.bepress.com/frcv_facultypubs/12

Fusion of Ranging Data From Robot Teams Operating in Confined Areas

Damian M. Lyons, Karma Shrestha, and Tsung-Ming Liu

Fordham University, Robotics and Computer Vision Laboratory, Bronx NY 10458

ABSTRACT

We address the problem of fusing laser ranging data from multiple mobile robots that are surveying an area as part of a robot search and rescue or area surveillance mission. We are specifically interested in the case where members of the robot team are working in close proximity to each other. The advantage of this teamwork is that it greatly speeds up the surveying process; the area can be quickly covered even when the robots use a random motion exploration approach. However, the disadvantage of the close proximity is that it is possible, and even likely, that the laser ranging data from one robot include many depth readings caused by another robot. We refer to this as mutual interference.

Using a team of two Pioneer 3-AT robots with tilted SICK LMS-200 laser sensors, we evaluate several techniques for fusing the laser ranging information so as to eliminate the mutual interference. There is an extensive literature on the mapping and localization aspect of this problem. Recent work on mapping has begun to address dynamic or transient objects. Our problem differs from the dynamic map problem in that we look at one kind of transient map feature, other robots, and we know that we wish to completely eliminate the feature.

We present and evaluate three different approaches to the map fusion problem: a robot-centric approach, based on estimating team member locations; a map-centric approach, based on inspecting local regions of the map, and a combination of both approaches. We show results for these approaches for several experiments for a two robot team operating in a confined indoor environment .

Keywords: cognitive robotics, navigation, sensory fusion.

1. INTRODUCTION

Increasingly robots are involved in applications such as responding to explosive threats, search and surveillance in a potential or actual disaster area, or area reconnaissance. Reid and Braunl [11] consider the scenario where a team of robots spread apart to survey a large area. A key issue involved in this kind of mission is the fusion of the sensory information from each of the team members into a single model. In that case of Reid and Braunl, this was accomplished by leveraging Konolige et al.'s [3] multi-robot SLAM approach. While they used between five and seven robots, additional robots would allow a faster construction of the fused map. And the faster the map is produced, the faster it can be communicated to, for example, the first-response or mission-command team. Also, like much of the SLAM work, Reid and Braunl produced a 2-D area map as their final result. While a 2-D map is very valuable, a 3-D model contains much more information about the scene that can be of value to the command team – the location of possible victims or threats on the floor for example. We address the application scenario where many robots are exploring a confined area with the objective of quickly producing a 3-D model for communication to a (possibly remote) command center.

In the situation where a large robot team is searching a confined area – a warehouse or office building, for example – it becomes very likely that the robots will encounter each other repeatedly (e.g., Figure 1). Since the robots produce 3-D scan data, each such scan will add extraneous information to the final, fused model. As a working example, we deploy a team of two Pioneer 3-AT robots in such a fashion that while producing a 3-D scan of an area they are intermittently within view of each other. As an example of the kind of problem the extraneous information due to this mutual interference causes, we look at identifying a floor plane from the joint dataset of both robot scans. We show that when all this extraneous information is manually removed, the error in the fit of the floor plane can be reduced by up to 98%. This is our motivation is look for automatic ways to remove the extraneous data, which for simplicity we will refer to as *ghost data.*

In the next section, we review the current literature on this problem. Removing dynamic objects from map data has been addressed in the literature, but usually indirectly, as a by-product of data fusion. Our problem is more specific than removing dynamic objects from the map: removing scans of robot team members from the data. In Section 3, we overview the data collection architecture we have developed and the two 3-D laser-scanning robot configurations we use in our team. These two configurations employ a statically tilted laser scanner configuration and exploit robot motion to carry out scanning. In Section 4 we describe the dataset used for the evaluations and results described in this paper including the evaluation of our estimated best performance using the floor plane test mentioned earlier. We go on to introduce a set of metrics that can be used to determine the benefits of each of our proposed automatic ghost removal algorithms. Section 5 presents four proposed methods for ghost removal: two map-based methods, really dynamic object removal methods; and two robot-based methods, leveraging the additional robot team member information available. While each method has performance pros and cons, we show that a combined map-based and robot-based method removes 91% of the ghost data and produces a 98% improvement in floor plane fit. Section 6 summarizes and comments on future work.

2. PRIOR WORK

The fusion of sensory information, usually sonar or laser, to produce an accurate 2-D area map has been a subject of much research in robotics: for example, laser scan registration (Lu & Milios [7]), fusion of 2-D maps (Carpin [1]), and multirobot SLAM (Liu and Thrun [6], Konolidge et al [3], Reid and Braunl [11]). Fusion of 3-D maps offers even more challenges. Nuchter [8] exploits parallelism to address the additional computational complexity of 3-D scan registration. Kaushik et al [4] accomplish the 3-D registration of two robots by using external fiducial markings.

However, the effect of dynamic objects in the sensory data used to produce the fused map is usually not directly addressed; instead the fusion method is relied upon to eliminate the transient data produced by dynamic objects. However Wangsiripitak and Murray [12] describe an approach to SLAM (for body mounted cameras) that does include dynamic object removal. Their approach is to link SLAM with an object-tracking module. Moving objects are identified and tracked for long periods. The background 3-D scene can then be reconstructed without these moving objects. While this method is effective, it requires the mechanisms of object tracking.

Litomisky and Bhanu [5] remove moving objects from 3-D point cloud scenes by identifying objects in RGB-D between successive point cloud frames using a Viewpoint Feature Histogram. This method requires the time consuming comparison of successive point clouds; in our application each of the many robots is producing point cloud scans many times a second. Ortega and Andrade-Cetto [9] remove dynamic objects from lidar data by leveraging a video camera that has been registered with the lidar sensor. The moving objects are detected in the video imagery using well-known computer-vision background estimation and subtraction algorithms (see Piccardi [10] for a review). The video imagery is then used to filter the lidar data. This method, and also that of Litomisky and Bhanu, requires image as well as depth data. Our robot team produces 3-D laser scan data with no RGB component. However, we are inspired by the comparison with background estimation in computer vision, and we follow this connection when we propose novel map based approaches to filtering moving object from our datasets in later sections.

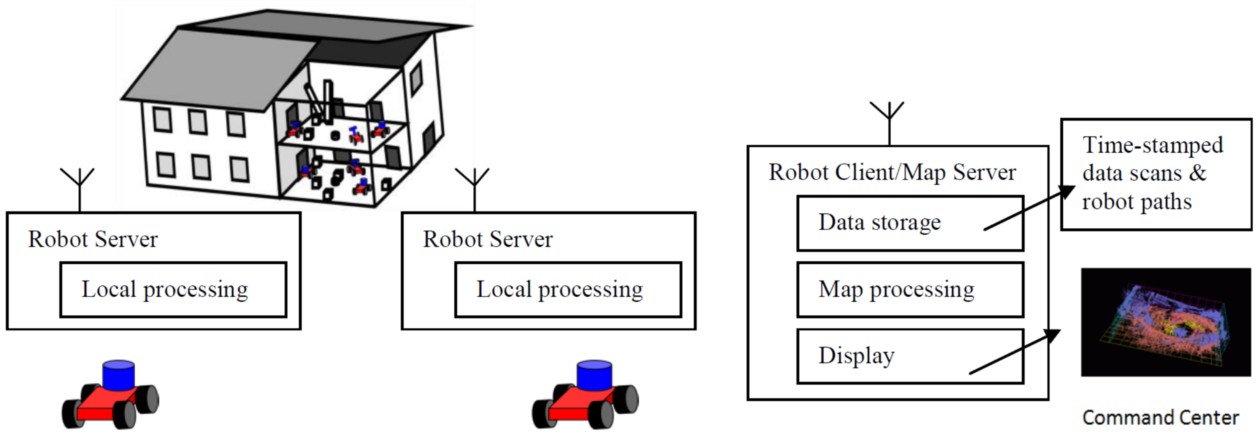

Figure 1: Robot Team Data Collection Architecture with Multi-Robot Building Search Scenario.

3.   DATA COLLECTION ARCHITECTURE

We are specifically interested in surveillance, reconnaissance or search and rescue applications where a team of robots is deployed to a building or other confined area and must operate quickly to assembly a model of the area. We have constructed a straight-forward software architecture to investigate issues that arise in this kind of application. Figure 1 shows a team of robots deployed to surveil/search an industrial building. The cutaway side of the building in Figure 1 shows six robots on two floors and indicates the common situation that multiple team members are operating in close proximity in order to assemble data on the site as quickly as possible. We expect that such teams will be of heterogeneous composition.

Data collected by each team member, as well as the robot pose information for each team member, is transmitted to the central map server. The map server time-stamps and stores all the data that is transmitted – in the Point Cloud Library (PCL) format – for later analysis or inspection. This is how the data sets used in the later sections of this paper originated. The map server uses PCL to display the data collected. All the point cloud imagery shown in this paper were visualized using PCL.

We have integrated three classes of robot team member into our architecture: two laser-based robots with the lasers mounted in different configurations and a stereovision based robot. We will only discuss data from the two laser-based robots in this paper. Both laser-based robots and their laser configurations are shown in Figure 2. A common configuration to mount a laser scanner on a robot is so that the laser scans in a horizontal plane with respect to the robot. This configuration enables the construction of accurate 2-D area maps and many results have been reported using it. However, this configuration limits the scan to a 2-D slice through the robot's environment – any objects above or below that slice are not seen at all. We use a simple modification of this common configuration to allow us to generate 3-D models: that is, a single, tilted laser.

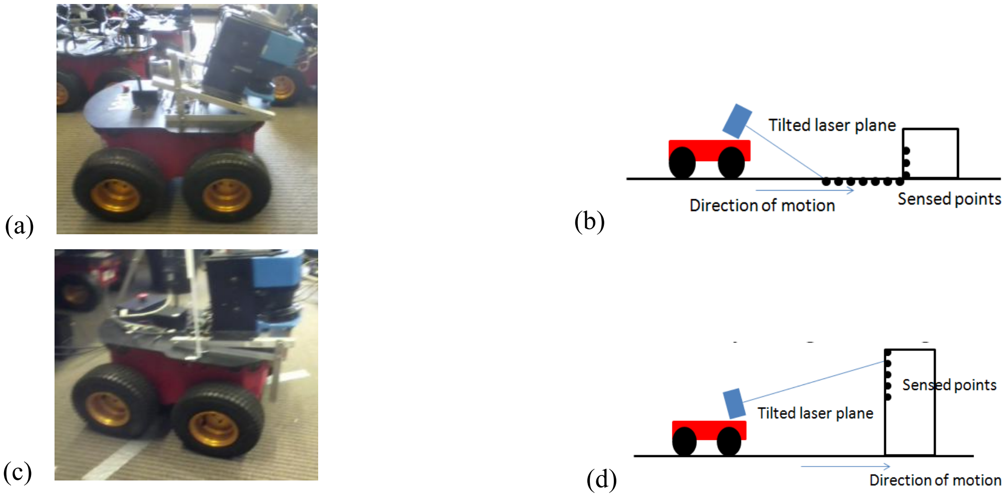

Figure 2: (a) Down laser mounted on Pioneer 3-AT; (b); Scanning action of down-laser;
(c) Up laser mounted on Pioneer 3-AT; (d) Scanning action of up-laser.

Figure 2(a) shows the down-laser robot, a Pioneer 3-AT which has its SICK LMS-200 laser mounted tilted downwards. Figure 2(b) illustrates how when the robot moves, it scans the laser plane over the terrain ahead, including any objects. The limitation is that nothing higher than the center of the laser can be scanned. Figure 2(c) shows the up-laser robot, which has its laser mounted tilted upwards. Figure 2(c) illustrates how when the robot moves forward the laser plane is scanned over parts of the scene above the center of the laser.

These two robots are complementary in that one can map the 3-D structure below the height of the laser scanner, and one maps the structure above the laser scanner. Both operating together produces a full 3-D model. However, if both operate in close proximity, then they may scan each other.

4. DATASETS AND METRICS

The architecture described in the previous section was used to collect several combined datasets from a team of one up-laser and one down-laser robot. Figure 3 shows one of these, the *Lap2* dataset. The oval track shown in Figure 3(a) was tracked by both up-laser and down-laser robot, with one robot being approximately one half oval ahead of the other. Each robot completed two entire laps of the oval track. The resulting data set is displayed as a point cloud in Figure 3(b) from the top-down, Figure 3(c) looking from the side, and Figure 3(d) in a perspective view. The data is color-coded (shaded) as follows: The up-laser data is all shown in a light shade of gray, whereas the down-laser data is all shown in a darker shade of gray. The points at which one robot scanned the second, which we will refer to for convenience as *ghost points*, are shown in black. The actual track (Kalman-filtered by the Map Server) for each robot is shown as superimposed shaded points. The floor and wall planes shown are fitted using RANSAC from the combined dataset.

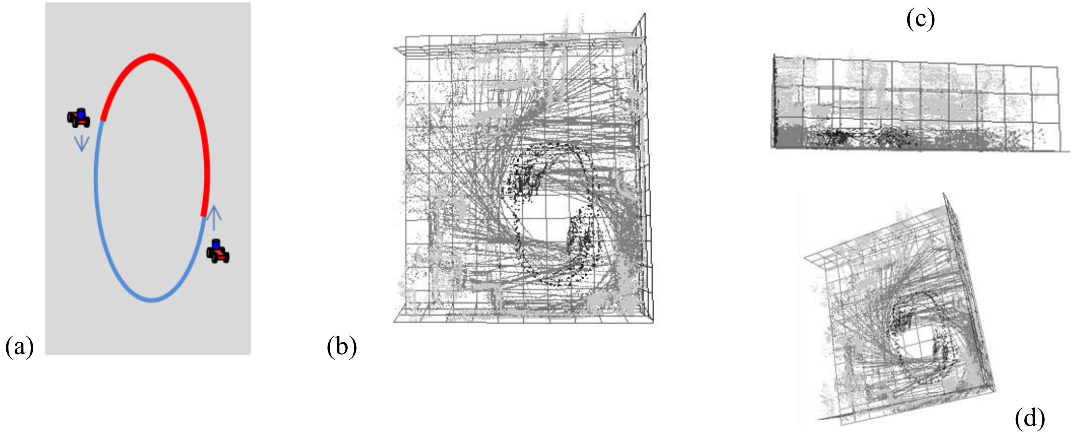

Figure 3: (a) Up-laser (dark) and down-laser (light) oval path (b, c, d); Color-coded data collected (down-laser, dark gray; up-laser, lighter gray; scans showing 'other' robot, black; each robot path is shown by superimposed shaded points. The floor and wall planes shown are estimated from the data.

We focus on the estimation of the floor plane, a crucial step in converting the point cloud to a surface or object representation. However the accuracy with which a floor plane can be determined is strongly influenced by the existence of the ghost data. Table 1 illustrates this influence for two data sets; the Lap2 data set shown here and an additional three lap 'figure of eight' track dataset. The floor plane is fitted by RANSAC and represented in the form $ax+by+cz-d=0$. The average error is calculated by evaluating $\in = ax+by+cz-d$ for all the RANSAC inliers on the floor plane and averaging the result. Table 1 shows the results of this measurement for the two datasets in their original form (first three columns) and with all the ghost data eliminated by hand (second three columns). In each case, the first column shows the average error, the second shows the percentage of all the points that were inliers on the plane and the third column shows the percentage of the inliers that were ghost points (and hence possibly corrupting the plane estimation). The final column shows that eliminating the ghost data would produce a potential improvement in the average error of 98%. This observation motivates our development and evaluation of methods to automatically filter this ghost data.

Table 1: Plane Error for two datasets with, and without, ghost data.

	Including ghost data			Not including ghost data			
Data	Error (mm)	% Inliers	% Ghosts	Error (mm)	% Inliers	% Ghosts	% Improve
Lap	-3.1	82%	75%	0.054	83%	0%	98%
Fig8	-1.5	82%	55%	-0.046	83%	0%	97%

For each method for automatic removal of ghost data that we develop in the next section, we will measure effectiveness of the method with four metrics as follows:

1. % Ghost Points: This is the percentage of the ghost data that was *identified* by the method. The larger this number, the more ghost points were identified and so the better.
2. % Other Points: Is the percentage of non-ghost data that was identified by the method. The smaller this number the fewer non-ghost points were mistakenly identified, and so the better.
3. % Improvement in Plane Error: This is the percentage improvement in the average error for the floor plane after filtering. The bigger this number, the smaller the error, and so the better.
4. % Improvement in Plane Inliers: This is the percentage improvement of the percentage of RANSAC inliers for the plane. The bigger this number, the more points are considered inliers for the plane, and so the better.

5. METHODS AND EVALUATIONS

In this section, we propose and evaluate several approaches to the filtering of the transient data, ghost data, due to robot team members being scanned by other team members because of the close proximity of operations. We will investigate two map-based approaches and two robot-based approaches. The map-based approaches have the advantage of not having to use the position of the robot team members and can thus be robust to errors in that estimation. However, the full map needs to be filtered. The robot-based methods include some knowledge about the relative location of robot team members. The filtering carried out by these approaches can be done on a robot by robot, scan by scan basis, and the entire map never needs to be known or used.

5.1 Map-Based Methods

Our inspiration for map-based methods for ghost removal is the computer vision technique of background subtraction (Piccardi [10]): background subtraction estimates the appearance of a visual scene without the (typically dynamic) foreground regions and then eliminates this background on each image with the objective of identifying the foreground objects. One background estimation approach is the median filter (Gutchess et al. [2]): statistics are generated for each pixel in the image and the median calculated as an approximation to the 'true' background pixel. Eliminating the transient, ghost data from our map is tantamount to calculating a 'background' map analogous to the background image. However, unlike the computer vision analogy, we do not have data on every voxel in the map, only on some. Also, we can only measure whether or not we scanned a point at that voxel; there is nothing analogous the 'color' of a pixel in background estimation. Nonetheless, we can calculate frequency of occurrence for each voxel and use that to estimate the background scene on the basis that the transient, ghost data will occur less often, since the robots are moving around, than the static objects that comprise the background.

To calculate the map spatial statistics, space is represented as a voxel array *V(x,y,z)*. The resolution *r* of the array is the dimension of a single voxel. In our examples, the entire space is a cube of *10×10×10 meters*. We conduct experiments for two resolutions, *r=0.1m* and *r=0.05m*. From a computational perspective, in the worst case *V* requires $(10/r)^3$ voxels. However, *V* is *extremely* sparse. For the largest data set we collected, only *0.002%* of voxels are occupied and need to be represented. Spaces in which robots and people move can typically be approximated as large hollow cubes. A hollow cube of side *n* would have an occupancy of $6n^2/n^3$ or *6/n*. With this linear space complexity, the voxel representation is not a space intensive one and is efficient for filtering even large spaces as long as only the occupied data is represented.

Method 1: Map-based frequency filtering.

Since robots will move in and out of each other's field of view, they will occupy specific locations in space for relatively short periods and hence give rise to occupied voxels that are observed less frequently. Let *S(r,t)* be the set of points in the laser scan of robot *r* at time *t*. For each *p* ∈ *S(r,t), p = (x,y,z)* in global map coordinates. The full point cloud that comprises the map is:

$$S = \bigcup_{r\epsilon\{R1,R2\}} \bigcup_{t\epsilon\{t0\cdots tn\}} S(r,t) \tag{1}$$

The voxel array is populated by the frequency of occurrence of point locations in the cloud:

$$V(b) = \sum_{p\epsilon S} \delta_{pb} \qquad \delta_{pb} = \begin{cases} 1 & b = \left(\frac{x}{r},\frac{y}{r},\frac{z}{r}\right) \\ 0 & else \end{cases} \tag{2}$$

Where the division operation here is integer division. The mean and variance of these frequencies are calculated:

$$\mu_f = \frac{1}{|B|}\sum_{b\epsilon B} V(b) \quad \sigma_f^2 = \frac{1}{|B|}\sum_{b\epsilon B} (V(b) - \mu_f)^2 \qquad B = \{\, b = \left(\frac{x}{r},\frac{y}{r},\frac{z}{r}\right) : \ p\epsilon S \wedge \delta_{pb} = 1\} \tag{3}$$

We can now define the map-based frequency filtering algorithm by the rule:

$$Filter(p)\ if\!: \ b = \left(\frac{x}{r},\frac{y}{r},\frac{z}{r}\right)\ and\ V(b) < \mu_f - k\sigma_f \tag{4}$$

The result of applying the filter rule (eq. (4)) to the Lap2 dataset (described in the previous section) is shown in Figure 4(a) for a 5 cm resolution and 4(b) for a 10 cm resolution. The light and dark gray points are the up and down laser scan data and the black points are the ghost data that should be eliminated. Any point that is filtered is shown on the floor plane as a white point. It is clear from both images that few ghost points are filtered; the majority of the affected points are from the static scene, not from other robots.

Table 2 evaluates the four metrics described in the previous section for this filtering rule. The percentage of ghost data removed is small for either resolution, as is evident in the images. There is a large improvement in the plane error and in the percentage of points that are inliers for the plane, but this is due to the removal of other low-frequency points rather than the ghost points.

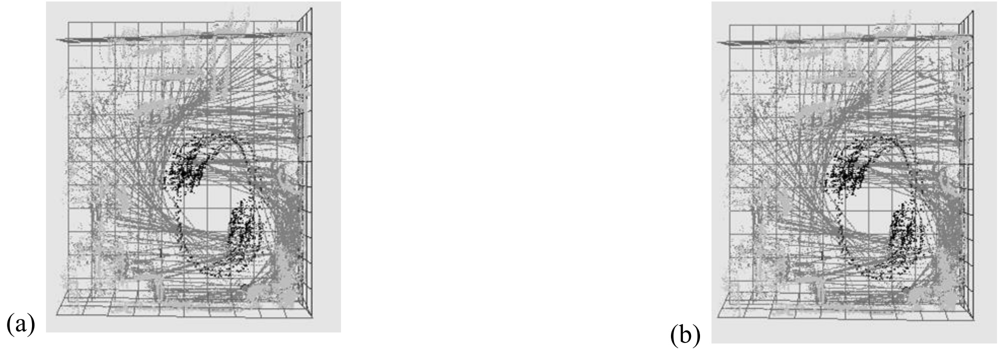

Figure 4: Map-based frequency filter for resolution *r=0.05 m* (a) and *r=0.1 m* (b). Up-laser shown in light, down-laser in darker, ghost points in black and filtered points in white. Original floor and wall planes are just for reference.

Table 2: Evaluation of map-based frequency filter.

Resolution	0.05 m	0.01 m
% Ghost points	28	26
% Other points	10	3
% Impr. Plane Error	42	83
% Impr. Plane Inliers	17	13

Table 3: Evaluation of map-based visibility filter.

Resolution	0.05 m	0.01 m
% Ghost points	43	59
% Other points	15	19
% Impr. Plane Error	35	48
% Impr. Plane Inliers	31	38

Method 2: Map-based Visibility Filtering.

Just looking for low frequency voxels catches relatively few ghost points. Background estimation in computer vision relies on the fact that foreground objects temporarily occlude background objects. A second analogy to exploit therefore is that the moving robots that give rise to the ghost data points occlude floor data points. Hence, if for any point we scan we find a voxel lower (smaller in Z) in the voxel array, then we should filter this point:

$$Filter(p)\ if\colon\ b = \left(\frac{x}{r},\frac{y}{r},\frac{z}{r}\right)\ and\ \exists b' = \left(\frac{x}{r},\frac{y}{r},k\right) \wedge \left(k < \frac{z}{r}\right) \tag{5}$$

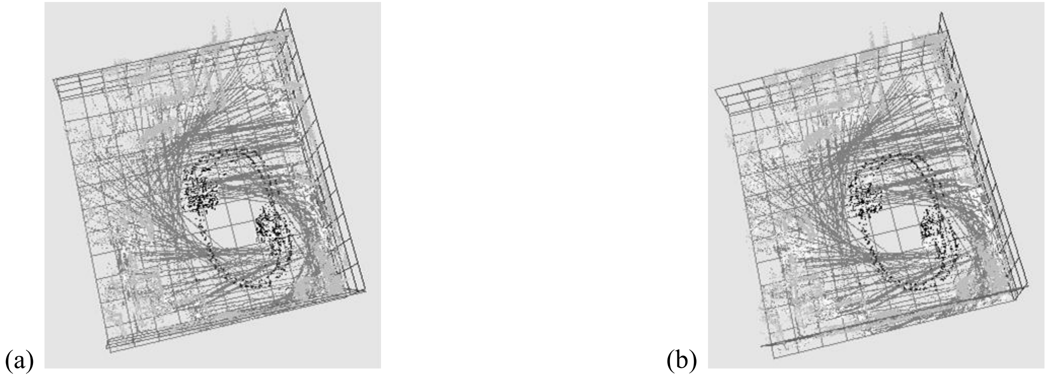

Figure 5: Map-based visibility filter for resolution (a) *r=0.05 m* and (b) *r=0.1 m.*

The result of applying the filter rule (eq. (5)) to the Lap2 data set is shown in Figure 5(a) for a 5 cm resolution and 5(b) for a 10 cm resolution using the same shading conventions as described for Figure 4. It is clear from the images in Figure 5(a) and 5(b) that this method does indeed capture more of the ghost data points. However, it's also clear that it still also filters a lot of other, static points. Because a number of points will typically be placed into a voxel in V (more for the 10 cm resolution and less for the 5 cm resolution), and since one of the points will be the lowest, then the rest will be filtered to this lowest point.

Table 3 shows the metrics for this filtering rule. Substantially more ghost points are filtered than for frequency based filtering, though more background points are affected as well. The plane error and percentage of inliers are improved.

5.2 Robot-Based Methods

The map-based methods seem to demonstrate a shotgun style approach to ghost detection, affecting the map widely in the hope of catching some ghost points. We will also look at two robot-based methods, methods that use some information about the position of other robots to do a more targeted filtering.

Method 3: Robot-based position filtering.

In the system shown in Figure 1, each robot transmits its position in its local coordinate frame back to the map server, which can in turn broadcast all robot positions back to each robot. Information on the location of the other robots can be used to eliminate the transient scans due to other robots, the ghost data, as soon as a scan is made. However, the location of any robot cannot be known with certainty by another robot in this framework because of three factors: Each robot only knows its own location with some degree of uncertainty based on its odometry and localization procedures; When a position is sent to the Map Server, it is then transformed to a global, map-based coordinate frame, whose registration with the robot frame is again only known with some degree of uncertainty; Finally, when a robot receives a list of other robot locations, it can only tell with some degree of uncertainty how the robots may have moved since the positions were measured. Thus, if we use robot position as a ghost filtering mechanism, we have to include a model of uncertainty.

We subsume the three uncertainty factors above into a single (multivariate) Gaussian uncertainty model for robot location $N(\mu_r , \sigma^2_r)$ where μ_r is the reported position of the other robot and σ^2_r is a constant based on the physical size of the robot. This gives rise to the following rule for robot position based filtering (where $d_r(p)$ is the distance from μ_r to p):

$$Filter(p)\ if: \quad d_r(p) < k\sigma_r \tag{6}$$

The result of applying the filter rule (eq. (6)) to the Lap2 dataset is shown in Figure 6. The comparison to Figures 4 and 5 is immediate; the filtering rule has specifically targeted the ghost points with very little other points affected. The metrics for this method are shown in Table 4. This method catches an impressive 93% of the ghost data points and only affects 5% of other data points. However, it also results in an increase in the plane error (-17%) and a small change to the percentage of inliers covered.

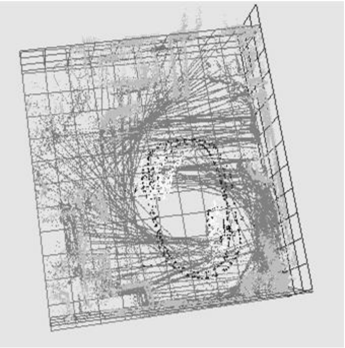

Figure 6: Robot Position Filtering.

Table 4: Evaluation for Robot Position Filtering.

Resolution	0.05m
% Ghost points	93
% Other points	5
% Impr. Plane Error	-17
% Impr. Plane Inliers	4

Method 4: Robot-based direction filtering.

Given the many sources of uncertainty relating to robot position, we investigate an alternate model that only involves a robot knowing, with an associated range of uncertainty, the direction to the other robot. We quantify this direction as a line L between the two robot positions and an angle ha^o on each side of L. This is illustrated in Figure 7(a), where R1 is the robot doing the filtering and R2 is any other robot that might give rise to ghost points. Let $d_p(p,L)$ be the perpendicular distance from the point p to the line L. In Figure 7(a), p_L is the (unknown) closest point on L to p, and the distance from p_L to $R1$, $d_r(p_L)$ is given by $d_r(p_L)^2 = d_r(p)^2 - d_p(p,L)^2$. If the point p is on the extreme edge of the triangle of uncertainty (dotted lines in Figure 7(a)), then the distance $d_p(p,L)$ is constrained by $d_p(p,L) = d_r(p_L)tan(ha)$ and we can introduce the filtering rule:

$$Filter(p)\ if:\ \ d_p(p,L) \leq d_r(p_L)tan(ha) \qquad (7)$$

The result of applying the filter rule (eq. (7)) to the Lap2 dataset is shown in Figure 7 (b) for $ha=5^o$ and in 7(c) $ha=7^o$. This method targets the points being filtered much better than the map-based methods. Looking at the metrics for this method in Table 5, this approach does not focus on the ghost data as well as the position-based rule (eq. (6)) but it also does not negatively impact the plane fit measures (probably because it affects less other points than the position-based rule).

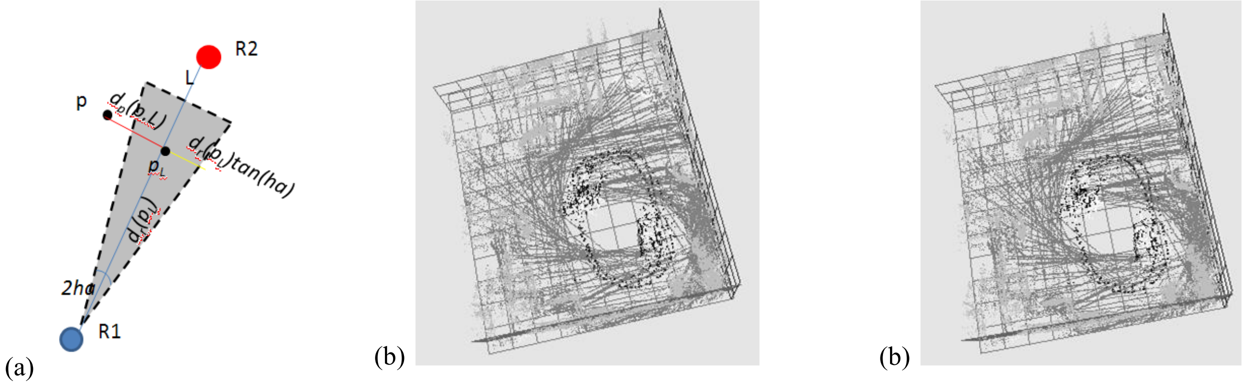

Figure 7: Robot-based direction filter (a); results for resolution (b) $ha=5^o$ and (c) $ha=7^o$.

Table 5: Evaluation of robot-based direction filter.

Half angle (deg.)	5	7
% Ghost points	50	66
% Other points	1.5	3
% Impr. Plane Error	13	4
% Impr. Plane Inliers	2	3

5.3 Combined Method

Since both robot and map-based methods seem to have pros and cons, a natural question is to consider whether a combination can deliver a '1-2 punch' and eliminate a substantial portion of the ghost points while not negatively impacting the plane error measures. In fact, the system architecture in Figure 1 lends itself naturally to such a combination. A robot-based filtering method can be implemented on each robot server and the locally-filtered data send to the central Map Server. The server can then apply a map-based filtering method on the combined data.

Since the robot-based direction filter (eq. (7)) demands less precise knowledge of robot locations we select that as the robot filter. It doesn't capture all the ghost points, but it also doesn't negatively impact the planer error measures. We select the map-based visibility filter (eq. (5)) as the map filter since it also captures a large percentage of ghost points and it shows a good improvement in the planer error measures. The result of apply the sequential combination to the Lap2 dataset is shown in Figure 8 with metrics shown in Table 6. This method captures approximately 91% of the ghost points and results in a 98% improvement in the plane error.

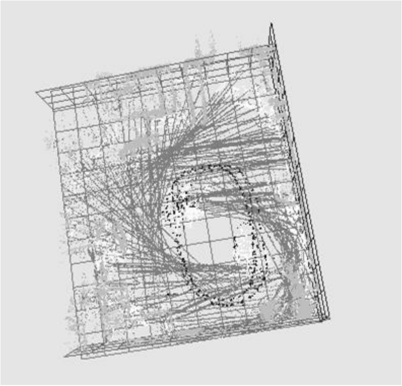

Figure 8: Combined Visibility and Direction Filtering.

Table 6: Evaluation for Combined Filtering.

Half angle (deg.)	7
Resolution	0.1 m
% Ghost points	91
% Other points	20
% Impr. Plane Error	98
% Impr. Plane Inliers	40

6. CONCLUSIONS

This paper has addressed the problem of removing the extraneous data, ghost data, caused when a team of robots working in close proximity scans a 3-D area and team members intermittently scan each other. Using a two robot team, each configured with a tilted laser scanner so as to generate 3-D scan data with robot motion, we generated a joint dataset that includes ghost data. The fit of the floor plane to the joint dataset was used as a measure of the effect of ghost data, and we showed that an up to 98% improvement in fit was possible if all the ghost data was manually removed. We then presented two map-based and two robot-based methods for automatic ghost removal. While each method had pros and cons, we then showed that a selection of a robot direction-based filter, using only information about the uncertain relative direction of robot team members and operating on the robot, and a map visibility-based filter operating at the central map server removed 91% of the ghost data and produced a 98% improvement in floor plane fit.

The map-based methods presented here are basically dynamic object removal methods, since they use no information about the robot team members. As such they affect the dataset widely and may remove important scene geometry. That aspect is not measured in this paper, and future work will include a metric to measure this. For example, the map-based visibility feature, which works best in our combination, can also remove 3-D data for tables and other horizontal surfaces that are suspended above the floor.

The robot-based methods presented include uncertainty information about the robot location. However, the parameters for the uncertainty (the variance in the position filter, or half-angle in the direction filter) were not linked to the Kalman filter based robot position estimates in the map server. Future work will complete this connection.

Finally, the robot team included two laser-based robots. We have also integrated a stereovision based robot into the data collection architecture with the objective of leveraging image-based filtering methods. Ortega and Andrade-Cetto [9] recently presented one such method for removing dynamic objects based on integrating scanning laser and video and we can include this in our evaluations if we incorporate stereo data.

References

[1] Carpin, S., “Merging Maps via Hough Transform,” *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems* (2008).

[2] Gutchess, D., Trajkovic, M., Cohen-Solal, E., Lyons, D., and Jain, A.K., “A background model initialization algorithm for video surveillance,” *IEEE ICCV 2001*, vol. 1, pp. 733—740 (2001).

[3] Konolige, K., Grisetti, G., Kummerle, R., Limketkai, B., and R. Vincent, “Efficient sparse pose adjustment for 2D mapping,” *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems* (2010).

[4] Kaushik, R., Xiao, J., Morris,W., and Zhu, Z., “3D Laser Scan Registration of Dual-Robot System Using Vision,” *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems* (2009).

[5] Litomisky, K., and Bhanu, B., “Removing Moving Objects from Point Cloud Scenes,” *Int. Workshop on Depth Image Analysis*, Tsukuba Japan (2012).

[6] Liu, Y., Thrun, S., “Gaussian Multirobot SLAM,” *Advances in Neural Information Processing Systems*, NIPS 2003, Vancover Canada (2003).

[7] Lu, F., and Milios, E., “Globally Consistent Range Scan Alignment for Environment Mapping,” *Autonomous Robots* 4 (1997).

[8] Nuchter, A., “Parallelization of Scan Matching for Robotic 3D Mapping,” *European Conference on Mobile Robots,* Freiburg Germany (2007).

[9] Ortega, A., and Andrade-Cetto, J., “Segmentation of dynamic objects from laser data” *European Conference on Mobile Robots*, Orebro Sweden (2011).

[10] Piccardi, M., “Background subtraction techniques: a review,” *IEEE International Conference on Systems, Man and Cybernetics*, The Hague, Netherlands, October (2004)**.**

[11] Reid, R., Bräunl, T., “Large-scale Multi-robot Mapping in MAGIC 2010,” *IEEE Robotics, Automation and Mechatronics*, Qingdao China (2011).

[12] Wangsiripitak, S., Murray, D.W., “Avoiding moving outliers in visual SLAM by tracking moving objects,” *IEEE Int. Conf on Robotics and Automation*, Kobe Japan, May (2009).