3D Reconstruction of Environments for Planetary Exploration

Sébastien Gemme, Joseph Nsasi Bakambu and Ioannis Rekleitis
Canadian Space Agency, Space Technologies
6767 Route de l'aéroport, Saint-Hubert, Québec Canada J3Y 8Y9
[sebastien.gemme,joseph.bakambu,ioannis.rekleitis]@space.gc.ca

Abstract

In this paper we present our approach to 3D surface reconstruction from large sparse range data sets. In space robotics constructing an accurate model of the environment is very important for a variety of reasons. In particular, the constructed model can be used for: safe tele-operation, path planning, planetary exploration and mapping of points of interest. Our approach is based on acquiring range scans from different view-points with overlapping regions, merge them together into a single data set, and fit a triangular mesh on the merged data points. We demonstrate the effectiveness of our approach in a path planning scenario and also by creating the accessibility map for a portion of the Mars Yard located in the Canadian Space Agency.

Keywords: *Automatic 3D Registration, Surface Reconstruction, Scene Reconstruction, Tele-operation, Path Planning, Terrain Traversability.*

Figure 1. The mobile robot used for navigation tasks at the Mars yard.

1. Introduction

In this paper we consider the problem of constructing a 3D environment model for a variety of space robotics applications. In general, creating a model of the environment is a very important task in robotics. In space robotics in particular, such a model becomes indispensable. The motivation of this work comes from a variety of problems in space exploration and operation. We propose an improvement for tele-operation approaches like the ones presented in Dupuis *et. al.* [14], Borst and Volz [6] , Kim and Bejczy [21] and Lipsett *et. al.* [24]. The main problem of tele-operating remotely located equipment in space is delays. Such delays can vary from a few seconds, i.e. when communicating with the International Space Station (ISS), to a few minutes when communicating with a rover located on Mars. To overcome this problem, we can incorporate the delays in the planning process, but when delays grow larger than a few seconds the risk of an accident increases to unacceptable levels. The latency between operator and robot precludes in many cases real time tele-operation. The alternative is to send a sequence of (more complex) commands which the robot executes and then reports back upon completion. In such a scenario, the commands used need to be verified for effectiveness and safety before send. One approach to achieve that is to create a model of the remote environment and use it to simulate the commands on the ground. Once a command is validated, it is uploaded to the remote location to be executed. This way, the delays are no longer a problem. The main drawback of this method is that most of the time the model used on the ground it is not up-to-date. For example, if the Space Station Remote Manipulator System (SSRMS) on the ISS is tele-operated, the model used is most likely the one that originates from the latest update of the ISS. That model is most likely outdated by the time we wish to use it, as more modules are added constantly, resulting in serious risks for safety, since a collision in space can have tragic consequences. For this reason, a system for 3D reconstruction of environments

Proceedings of the Second Canadian Conference on Computer and Robot Vision (CRV'05)
0-7695-2319-6/05 $ 20.00 **IEEE**

would be very useful on the ISS, to rebuild the environment where the arm will operate. Once the model is rebuilt, the model is downloaded to the ground to be used for simulation purposes before sending the real command.

Planetary exploration is another application where 3D reconstruction is crucial. A similar system to the one proposed here was used to control Sojourner during the Mars Pathfinder mission. In our approach the 3D reconstruction system is used for a dual purpose. First, on-site, the rover uses the range data for obstacle avoidance and to replan more efficient trajectories. Second, on earth, updated models of the Martian terrain are used for asserting the feasibility/safety of the commands to be uploaded to the rover, and also for identifying areas of interest (in conjunction with visual data) for the rover to investigate.

Currently, the data is acquired using a LIDAR device from Optech (see Figure 2), a laser scanner that offers great performances in terms of the quality of the acquired data. The challenge is to deal with large data sets, as each view has about 500K 3D points, and the combination of multiple views is required. Therefore, the data sets can easily grow to millions of points. Any approach used has to be robust for high volume of data.

Figure 2. The LIDAR sensor from Optech.

Another challenge in this application is the sparsity of the point cloud, and the fact that the density is not constant. The non-uniform density is due to the perspective nature of the data acquisition as can be seen in Figure 3. The further the points are, the sparser they get. Moreover, when the angle between the terrain and a scanline of the LIDAR is small, the distance between two consecutive points increases rapidly. Figure 5 further illustrates that effect giving a topographical map impression. As such, an algorithm that is robust in the face of irregular sampling was chosen.

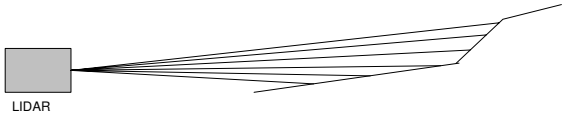

Figure 3. Illustration of the cause of the undersampling

The process of reconstruction involves multiple views. The assembly of different views has to be made in an automatic way in order to estimate the rigid transformation between the views. Once the first estimate is computed, an Iterative Closest Point (ICP) algorithm is used for the final registration since this algorithm has proven to be very good with rigid transformations. The next section presents related work. Section 3 contains the description of our approach. Experimental results and discussion of various issues are in section 4. Finally, section 5 contains our conclusions and discussion of future work.

2. Background

During the construction of 3D models of the environment, multiple views have to be combined in order to compensate for the limitations of the field of view of the sensor and for self occlusions. Besl and McKay [5] and Zhang [31] from INRIA introduced the ICP (Iterative Closest Point) algorithm for the registration of 3D shapes. ICP is widely used and many variations have been developed over the years; some of them can be found in Rusinkiewicz and Levoy [28], Greenspan and Godin [16] and Langis *et. al.* [22]. In general, for these algorithms to perform satisfactorily a good estimate of the two views that are be registered (merged) needs to be established. In other words, a preliminary step of estimating the rigid transformation $(t_x, t_y, t_z, \theta_x, \theta_y, \theta_z)$ between two views has to be performed. This step consists of identifying common points in the two cloud of points. From these common points, a good estimate of the rigid transformation can be determined and used as a starting point for the ICP algorithm. For a comprehensive survey of those methods please refer to Campbell and Flynn [8].

The main idea of the automatic registration methods is the detection of common features in two views. Many methods reduce the 3D shape into 2D in order to do the matching, like the spin image approach pre-

sented in Johnson and Hebert [20], the points finger print work presented in Sun *et. al.* [29] or other types of 2D signatures like the ones presented in Burtnyk and Greenspan [7]. Huber and Hebert [19] present a fully automatic method that assembles multiple views coming from 3D sensors. Another very interesting way of solving the problem of automatic surface matching is to formulate the problem as an optimization question and to use a genetic algorithm to search for the best match, as in Chow *et. al.* [10]. The problem with using genetic algorithms is that the optimality of the solution can never be proven.

It is worth noting that, when registering two views that do not have a big difference in their pose, ICP works without any need for prior pose estimation, as observed in the Great Buddha project by Nishino and Ikeuchi [27].

The field of 3D scene reconstruction from a set of points has a long history and can now be considered fairly mature. The main objective is to reconstruct a surface from a set of points in such a way that discrepancies between the points and the surface are minimized. Of particular interest are free-form surfaces [4] which have well defined normals everywhere (with a few exceptions). Planar and quadratic surfaces are of particular interest [8]. A common approach is using NURBS (Non-Uniform Rational B-Spline), but sometimes NURBS-surfaces are impossible to accurately fit on point clouds [4]. Polygonal meshes continue today to be the most popular choices; for a more extensive review please refer to Campbell and Flynn [8].

Hoppe *et. al.* [18] proposed an algorithm that is robust to undersampling and can handle large volumes of data. Their marching cube approach is an extension of the Lorenson and Cline [26] work. The main idea is to divide the space into cubes and retrieve the crossing points of the surface (that is, at this stage, still represented with points) with the cubes. By collecting these intersection points we can rebuild the mesh. Following that, a mesh simplification is applied. Finally, a subdivision surface is generated. A variant of this method was used in the Michelangelo project by Levoy *et. al.* [23] that also had to deal with very large data sets.

Another approach to free-form surface generation is based on the Delaunay triangulation, see Amenta *et. al.* [2, 1]. The original approach was appropriate for small data sets and uniform sampling. Dey and Giesen [11] have extended the previous method to deal with undersampling – an important consideration when dealing with range sensors. Further variations of these algorithms has been developed in order to address the problem of large data set using a Delaunay based method; *e.g. SUPERCOCONE* by Dey *et. al.* [12]. Torres and Dudek [30] combine information from intensity and range images in order to compensate for sparse data, and fill in any gaps.

By adding many views together another problems occurs: the addition of registration error which know as *Multi-view optimization* or *Global Registration*. Many methods have been developed to overcome this problem, Campbell and Flynn [8] give a review of those methods. Chen and Medioni [9] proposed to register a new view according to not only the view's neighbors that are already registered but to all the merged data, this way, the error accumulation is somehow avoided. Eggert *et. al.* [15] proposed a force-based optimization to that problem. In this approach, the connection between the views are modeled using a springs.

3. Environment Reconstruction

The reconstruction of the environment is performed in two steps. The first step consists of the assembly of the different views by estimating the rigid transformation between the poses from where each view was taken. The second step is the surface reconstruction, achieved by fitting a triangular mesh on the point cloud that combines the data from all views. Next, we discuss the choice algorithms depending on the nature of the data: undersampled and large data sets.

3.1. Assembly of the different views

Complete 3D reconstruction of a free-form surface requires acquisition of data from multiple viewpoints in order to compensate for the limitations of the field of view and for self occlusion. In this paper, we used a LIDAR for scanning views of a 3D surface to obtain $2\frac{1}{2}$D images in the form of a cloud of points. These views are then registered in a common coordinate system. Since the coordinates of the viewpoint may not be available or may be inaccurate, the original ICP in Besl and McKay [5] may not converge to the global minimum. Thus, to assemble all views in the same coordinate frame, we used a variant of ICP, which differs from the original ICP by searching for the closest point under a constraint of similarity in geometric primitives. The geometric primitives used in this paper are the normal vector and the change of geometric curvature. The change of geometric curvature is a parameter of how much the surface formed by a point and its neighbors deviates from the tangential plane [3], and is invariant to the 3D rigid motion. Hence, in our algorithm, surface points are represented in $\Re^7$. Coordinates of a point P on the surface are $(x, y, z, n_x, n_y, n_z, k)$ where $[x, y, z]$ are the Cartesian coordinates of P, $[n_x, n_y, n_z]$

are the coordinates of the normal vector and k is the change in curvature.

Geometric primitives are used in matching by incorporating them in a 7D distance metric D_α of the form:

$$D_\alpha^2(p,q) = \sum_{i=1}^{7} \alpha_i(\lambda_{pi} - \lambda_{qi})^2$$

where the λ_i are coordinates of a 7D point and the α_i are the weights of each coordinate. Using this distance metric for finding the closest point is a combination of the 3D distance, the difference of the orientation of the normal vectors, and the difference of the change of curvature.

Our ICP algorithm can be summarize as follow: Let $P = \{p_1, p_2, \ldots, p_n\}$ and $Q = \{q_1, q_2, \ldots, q_m\}$ be two sets of points in $\Re^7$. The goal is to find a rigid transformation $T = (R, t)$ composed of a rotation matrix R and a translation vector t that best aligns P to match Q. An informal description of the algorithm follows:

1. Compute the normal vector and the change of curvature at each point of each cloud of points P, Q. Build a k-D tree representation of the cloud of points Q .
2. Initialize the matching process.
3. Repeat until the termination criterion is reached:
   - Compute the closest points $Y_i^k = ClosestPoint(P_i^k, Q)$. Where $i = 1, 2, \ldots, n$, $Y_i^k \in Q / D_\alpha(P_i^k, Y_i^k) = min(D_\alpha(P_i^k, Q))$. A k-D tree is used to speed up this search.
   - Discard undesired matches through statistical analysis of the distances, as described in Zhang [31].
   - Compute the rigid transformation $T = (R, t)$ from the remaining matches, as in Zhang [31] and Besl and McKay [5].
   - Apply the rigid transformation to all points in P : $P_i^{k+1} = RP_i^0 + t$ and rotate accordingly the normal vectors.
   - If the mean square error drops below a threshold, TERMINATE.

To illustrate the performance of different view assemblies, several views of the Mars Yard (figure 7) were taken from different viewpoints. Figure 4 illustrates three different views and figure 5 gives the result of the assembled views using the above ICP algorithm.

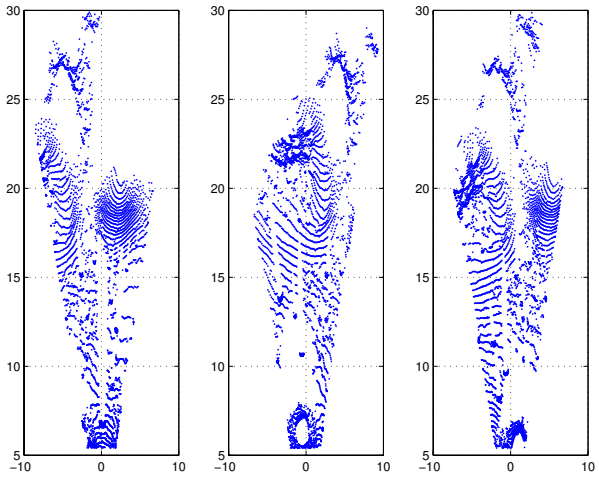

Figure 4. The three scans to be assembled

3.2. Surface Reconstruction

Our approach to surface reconstruction has to be robust in the face of data discontinuities and efficient in order to deal with large data sets. Our current implementation of surface reconstruction uses a variant of the method described by Hoppe *et. al.* [18].

Figure 5. The combined results in a single data set. Note the non-constant sampling.

The method can be classified as a marching cube. To achieve the goal of retrieving the surface, a division

of space into cubes is performed. Hoppe [17] suggests to set the cube size equal to the sampling size; experimental results have shown finer reconstruction with a cube size of 80% of the sampling size. However, as mentioned earlier, the data points are not uniformly sampled from the environment, therefore a global maximum value of sampling has to be determined. This is done by taking different samples on the cloud of points; each sample contains a center point and its k closest neighbors. From those neighbors, an average distance between the points is computed. By taking a random selection of these samples, a good estimate of the suitable sampling distance of the cloud of points can be calculated.

Once the set of cubes is generated, a signed distance from the surface to every vertex of each cube has to be estimated. That signed distance is defined as:

$$\tilde{d}_U(p) = (p - o_i) \cdot \hat{n}_i$$

where p is the point for which the signed distance has to be computed, in this case a vertex of a cube. U is the surface, and o_i is the center of the closest tangent plane from p. Finally, $\hat{n}_i$ is the normal of that closest tangent plane.

By estimating the signed distance, the points where the surface is crossing the cubes can be determined. For example, if, for a cube edge, one vertex has a signed distance less than zero and the other has a signed distance greater than zero, the surface crosses that edge. Then, by collecting all the crossing points, a triangulation is generated.

The accuracy of the crossing point estimation directly affects the estimation accuracy of the surface. For an edge that satisfies the condition mentioned above, let v_i be the vertex that has a positive signed distance from the surface and let v_j be the vertex that has a negative signed distance from the surface. The crossing point is located at a distance $\tilde{d}_U(v_i)$ from vertex v_i on the line defined by v_i, v_j.

To calculate the crossing point, we used a parametric representation of the line v_i, v_j using a Hermite polynomial that is defined as follows:

$$\vec{P}(u) = [H_1, H_2, H_3, H_4] \begin{bmatrix} \vec{P}(0) \\ \vec{P}(1) \\ \vec{P}'(0) \\ \vec{P}'(1) \end{bmatrix}$$

$$\begin{aligned} H_1 &= 2u^3 - 3u^2 + 1 \\ H_2 &= -2u^3 + 3u^2 \\ H_3 &= u^3 - 2u^2 + u \\ H_4 &= u^3 - u^2 \end{aligned}$$

where $\vec{P}(0)$ and $\vec{P}(1)$ are the starting and ending points of the curve (in this case v_i and v_j). $\vec{P}'(0)$ and $\vec{P}'(1)$ are the starting and ending tangent values; since a straight line is represented, those values are set to zero.

The polynomial $u = \tilde{d}_U(v_i)/d(v_i, v_j)$ is evaluated, where $d(\cdot, \cdot)$ is the Euclidean distance between v_i and v_j. Thus, the crossing point p becomes:

$$p = v_i(2u^3 - 3u^2 + 1) + v_j(-2u^3 + 3u^2)$$

In order to compute the signed distance, the tangent plane associated with every point of the cloud of points is computed. Note that, the normals of the tangent planes need to have consistent direction (all point to the same side).

Computing the tangent plane at point x_i in the point cloud is performed as follows:

1. Find the neighbors of x_i within the sampling distance α: $Nbhd(x_i)$.
2. Compute the center of $Nbhd(x_i)$: o_i, which becomes the center of the tangent plane.
3. Find the normal of the tangent plane. First, compute the covariance matrix as follows:

$$CV_i = \sum_{y \in Nbhd(x_i)} (y - o_i) \otimes (y - o_i)$$

where $\otimes$ is the outer product. Then the normal can be found by using the Singular Value Decomposition (SVD) of CV_i.

The above procedure is performed for every point in the cloud of points. Once this procedure is completed, the normals are known but their direction is not consistent, therefore, they all have to be processed in order to have consistent direction and thus provide a signed distance. To achieve this task, an undirected, weighted graph of all the vertices is built. An edge is created between two points if they are closer than a sampling distance α from each other. The weight of an edge connecting two points p_i, p_j is $1 - |\hat{n}_i \cdot \hat{n}_j|$. This means that the weight will be less if two points are on a similar part of the surface. To orient the normals, we find the minimum path to cover the graph. We used a *breadth-first* algorithm to perform this task. While going through the minimum path, the normals should be pointing in a similar direction due to the weight that has been defined for an edge. Following that idea, if, for two normals $\hat{n}_i, \hat{n}_j$, $\hat{n}_i \cdot \hat{n}_j < 0$, then $\hat{n}_j \leftarrow -\hat{n}_j$.

We dealt with the non-uniform sampling effect, in our implementation of [18], by specifying a larger sampling distance, which results in using larger cubes in the marching cubes process. As suggested in [17], we

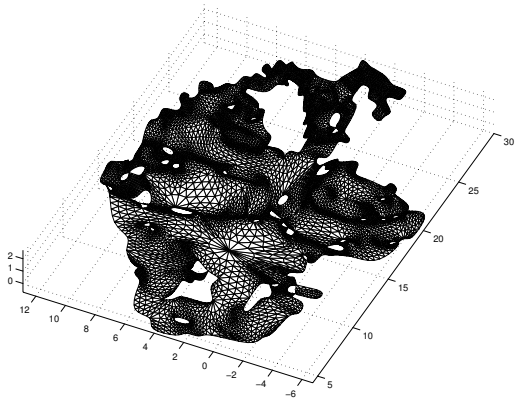

Figure 6. Mesh reconstruction

then apply a decimation filter to simplify the mesh, finally, a *Loop* subdivision [25] is applied to converge to the initial surface.

A *Loop* subdivision is a refinement scheme that transforms a coarse mesh to a more refined one using a set of rules. Those rules are applied while adding new vertices to a triangular mesh; different rules apply when border vertices and inside vertices are used. One interesting property of subdivision surfaces is that they converge to a limit surface. In our case, the decimation creates the initial control mesh used by the *Loop* subdivision. By subdividing, the control mesh converges to the limit surface, in our case, the limit surface is very close to the initial cloud of points.

Figure 6 shows the final result of the three registered views of the reconstructed mesh which provide pretty accurate estimate of the terrain.

4. Experimental Results

We have tested our approach in the terrain of the Mars Yard constructed outside the Canadian Space Agency (CSA). Figure 7 offers a view of the yard with the CSA building in the background. Three scans were taken from the same approximate location where the photograph in Figure 7 was taken. At the bottom part of the center and right scan in Figure 4, the crater visible in Figure 7 is easily detected. Figure 5 presents all the range points merged in a single data set and Figure 6 presents the resulting surface. Next we are going to discuss the use of the reconstructed surface in two applications: path planning and mapping the terrain traversability.

Figure 7. A view of the Mars Yard located at CSA Headquarters in St. Hubert, Quebec Canada.

4.1. Path Planning

As mentioned earlier, 3D terrain/scene reconstruction is important in planetary exploration tasks in order to plan trajectories safely before the remotely located rover is instructed to execute them. In Figure 8 we present a trajectory generated using the reconstructed surface, starting at a location near the viewpoint from where the first scan was taken.

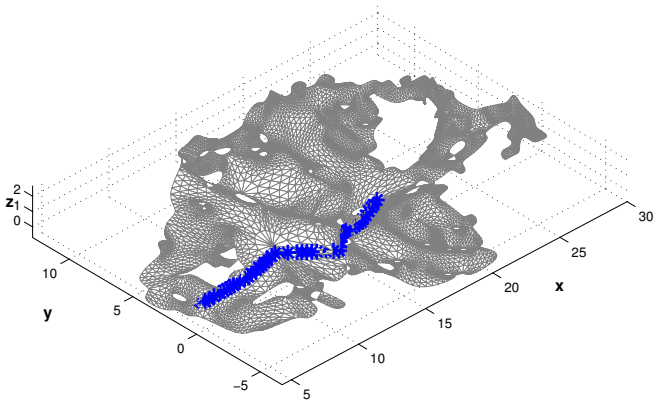

Figure 8. Path planning on the reconstructed surface

The triangulation data structure selected for the terrain representation provides valuable information in the form of the connectivity graph that allows for efficient retrieval of the adjacent triangles from the current triangle. See [13] for details.

4.2. Traversability Map

Closely associated with path planning is the concept of traversability maps. Such a map indicates how suitable is an area for the rover to pass through. Moreover, flat/accessible areas surrounded by steep/inaccessible sections can be also eliminated. Path planning on a preprocessed traversability map increases the efficiency by eliminating paths that lead to dead ends. Figure 9 present a shade coded accessibility map based on the terrain seen on Figure 6.

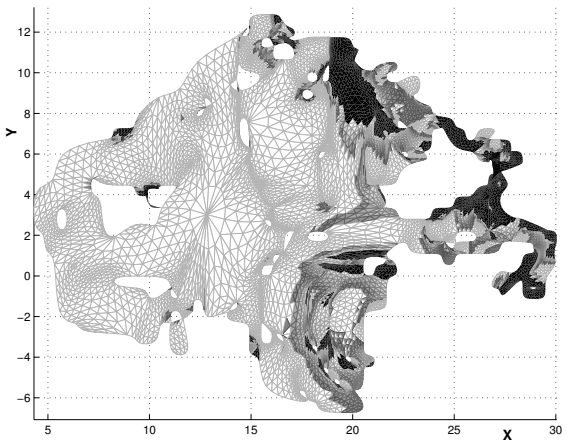

Figure 9. Top view of the traversability map. Light gray colored triangles means accessible terrain, dark gray is accessible with difficulty, black is inaccessible, and white means no data. The mesh is drawn also with slightly darker color.

A narrow corridor is visible in the middle right of the terrain in Figure 9 and it is this corridor the path planner used in Figure 8. The two white regions at the left of the terrain signify no-data zones where the view was obstructed by crater elevation in front of them.

5. Conclusions

In this paper we presented a methodology for the 3D terrain reconstruction from a set of clouds of points. The first step was to establish the coordinate transformation among the different viewpoints and to merge all the clouds of point into a single coherent data set. Then we presented an extension to a well known algorithm for constructing a surface from that data set.

Experimental results from the Mars Yard (located at CSA) were used to validate our approach. Furthermore, the resulted surface was used for planning a safe path and for constructing a map of the terrain traversability.

In the immediate future we are planing to take a complete scan of the Mars Yard (located at CSA) and build a precise terrain model. The resulting terrain then is going to be incorporated in existing mobile robot simulation packages (such as the Player/Gazebo package) in order to perform experiments during the winter months when the Mars Yard is inaccessible. Furthermore, in future work we plan to combine range data with intensity information in order to create more realistic scene models.

Acknowledgements: We would like to thank the people at the Robotics group in Space Tech. at CSA for their valuable contributions in numerous discussions. Special note goes to P. Allard for his help with path planning and T. Lamarche for help with the terrain reconstruction. E. Dupuis provided valuable insights during the duration of this project.

References

[1] N. Amenta, S. Choi, T. K. Dey, and N. Leekha. A simple algorithm for homeomorphic surface reconstruction. *Int. Journal of Computational Geometry and Applications*, 12:125–141, 2002.

[2] N. Amenta, S. Choi, and R. K. Kolluri. The power crust. In *SMA '01: Proc. of the sixth ACM symposium on Solid modeling and applications*, pg. 249–266. 2001.

[3] K.-H. Bae and D. D. Lichti. Automated registration of unorganised point clouds from terrestrial laser scanners. In *The 20th ISPRS Congress*, Istanbul, Turkey, July 2004.

[4] P. Besl. The free-form surface matching problem. In *Machine Vision for Three-Dimensional Scenes*, pg. 25–71, San Diego, 1990. Academic Press.

[5] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[6] C. W. Borst and R. A. Volz. Telerobotic ground control of a space free-flyer. In *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, Victoria, BC, Canada, Oct. 1998.

[7] N. Burtnyk and M. Greenspan. Multiple view registration of range data using signature. In *Proc. of the American Nuclear Society Sixth Topical Meeting on Robotics and Remote Systems*, pg. 5–10, Montery, CA, Feb. 1995.

[8] R. J. Campbell and P. J. Flynn. A survey of free-form object representation and recognition techniques. *Com-*

puter Vision and Image Understranding, 81(2):166–210, 1993.

[9] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. *IVC*, 10(3):145–155, April 1992.

[10] C. K. Chow, H. T. Tsui, and T. Lee. Surface registration using a dynamic genetic algorithm. *Pattern Recognition*, 37(1):105–117, 2004.

[11] T. K. Dey and J. Giesen. Detecting undersampling in surface reconstruction. In *SCG '01: Proc. of the seventeenth annual symposium on Computational geometry*, pg. 257–263. ACM Press, 2001.

[12] T. K. Dey, J. Giesen, and J. Hudson. Delaunay based shape reconstruction from large data. In *IEEE Symposium in Parallel and Large Data Visualization and Graphics (PVG2001)*, pg. 19–27, 2001.

[13] E. Dupuis, P. Allard, J. Bakambu, T. Lamarche, and W. Zhu. Toward autonomous long range navigation. In *Advandced Space Technologies for Robotics and Autonomous (ASTRA) 2004*, Netherlands, Nov. 2004.

[14] E. Dupuis, G. R. Gillett, P. Boulanger, E. Edwards, and M. G. Lipsett. Interactive, intelligent remote operations: Application to space robotics. In *Proc. of the SPIE Telemanipulator and Telepresence Technologies VI*, Boston, MA, Sept. 1999.

[15] D. W. Eggert, A. W. Fitzgibbon, and R. B. Fisher. Simultaneous registration of multiple range views for use in reverse engineering of CAD models. *Computer Vision and Image Understranding*, 69(3):253–272, Mar. 1998.

[16] M. Greenspan and G. Godin. A nearest neighbor method for efficient icp. In *3DIM01 : Proc. of the 3rd Int. Conf. on 3-D Digital Imaging and Modeling*, Quebec City, Quebec, Canada, May 2001.

[17] H. Hoppe. *Surface Reconstruction from Unorganized Points.* PhD thesis, University of Washington, Seattle, USA, 1994.

[18] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *Proc. of the 19th annual conference on Computer graphics and interactive techniques*, pg. 71–78. ACM Press, 1992.

[19] D. F. Huber and M. Hebert. Fully automatic registration of multiple 3d data sets. *Image and Vision Computing*, 21(7):637–650, 2003.

[20] A. E. Johnson and M. Hebert. Usgin spin images for efficient object recognition in cluttered 3d scenes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(4):433–449, 1999.

[21] W. S. Kim and A. K. Bejczy. Demonstration of a high-fidelity predictive/preview display technique for telerobotic servicing in space. *IEEE Trans. on Robotics and Automation*, 9(5), Oct. 1993.

[22] C. Langis, M. Greenspan, and G. Godin. The parallel iterative closest point algorithm. In *3DIM01 : Proc. of the 3rd Int. Conf. on 3-D Digital Imaging and Modeling*, Quebec City, Quebec, Canada, May 2001.

[23] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. The digital michelangelo project: 3d scanning of large statues. In *SIGGRAPH '00: Proc. of the 27th annual conference on Computer graphics and interactive techniques*, pg. 131–144. ACM Press/Addison-Wesley Publishing Co., 2000.

[24] M. Lipsett, W. Ballantyne, and M. Greenspan. Virtual environments for surface mining operations. *CIM Bulletin*, 91(1016):80–85, 1998.

[25] C. Loop. Smooth subdivision surfaces based on triangles. Master's thesis.

[26] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *SIGGRAPH '87: Proc. of the 14th annual conference on Computer graphics and interactive techniques*, pg. 163–169. ACM Press, 1987.

[27] K. Nishino and K. Ikeuchi. Robust simultaneous registration of multiple range images. In *Proc. of the Fifth Asian Conf. on Computer Vision*, pg. 454–461, Jan. 2002.

[28] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *Third Int. Conf. on 3D Digital Imaging and Modeling (3DIM)*, June 2001.

[29] Y. Sun, J. Paik, A. Koschan, D. Page, and M. Abidi. Point fingerprint: A new 3-d object representation scheme. *IEEE Trans. On Systems, Man and Cybernetics, Part B*, 33(4):712–717, 2003.

[30] L. A. Torres-Mendez and G. Dudek. Reconstruction of 3d models from intensity images and partial depth. In AAAI, editor, *Proc. of the American Association for Artificial Intelligence (AAAI)*, pg. 476–481, 2004.

[31] Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. *Int. Journal of Computer Vision*, 13(2):119–152, 1992.