Practical Considerations for Implementing ICP Registration Algorithms in a 360° 3D Point Cloud Stitching System

Jesus Sanchez-Perez, Manuel Jimenez, Ph.D. – Advisor, Enrique Torres-Rivera, Nayda Santiago, Ph.D. – Advisor
University of Puerto Rico, Mayaguez, PR 00681-9000
jesus.sanchez1@upr.edu, mjimenez@ece.uprm.edu, enrique.torres2@upr.edu, nayda.santiago@ece.uprm.edu, danilo.rojas@upr.edu

Abstract*– The last two decades have witnessed how three-dimensional (3D) point cloud scanning and registration algorithms have become increasingly popular in areas as diverse as cinematography, robotics, and medicine, among others. Despite their broad application range, such algorithms remain to be computationally demanding and difficult to implement. In particular, the task of choosing and implementing suitable registration-pipeline processes for specific applications continues to be challenging in most practical cases. This paper presents the implementation of a point cloud stitching system to produce 360° 3D images from individual, partial views of a solid model. Performance analyses and evaluations supporting the decision-making process allow for identifying factors leading to the best accuracy and computational speed of the iterative closest point (ICP) registration algorithms considered for the task at hand. The outcomes of our analysis lead to interesting findings related to two well-known ICP variants, while also providing useful implementation guidelines for developing a practical 360° 3D scanning system.

***Keywords*– 3D point cloud, stitching system, ICP registration**

I. Introduction

Three-dimensional (3D) point clouds allow for digitally representing our surrounding world. Once in a digital form, a 3D object image can be processed to obtain properties such as area, volume, distribution and density, among others. With the advent of reliable, high-quality, and cost-effective 3D scanners, 3D point cloud data can be readily obtained for multiple faces of a solid model.

The generation of 3D point cloud data requires two main steps: acquisition and processing. A variety of 3D surface data acquisition techniques have been proposed through the years, including imaging and microwave radar, computed tomography (CT), coded structured light, and holography, among others [1]. From these techniques, coded structured light (CSL) is one of the most cost-effective choices for surface reconstruction. A CSL system consists of three main components: a structured light projector (SLP), an imaging stage, and a processing stage.

The SLP transmits coded patterns onto the surface to be measured. The imaging stage contains one or multiple digital cameras that capture images from the surface of interest when illuminated by the SLP. The processing stage typically uses a triangulation technique that allows for computing the positions of 3D features on the model surface. The pattern-decoding stage can be considered simple as the acquired patterns are already coded. Salvi et al. provide a detailed description on how coded structured light works [2]. As this method is readily accessible, portable, comparably cost-effective, and requires a small number of cameras and a single projector, it was the 3D scanning technique used for the analysis and implementation in this paper.

The processing of point clouds requires efficient and reliable algorithms. Common processing areas include registration, segmentation, filtering, recognition, and visualization, usually available from various open software libraries online. However, not all libraries support all the above processing algorithms. One usually must resort to incorporating multiple independent libraries, which can lead to software incompatibilities. The Point Cloud Library (PCL), provides a reliable and practical solution to these issues since it incorporates other third-party libraries as dependencies. PCL supports all the mentioned processing areas, facilitating the process of implementing a practical solution [3], [4].

Nonetheless, gathering all these components does not solve the entire problem. Additional challenges in the algorithm selection, parameter configuration, data size, computational complexity, quality of 3D point clouds, and validation, need to be correctly addressed to achieve an effective implementation.

In the recent literature, several approaches have addressed the performance evaluation of registration algorithms. Bellekens et al. studied state-of-the-art registration algorithms including ICP and four of its variants. They included a theoretical formulation as well as a performance evaluation in terms of accuracy and computational speed [5]. Salvi et al. provided a classification scheme and an analysis of different coarse and fine range image registration techniques [6]. Attia et al. evaluated the performance of most used 3D point clouds registration algorithms focusing on their suitability for dimensional control. They provided a theoretical and experimental comparison of four registration algorithms, including the ICP [7]. Although these approaches accomplished their analysis goals, they did not provide practical descriptions of the underlying 3D scanning system or reported the usage of structured light projector data. This paper provides a reference for PCL-based applications,

Digital Object Identifier (DOI): http://dx.doi.org/10.18687/LACCEI2018.1.1.62
ISBN: 978-0-9993443-1-6
ISSN: 2414-6390

discussing practical considerations for physical implementation and evaluating algorithmic performance under different registration parameters in the framework of a functional prototype to produce 360° 3D stitched images.

The rest of this paper is organized as follows. Section II provides a theoretical definition of ICP and two of its variants. Section III details a practical 3D scanning setup where algorithms could be evaluated. Section IV describes the experimental setup to acquire evaluation data, the obtained results, and their analysis. Lastly, Section V provides concluding remarks.

II. Iterative Closest Point Algorithms

ICP algorithms provide a framework for resolving geometric registration problems. Registration is of particular interest in computer vision since it allows for image reconstruction. A registration algorithm is able of producing a single point cloud representing the union of different, overlapping point clouds acquired at distinct times and views. Registration algorithms can be classified into rigid and non-rigid. The former assumes point clouds with six degrees of freedom (DoF), whereas the latter deals with additional DoF as it can be applied to object shapes that could change over time.

Following the classification proposed by Rusinkiewicz and Levoy [8], registration algorithms can be seen as a sequence of six steps: (1) point selection, (2) matching, (3) rejection, (4) weighting, (5) assigning error metric, and (6) minimizing error. Hereafter we will use this classification to describe Iterative Closest Point (ICP) algorithms and their variants.

A. *ICP Generalities*

The ICP developed by Besl and McKay [9] belongs to the class of rigid and fine registration techniques that assume an initial alignment is available. Registration can be pairwise, when only two point clouds are processed at a time; or multi-view, where an entire dataset is processed on the spot. Multi-view techniques are beyond the scope of this paper.

In pairwise registration we shall refer to one of the point clouds as the source and the other one as the target. Let the source be $P = (p_1, p_2, ..., p_n)$ and the target $Q = (q_1, q_2, ..., q_m)$, where p_i and q_j represent individual points of each cloud. The goal of an ICP is to iteratively minimize the Euclidean distance (*error step*) between corresponding pairs of correspondences p_i, q_j (*matching step*) aiming at overlapping the point clouds as well as possible. But not all correspondences are necessarily good, and thus, a correspondence rejection method is applied to increase convergence. Holtz et al. provide a detailed description of such methods [4].

Further improvements to the original ICP include initial down-sampling of points to speed up the process (*point selection step*) and the application of a coarse registration as the initial alignment assumed by fine techniques.

B. *ICP Variants*

Although several variants of the original ICP algorithm have been proposed, two of the most widely used include point-to-point and point-to-surface.

ICP Point to Point (ICP-PP) defines its error metric as the sum of the Euclidean distances between pairs of corresponding points. The error function has a closed-form solution that can be found by means of the Singular Value Decomposition (SVD) method [9].

ICP Point to Surface (ICP-PS), proposed by Chen and Medioni [10], utilizes a point-to-plane error metric. As opposed to ICP-PP, no closed-form solution is available. The Levenberg-Marquardt method may be used to solve the problem.

The fundamental difference between ICP-PP and ICP-PS stems from the definition of their error metrics and, consequently, their minimization methods. Experimentally, their implementations provide different execution times and convergence rates, as discussed in Section IV.

III. A Practical 3D Point Cloud Scanning Setup

In order to evaluate the algorithm variants, a complete 360° 3D scanning-through-stitching system was implemented. Figure 1 illustrates the system organization. By changing the ICP algorithm, the system performance could be evaluated. Before delving into the evaluation details, we describe how the system was integrated.

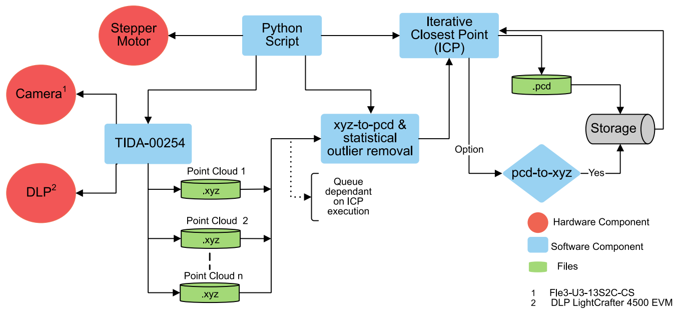

Fig. 1. 3D point cloud scanning and stitching system setup.

The system had three main components: (1) point cloud generation, (2) point cloud registration, and (3) system control. These components are explained below.

A. *Point Cloud Generation*

The generation of point clouds was handled by means of the Texas Instruments (TI) 3D Machine Vision Reference Design (TIDA-00254), which provides a three-dimensional photography system that employs structured light to support 3D scanning.

The TIDA-00254 reference design is fast and reliable. It is available as a 3D scanner solution as part of TI's Digital Light Processor (DLP) portfolio [11]. Our implementation was integrated using a Point Grey Flea3 digital camera, a TI's

LightCrafter 4500 evaluation module pattern programmable projector, and the DLP Structured Light software development kit (SDK). The TIDA includes a user's guide illustrating practical system implementations from calibration to point cloud generation [12].

Despite the advantage of having the TIDA-00254 and integrating the basic 3D scanning system, this was not enough. As any 3D scanner of its type, this assembly could only generate individual 3D point clouds of specific objects' poses, due to the limited field of view of a single optical sensor. In order to see the entire object surface, a means for changing the sensor's object view was needed. In our implementation we chose mounting the object on a rotating stage, allowing for exposing all its views in 360°, to the sensor.

A rotating stage provided the advantages of maintaining the same rotational axis for all solid model views, allowing for a controllable angular position, and enabling the solution to be integrated as a package. Such a stage was implemented through a 200-step stepper motor controlled by an MSP430G2553 microcontroller, interfaced via a USB 2.0 port to the host computer. This solution enabled precise rotation with a resolution of 1.8° per step, synchronized and controlled from host. These attributes contributed to producing a precise and streamlined 360° 3D scanning solution.

B. *Point Cloud Registration*

The usage of a rotating stage also facilitated the selection of the registration algorithm, as each rigid transformation between scans was known. Therefore, as explained in Section II, an ICP algorithm providing rigid registration techniques was utilized to provide for the point cloud registration.

C. *System Control*

With all components identified, it was realized that multiple software, data, and hardware modules needed to be integrated into a minimally supervised solution. To this end, a Python script was developed, able to work as "glue-logic", integrating and synchronizing the proper execution of all components. Figure 1 illustrates the organization of this code.

D. *Practical Considerations*

Several practical challenges needed to be overcome to achieve proper system performance. One major problem was controlling the amount of glare in the captured images to be able to optically calibrate the system. The usage of non-reflective materials to cover the camera's field of view proved to be effective for such a purpose. Another problem was the relative point of view of the camera with respect to the projector to capture the calibration pattern. Using a fixed-frame mount or a camera-behind-projector set-up helped in the process. Figure 2 shows the particular setup used in this implementation. After the first few clouds were captured, the presence of outliers in the image became a problem for the stitching process. The application of an outlier removal algorithm prior cloud processing helped reducing the number of bad-correspondences during registration. Lastly, but not less important, keeping the amount of overlap between successive captures between 10% and 15% proved a good measure to support the registration process.

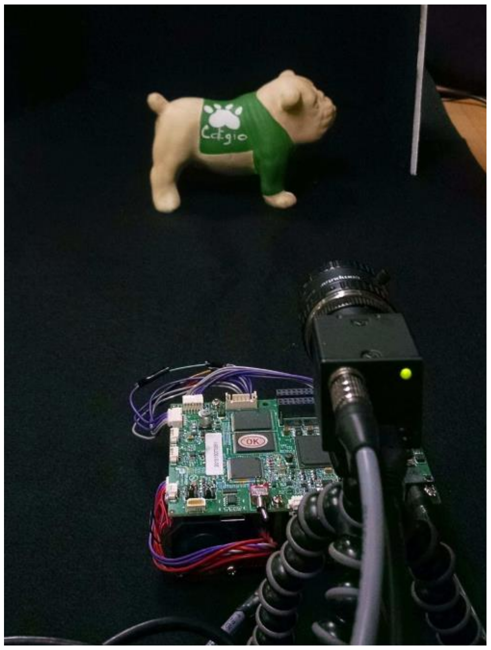
Fig. 2. Camera-behind-projector setup with non-reflective black cloth used for improving calibration process.

IV. Experimental Design, Results, and Analysis

To validate the system functionality, we started out by generating 3D point clouds with the TIDA-00254. After the successful generation of multiple point clouds, the main data set was generated. It consisted of 35 point clouds representing different poses of a figurine of our campus' mascot (Figures 3 and 4). Then, two main steps were carried out: algorithm selection and algorithm analysis.

A. *Algorithm Selection and Setup*

The clouds were registered by means of an ICP algorithm implemented using PCL. PCL is a C++-coded, BSD[1]-licensed, open-source, software library that supports n-dimensional point clouds processing and visualization [3], [4]. PCL provides multiple implementations of ICP, including non-linear, generalized, joint, point to surface, point to plane, among others. Each implementation requires the user to provide multiple parameters for the process to work properly. Evaluation of the ICP-PP and ICP-PS variants was important because although they both provide the best tradeoff between accuracy and speed [7], it was not clear how they performed with respect to each other.

Its implementation was done on a Windows-based PC with a *2.20 GHz Intel i5-5200U Core* CPU. The fixed parameters for both ICP methods were $\varepsilon=1e^{-8}$ and 40 iterations. It can be shown that the registration error function

[1] BSD is the Berkeley Standard Distribution, providing open software through the Open Source Initiative (OSI) http://opensource.org

had a minimum at 40 iterations [5]. Moreover, the PCL *StatisticalOutlierRemoval* class was used with a 2.5 threshold and a mean *k* of 50.

Fig. 3. UPRM mascot figurine utilized to generate 3D point clouds.

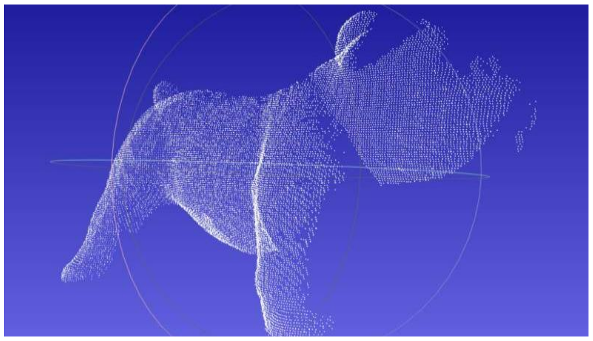
Fig. 4. Single 17,859-point, 3D point cloud of mascot figurine.

The parameters evaluated included: Maximum Correspondence Distance (*MCD*), Fitness Score (FS), *k*-neighborhood (*k*), and down-sample Leaf Size (*LS*). The *MCD* is defined as the maximum distance at which a correspondence between two clouds is considered. The fitness score is the sum of the squared distances between source and target clouds. Correspondences were determined by means of a Nearest Neighbor Search which, in turn, was specified by the *k* parameter. Finally, the *VoxelGrid* class of PCL, filtered and down-sampled the clouds by approximating each set of points contained in a 3D voxel grid with their centroid [3]. So, the leaf size determined the filter size.

B. *Algorithm Analysis*

The performance of both ICP-PP and ICP-PS methods, in terms of execution time (ET) and FS was evaluated against *MCD*, *LS*, and *k* parameters. The algorithms were first fed with the Stanford Bunny point cloud data set to assess their functionality [13]. Then, the UPRM mascot data set was used throughout the experiments.

In the first evaluation, the results for the *MCD* showed that ICP-PS outperformed ICP-PP in both evaluation criteria (see Figure 5). However, the differences in ET and FS were about 1 minute and 0.2mm, respectively, for *MCD > 35cm*. Moreover, the tendency was towards a decrease in *FS* as *MCD* increased. The second evaluation revealed an inversely proportional relation between execution time (ET) and filter size (see Figure 6). The fitness score (*FS)* showed a rapid growth for filter sizes above 0.08. In the third evaluation, results for the *k* parameter showed that ICP-PS outperformed ICP-PP for most of the tests (see Figure 7). The *FS* curves showed a dome-like shape, so we had minima at *k=3* (minimum feasible *k* value) and *k=100* (maximum evaluated). ET exhibited its largest evaluated value for the latter case.

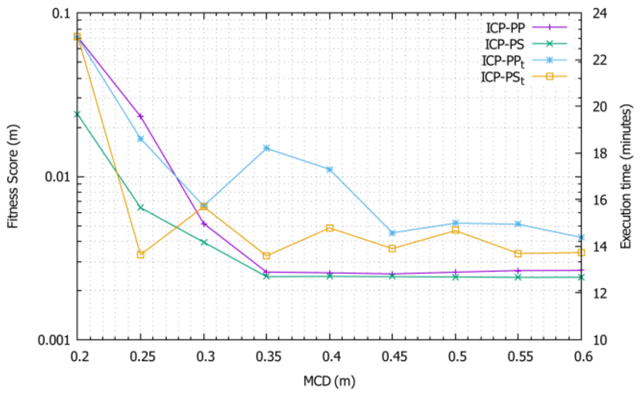

Fig. 5. Fitness score (FS) and Execution time (ET) as a function of MCD values for ICP-PP and ICP-PS methods.

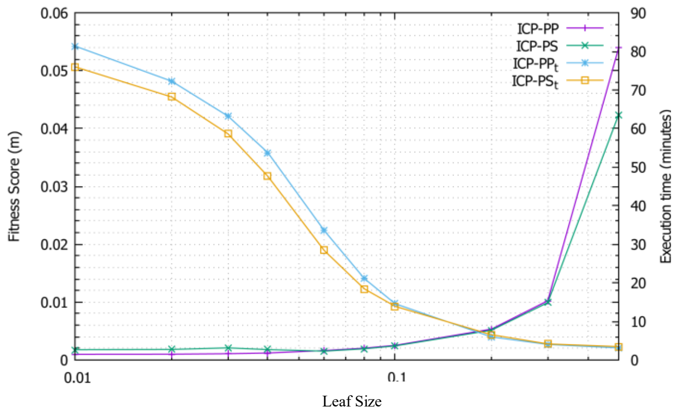

Fig. 6. FS and ET as a function of LS for ICP-PP and ICP-PS methods.

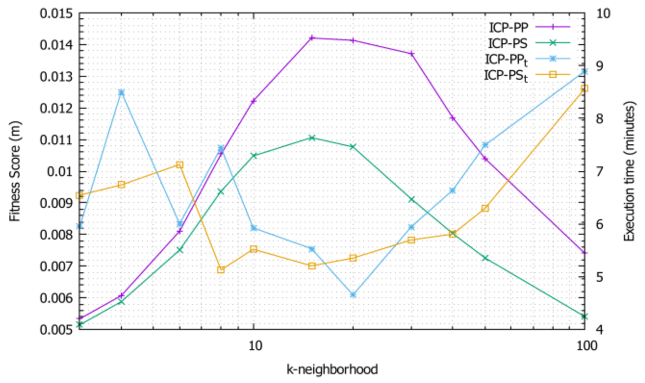

Fig. 7. FS and ET as a function of k values for ICP-PP and ICP-PS methods.

C. *Lessons Learned and Recommendations*

The implementation approach and ensuing evaluation led to interesting findings. First, the proposed setup proved to be effective in generating high-quality 3D point clouds registration (see Figure 8). To the best of our knowledge there is no existing literature that used FS as accuracy metric, however the results herein provide substantial evidence to validate the metric. Second, the performance analysis showed that, in general, ICP-PS performs better than ICP-PP in terms of computing speed and accuracy. The computing speed and accuracy were characterized by means of ET and FS, respectively. It was shown that the computing speed decreased when only a subset of the clouds was used for computation. This result aligns well with previous theoretical results predicting that sampling the points accelerates the registration process [4]. On the other hand, it was also evidenced that both algorithms failed for *LS* values greater than or equal to 0.5, for *k* values smaller than 3, and for *MCD* values smaller than 30 cm. Lastly, the results presented here have the potential of decreasing the implementation time for PCL-based applications or, in general, for similar applications, by reducing uncertainty in the decision-making process.

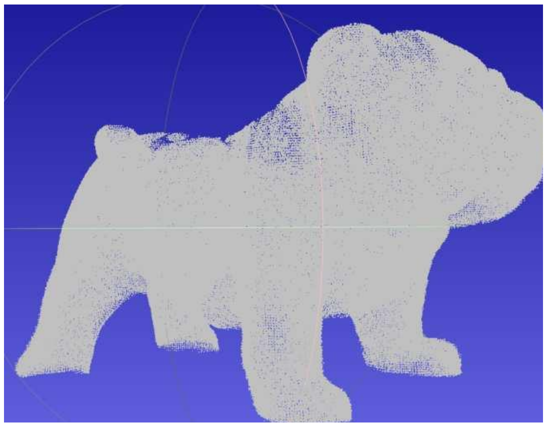
Fig. 8. 3D 360-degree view of Tarzan generated from the registration of 35 separately-scanned point clouds. The resulting cloud contains 526,842 points.

V. CONCLUSION

This paper presented a step-by-step process to implement a functional 3D point cloud scanning system able to produce 360° images of a solid model constructed by stitching individual partial scans of it. Performance analyses carried on two ICP variants: point-to-point and point-to-surface, allowed for identifying the algorithms and factors leading to the best accuracy and computing speed in the system implementation. It was found that ICP-PS performed better than ICP-PP for the evaluated criteria, making the former the algorithm of choice for this application. The system implementation took advantage of multiple readily available technologies: a TI DLP and camera subsystem enabled the point cloud generation, an open-source PCL facilitated the point cloud registration algorithms, an MSP430 microcontroller enabled the rotating stage, and the python scripting language enabled the development of the necessary programming to perform the system integration and process automation. The general process followed in this implementation provides valuable guidelines for implementing similar applications while the analyses and evaluations provide objective criteria for quickly making time consuming design decisions. The validity of the proposed process was assessed through the stitching of a solid model composed of 35 individual point clouds, resulting in a single representation of 526,842 points.

ACKNOWLEDGMENT

The authors would like to thank Gretchen Quiles, Luis Francisco, and Roberto Guzman for their help in the project. We also thank Matt Soucek, Clinton Rodgers, Eric Droge, and Trevor Dowd for their guidance. This work was partially supported by the Texas Instruments, Inc. under the UPRM-TI Collaborative Program. The views presented in this paper are the opinions of the authors and do not represent any official position of the sponsoring organizations.

REFERENCES

[1] J. Pages, J. Salvi, R. Garcia, and C. Matabosch, "Overview of coded light projection techniques for automatic 3D profiling," in 2003 *IEEE Int. Conf. on Robotics and Automation*, vol. 1, Sept 2003, pp. 133–138.

[2] J. Salvi, J. Pages, and J. Battle, "Pattern codification strategies in structured light systems," *Pattern Recognition*, vol. 37, pp. 827–849, 2004.

[3] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in 2011 *IEEE International Conference on Robotics and Automation*, May 2011, pp. 1–4.

[4] D. Holz, A. Ichim, F. Tombari, R. Rusu, and S. Behnke, "Registration with the point cloud library: A modular framework for aligning in 3-D," *IEEE Robotics Automation Mag*., vol. 22, no. 4, pp. 110–124, Dec 2015.

[5] B. Bellekens, V. Spruyt, R. Berkvens, R. Penne, and M. Weyn, "A benchmark survey of rigid 3D point cloud registration algorithm," *Int. J. Adv. Intell. Syst*, vol. 8, pp. 118–127, 2015.

[6] J. Salvi, C. Matabosch, D. Fofi, and J. Forest, "A review of recent range image registration methods with accuracy evaluation," *Image and Vision Computing*, vol. 25, no. 5, pp. 578 – 596, May 2007.

[7] M. Attia, Y. Slama, and M. A. Kamoun, "On performance evaluation of registration algorithms for 3D point clouds," in 2016 13th Int. Conf. on *Computer Graphics, Imaging and Visualization (CGiV)*, March 2016, pp. 45–50.

[8] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *Proc. Third Int. Conf. on 3-D Digital Imaging and Modeling*, 2001, pp. 145–152.

[9] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," IEEE *Trans. on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, Feb 1992.

[10] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image and Vision Computing*, vol. 10, no. 3, pp. 145 – 155, 1992.

[11] Accurate point cloud generation for 3d machine vision applications using dlp technology. [Online]. Available: www.ti.com/tool/tida-00254

[12] Tida-00254: Accurate point cloud generation for 3d machine vision applications using dlp technology and industrial camera. [Online]. Available: www.ti.com/lit/ug/dlpu019a/dlpu019a.pdf

[13] The Stanford 3D Scanning Repository. [Online]. Available: graphics.stanford.edu/data/3Dscanrep/