A Comparison Study of Five 3D Modeling Systems Based on the SfM Principles*

Yuan-Fang Wang, Ph.D.
Visualsize Inc.
Goleta, CA 93117, USA
yfwang@visualsize.com

September 8, 2011

Abstract

We present a comparison study of five 3D modeling systems based on the structure-from-motion principles (Bundler, Bundler+PMVS2, Project Photofly from Autodesk, ARC 3D Web Service, and our own). To ensure that the comparison is fair, we have included only those 3D modeling systems that are available for use on the Web or locally in a binary format, and comprise a complete, fully-automated 3D pipeline that leads from input images to 3D models, without any user intervention, and without data-dependent parameter tuning. In addition to ground-truthed 3D data, we have used a testbed comprising over 100 data sets, with over three thousand images, representing a variety of 3D scenes, collected from a large number of consumer-market digital cameras and camera phones of many makes/models, and all without prior camera calibration, use of special equipment (tripod, lens, etc.) and lighting (laser and structured light projection), and user training in image acquisition. In the paper, we introduce the methodology of the comparison, justify the crucial choices made in the study, present the results, and provide an analysis of these results.

1 Introduction

In this report, we present a comparison study of five 3D modeling systems (Bundler, Bundler+PMVS2, Project Photofly from Autodesk, ARC 3D Web Service, and our own) based on the structure-from-motion principles [18, 14]. The usage scenario we try to emulate in this study is that of a commercial 3D modeling system that accepts 3D modeling requests from clients (Apple iphone, Android phone, PC, etc.) over the Web, executes the 3D modeling pipeline on a back-end server, and returns the 3D model as a result. The users (1) are not computer vision experts and cannot provide additional information other than the photos

*The manuscript is about 3D modeling. We have included many figures that depict 3D models constructed by a variety of modeling systems in both discrete-point and textured-surface format. However, in order to appreciate the quality of a 3D model, there is no substitution for examining such a model in 3D (instead of as a few 2D screen shots) . Hence, we strongly urge interested readers to visit the Web site http://www.visualsize.com and browse the many demo and comparison results there.

themselves, (2) are not willing to go through lengthy training, or purchase expensive cameras or specialized photography equipment for building 3D models, (3) may be cost conscientious especially when connecting to the back-end server through a mobile device where the user may have to pay for the bandwidth usage (and hence, no uploading large photos that tie up Web links for a long time), and (4) worst still, are accustomed to the "instant gratification" Web experience, and hence, are impatient to get the results back.

While similar performance comparison has been attempted before [36, 38, 41], our study stands out by performing "rubber-meets-the-road" validation tests that closely mimic what a commercial 3D modeling system needs to accomplish in the real world. The novelty of our comparison study is thus three-fold:

(1) The comparison was performed by exercising the full 3D modeling pipelines, from input images all the way to 3D models, instead of testing some isolated components in a 3D pipeline [38],

(2) In addition to the ground-truthed 3D data provided by [41], we have used over 100 data sets (122 to be exact at the time of the submission of this manuscript), with over three thousand images, representing a variety of 3D scenes, collected from a large number of consumer-market digital cameras and camera phones of many makes/models, and all without prior camera calibration, use of special equipment (tripod, lens, etc.) and lighting (laser and structured light projection), and user training in image acquisition (in contrast, [36, 38, 41] have used small, calibrated data sets), and

(3) To ensure that the comparison is fair and the results do not depend on the details of implementation, we have included only those 3D modeling systems that are available for use on the Web or locally in a binary format; comprise a complete, fully-automated 3D pipeline that leads from input images to 3D models, without any user intervention, and without data-dependent parameter tuning; and are able to perform the feats using images of a reasonable size. Furthermore, a diligent Web search has unearthed no other 3D modeling system that fits the comparison requirements, and hence, our selection is believed to be comprehensive and provides a holistic view of the state of the art.

A 3D computer model can have many applications in both the civilian and military sectors. While one can generate 3D models using active sensing and structured lighting, a more economical way is to build 3D models using the images from billions of camera phones and digital cameras in circulation today. The general principle of such a 3D modeling enterprise is well established, and is alternately called structure-from-motion (SfM) in the computer vision and computer graphics communities or simultaneous localization

and mapping (SLAM) in the robotics community [18, 14, 10, 21].

Regardless of the nomenclature, the general principles of such a system are to exploit the locations and correspondences of image features (points, corners, lines, or other high-level features) in multiple images to infer the 3D feature locations and the camera poses. However, it is highly non-trivial to develop a robust, efficient, and accurate 3D modeling system because going back from 2D images to 3D models is an ill-posed problem [14], whose solution can be numerically unstable and sensitive to noise and outliers in the input data. The level of difficulty can best be appreciated by observing that while textbooks have been written on this subject almost 20 years ago [14], only in July of 2010 and after 12 years of research, development, and acquisition, was first such commercial system (Project Photofly from Autodesk) announced [4].

2 Comparison Methodology

We describe here some critical choices made in the study:

2.1 Selection of 3D Modeling Systems

(1) A 3D modeling system represents highly sophisticated software artifacts with many subtle details (e.g., our system involves a sequence of over 20 programs that perform a variety of functions from feature detection and matching, to structure and motion computation, and to texture mapping). Hence, it is not practical to "reverse engineer" such software by studying the descriptions of such a system in published papers and then re-implement the ideas from scratch. Without direct access to the sources or the binaries, the comparison opens itself to attack that the implementation is plainly wrong, or the parameters are not tuned properly.

(2) A 3D modeling system must be "complete" in the sense that it must be able to lead directly to 3D models from input images. It is not our goal to compare isolated components of such a 3D pipeline. For example, the comparison study reported in [38] has focused on multi-view stereo matching and texture mapping, but assumes that ground-truth intrinsic and extrinsic camera parameters are available. In the real world, such ground-truth data are not available, and hence, the comparison results on isolated processing components do not properly reflect the performance of a whole pipeline on real-world data.

(3) If a choice exists between the source and binary releases of a 3D modeling system, we prefer using the binary release to avoid the possibility that mistakes might be made in the compilation to cause a 3D pipeline

to perform below its capabilities.

(4) The 3D modeling program should ideally have no end-user tunable parameters. If there are any such parameters, there must be clear instructions on how these parameters should be set for different image collections. This requirement is to avoid the criticism that a 3D program produces erroneous results because of wrong parameter setting.

Based on the above requirements, we have selected five 3D modeling systems for comparison:

(1) Bundler [30] — which is the core of the Photo Tourism line of research [39, 40, 2, 15] at the University of Washington (Drs. Snavely and Seitz) and Microsoft (Dr. Szeliski), arguably one of the best known R&D projects in 3D modeling. The latest v0.4 binary release (released on April 10, 2010) was used.

(2) Bundler and PMVS2 combo [30, 17] — PMVS2 [17, 16] was developed by Drs. Furukawa and Ponce at the University of Illinois. PMVS2 is a multi-view stereo software that takes a set of images and camera parameters to reconstruct the 3D structure of an object or a scene visible in the images. As PMVS2 does not perform the SfM computation, its primary purpose is to serve as a "post-processing" step to increase the 3D point-cloud density of an SfM engine, and in our comparison, Bundler. We have used the latest release (updated on July 13, 2010) [17].

(3) Project Photofly from Autodesk [4] — Project Photofly is based on the technologies of RealViz, acquired by Autodesk in 2008. RealViz was founded in 1998 with technologies acquired from INRIA's RobotVis Research Group, headed by Dr. Faugeras.

(4) ARC 3D Web Service [42, 3]—A Web-based 3D modeling service of Drs. Vergauwen and Van Gool of KU Leuven.

(5) Our own 3D modeling program [23, 5, 7, 6]. We mention here that our system is based on the same SfM principles as all others. We perform feature detection and matching, then use such feature correspondences in a non-linear optimization computation to recover discrete 3D coordinates and the camera's intrinsic and extrinsic parameters, and finally, we perform texture mapping on the 3D point clouds to obtain the 3D surface description. Some sample results of our system are shown in Fig. 1. Short movies of the 3D models of the 122 test data sets are available for viewing on the Web at: http://www.visualsize.com/3ddemo/index.php.

The situation is a lot like Web search: Google, Yahoo, Microsoft and many others are all doing it. The

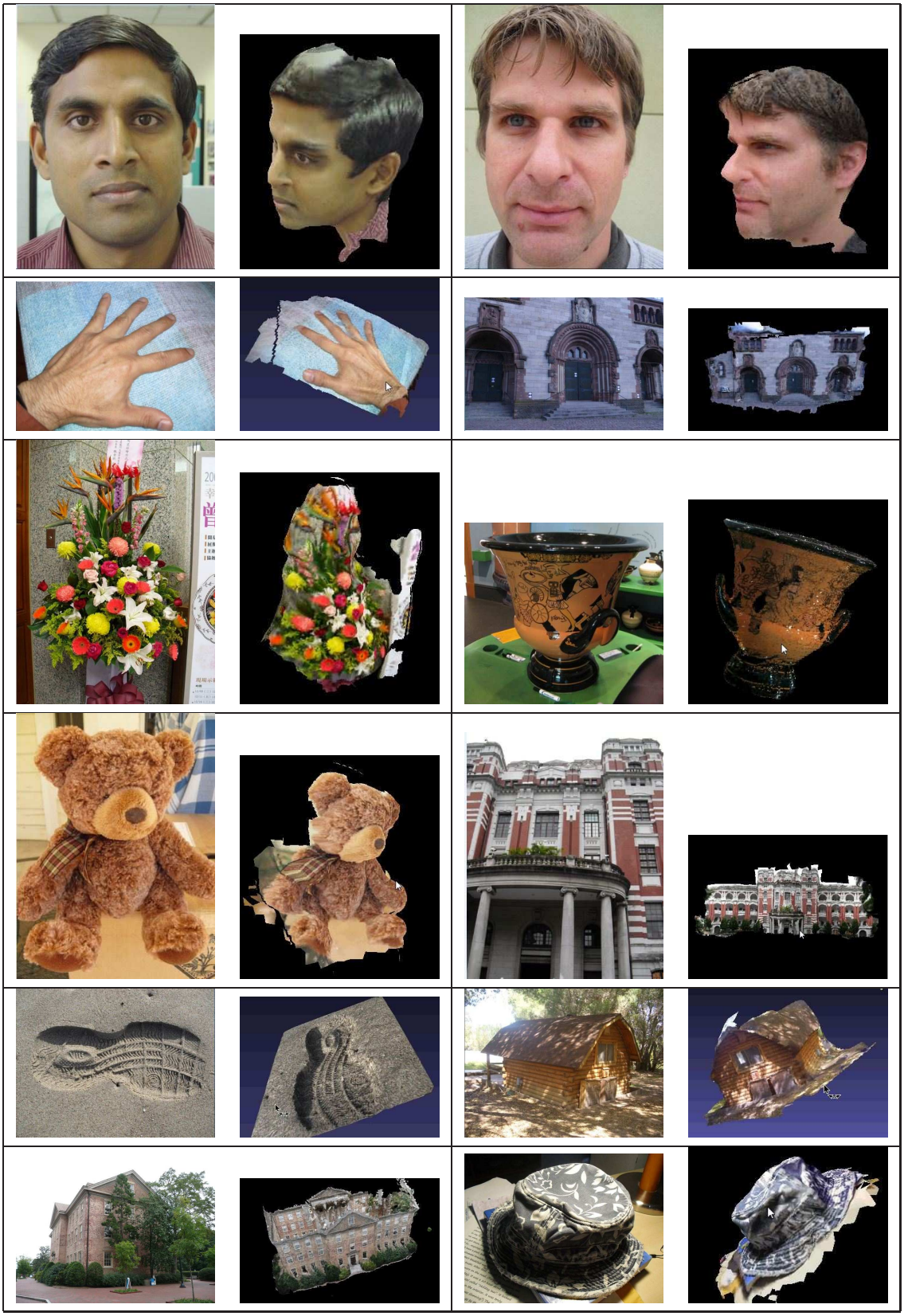


Figure 1: Sample results of our 3D modeling system. Each data set is represented by one input image (left) and one image of the 3D model (right). Short movies of the 3D models of the 122 test data sets are available for viewing on the Web at: http://www.visualsize.com/3ddemo/index.php.

September, 2011

underlying principles of crawling the Web, building index structures, calculating page ranks by popularity and cross references, etc. are the same. But submitting the same query to different search engines, you are liable to get different results back. So "the devil is in the details." The same is true with SfM. The general principle is beautifully exhibited in books like [18, 34, 43] and a careful perusal of many relevant publications [39, 40, 2, 15, 42, 3, 17, 24, 16, 32, 33, 41] reveals more similarity than difference. Some discussions on the accuracy of our modeling system and why our system performs much better than others are presented in Sec. 4.

Finally, a diligent Web search has unearthed no other 3D modeling system that fits the comparison requirements, and hence, our selection is believed to be quite comprehensive.[1]

2.2 Selection of Testbed Image Data

We have used as our testbed real-world image data gathered from consumer-market digital cameras and camera phones of a variety of makes/models, with many of these images contributed from anonymous users using cameras of an unknown origin. We have used over 100 data sets with over three thousand images. These data sets comprise face/non-face sequences, soft/hard objects, shining/dull surfaces, complete/partial descriptions, indoor/outdoor collections, and short (as few as 5 images)/long (as many as 130 images) sequences. These data sets were collected to be representative of a significant spectrum of possible 3D modeling applications (see Fig. 1 for sample results and our Website http://www.visualsize.com/3ddemo/index.php for all 122 examples).

The most important reason for the testbed choice is practicality. Approximately one billion camera phones and another 100 million digital cameras are sold worldwide each year. The wide availability means that everyone with a digital camera or a camera phone can be a content producer. Furthermore, as these cameras use a wide variety of lenses and CCD arrays in construction, it makes the comparison realistic and mimicking what one would expect running a 3D modeling system in the real world.

Furthermore, images with ground-truth 3D profiles are very expensive to gather. [38] uses the Stanford Spherical Gantry to acquire images with ground-truth camera motion information, but the experimental

[1] A commercial system, TopoMap from 2d3 [1], is supposed to use the SfM principle for reconstructing 3D topological maps from images alone. However, repeated inquires to 2d3 on including TopoMap in our comparison study did not solicit a response. Insight3D [26] is an open-source image-based 3D modeling software that is purported to automatically recover the camera's optical parameters and poses, along with a 3D point cloud of the scene. However, we were not able to get the software to run properly on our data sets and, again, repeated inquiries to the distributor of the software archive did not solicit a response.

setup is most suitable for small 3D objects and only two data sets (Dino and Temple) were provided by [38]. Middlebury Stereo Datasets [9, 36] comprise only short sequences (up to 7 images) using a fixed linear camera translation for all of them. These sequences are therefore more suitable for validating short-baseline stereo algorithms, but are inappropriate for algorithms designed for wide-baseline SfM computation. [41] uses LIDAR to acquire 3D data with the ground truth. It is an expensive proposition and only two such ground-truthed data sets (Fountain and Herz-Jesu) were made publicly available. While we do use many of these data sets in our comparison experiments and accuracy studies (Figs. 1, 4, 5, 6, 7 and 8), we augment them with significantly more real-world data sets as described above.

Using real-world image data from consumer-market digital cameras/phones for comparison does raise one serious issue, that is, no ground truth for the camera's intrinsic and extrinsic parameters and for the 3D object structures is available. What we did was to compare only the recovered 3D structural traits, or more specifically, the density and quality of the 3D point clouds—which all five 3D modeling systems produce as an intermediate step toward the final texture-mapped models. While the cloud density can be easily quantified by counting the number of 3D points in a model, quality of a recovered 3D model can only be judged by eye-balling the graphic display of that model—as no ground-truth data are available. While judging quality by eye-balling may appear *ad hoc*, such a qualitative evaluation is meaningful because

(1) Human eyes are actually very adept at judging, qualitatively, the correctness and proportionality of a 3D structure, and an abundance of research from psychology [37] indicates that humans are experts at the task of 3D visual reconstruction and evaluation. Furthermore, in Sec. 4, we do provide a more rigorous analysis of the accuracy of our 3D modeling system using the two ground-truthed 3D data sets provided by [41].

(2) While one might argue that the 3D cloud density can be arbitrarily increased to make a 3D model appear "dense," and hence, tilt the scale in one's favor, such an argument ignores an elementary tenet in pattern recognition. Increasing the 3D cloud density necessitates the inclusion of more 2D features in the SfM analysis. However, given the same input images, the amount of "information" therein is fixed. Hence, including more 2D features invariably decreases the feature quality. This density vs. quality trade-off is known as precision vs. recall in the database community and false-positive vs. false-negative in the pattern-recognition community [13, 20]. That is, using more features runs the risks of including weak, ambiguous features that cannot be matched reliably, and hence, introduces outliers into the computation and degrades the robustness. All 3D modeling pipelines must therefore deliberately weigh the choice between 3D cloud

density and quality. It is logically flawed to argue that one can increase the 3D cloud density blindly and somehow be immune to this fundamental limitation of precision vs. recall.

(3) A more formal way to judge the quality of a reconstructed 3D model in the absence of the ground truth is to compute the re-projection error [18]. Our algorithm produces an average re-projection error (per feature for all images and all features) about 0.5 pixel (based on an input image size of 640×480) for the data sets shown in this paper. Unfortunately, such re-projection error information is not available for other 3D programs, and hence, only qualitative comparison is possible. Furthermore, while one might argue that it is possible that, for certain degenerate 3D configurations, multiple 3D feature positions and camera poses may produce re-projected 2D features that match well with the image observations, in reality we believe that such coincidence is extremely rarely—especially when a large number of images and tens and thousands of features are involved.

2.3 Testing and Reporting

To ensure fairness in the comparison:

(1) When program run times are compared, the programs are executed on the same platform under the same condition (the same number of CPU cores used with the same amount of memory, and without any hardware, e.g, GPU, acceleration) so as not to give an unfair advantage to some[2]. Do note that Project Photofly and ARC 3D Web Service provide only a client GUI to upload input images onto their cloud computing server facilities for processing. The exact runtime is unknown and is not compared.

(2) We try to perform an "apples-to-apples" comparison. Hence, the factors used to judge the results must be the common denominators of all these 3D modeling systems. Bundler and Bundler+PMVS2 combo do not generate texture-mapped models. Even when texture-mapped models are generated (by Project Photofly and our system), the systems may use very different algorithms. Fig. 2 shows that Project Photofly uses a local, view-dependent splatting technique [22] for texture mapping. With a zero splat size, only discrete 3D point clouds are displayed (left). As the splat size increases, progressively larger color patches are generated around discrete 3D points to fill in the void (right). However, such a scheme will fail if the 3D cloud is not dense enough to start with. As all such 3D modeling systems generate 3D point clouds as an intermediate

[2]We observed that Bundler and our system were set up to use a single CPU core in the computation. PMVS2 actually exploited all CPU resources, and hence, PMVS2 could potentially run twice as fast on the dual-core machines we used in our comparison study. However, we did not change the default execution script of PMVS2 for fear of breaking the codes.

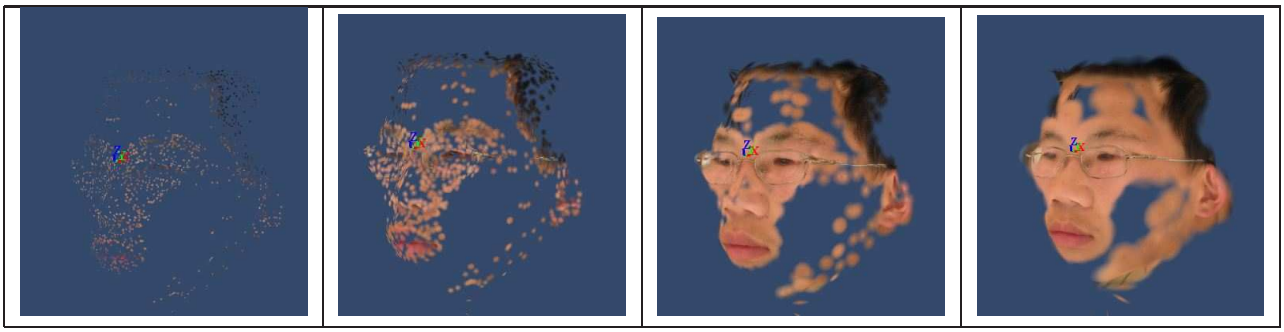


Figure 2: Auodesk's Project Photofly uses a local splatting technique for texture mapping. As the size of the splat increases from left to right, more 3D surface is generated.

result, and these results strongly influence the final texture-mapped models, we have chosen to compare the density and quality of the 3D point clouds from the SfM computation.

(3) In our comparison study, a single control script is used for all data sets, with no per-data-set tuning allowed. This is true for Bundler and Bundler+PMVS2 combo (the default scripts distributed with the binary releases of Bundler and PMVS2 were used without modification), Project Photofly and ARC 3D Web Service (no end-user tunable parameters are exposed on the client-side GUI), and our system. This is to ensure that no accidental parameter tuning error could bias the results.

(5) One important choice in the comparison study is the input image resolution, and we have down sampled input images to roughly the VGA size (640×480) for all testing data sets reported here. As mentioned before, the goal of the comparison is to emulate, as closely as possible, what a commercial 3D modeling system needs to accomplish in the real world. The most likely scenario, we believe, is that such a system will use a Software-as-a-Service (SaaS) model simiar to that of Autodesk Photofly and ARC3D Web Service. The main reason is that the 3D modeling enterprise often involves lengthy computation, and, using even a modest number of, say, 30 images, such a computation can easily tie up a computer for hours. Hence it is not feasible to run such computation and resource intensive tasks on client computers with varying capabilities.

However, the bandwidth requirement for transmitting high-resolution image data from a client (a PC or a mobile device) to a Web server can be costly—this is especially true for a mobile device communicating through a wireless network and the user may have to pay for the bandwidth usage. For example, the two data sets provided by [41], at the original resolution of 3072x2048, total 151MB for Herz-Jesu-P25 and 61MB for Fountain-P11. Slow, unreliable uploads are particularly problematic for modeling tasks using a large number of images, e.g., the UNC data set [44] shown in Fig. 4 with 128 3072x2048 images totals 222MB

even with space-efficient JPEG encoding. It is extremely unlikely that an end user will be willing, or able, to upload such large amounts of data onto a Web server. In fact, our first-hand experience with ARC3D Web Service has been that it is next to impossible to maintain the upload link long enough to upload large, high-resolution data sets onto their server. And the few times the upload was successful, the server failed to start properly or generated any results.

Furthermore, we have observed that arbitrarily increasing images' spatial resolution does not improve the quality of the resulting 3D models. Quite the contrary, small details exhibited only in high-resolution images are not stable. Hence, including such evanescent features can actually deteriorate the performance by introducing outliers. This same sentiment that larger-scale features are more reliable and much more efficient to process is also shared by David Lowe [25]. In fact, the demo version of SIFT used in Bundler will not run if the image resolution is higher than 1800 in any dimension. However, using an image resolution lower than VGA often gives erratic results (true for all five systems). Our experience has been that VGA seems to be a good compromise between quality and speed. We provide some concrete accuracy analysis data in Sec. 4 to support this claim.

(4) We try to present as much data as possible. However, not all systems expose all quality measurements (e.g., the re-projection error). Timing and density measurements are available only if we can run a 3D modeling pipeline locally and if the outputs are in some public-domain format that is easily deciphered. The above is true for Bundler and Bundler+PMVS2 combo, partially true for ARC 3D Web Service (no runtime comparison but output in the public-domain OpenInventor format), and not true for Project Photofly (no runtime comparison and output in Autodesk's proprietary format). Hence, we perform mostly visual comparison in the case of Project Photofly.

3 Procedures and Results

The test platform is a PC with a 2.8Ghz Intel Core 2 Duo CPU, 4G RAM, running Windows 7. The experimental procedures are extremely straightforward: As all these 3D modeling systems need are input images, all we did was to provide them with the input images and then wait for the computation to finish. We feel that the simplicity of the procedures better ensures fairness. We have used the binary releases of these programs so we could not have compiled them incorrectly, we have used the default execution scripts supplied with the releases without modification so we could not have tuned the parameters wrongly, and we

have run these programs on the same machine using the same number of CPU cores, the same amount of memory, and with no hardware acceleration. For Project Photofly and ARC 3D Web Service, the images were uploaded to their servers on the Web and there is no end-user tunable parameters on their GUI.

Fig. 4 presents the results of running the five modeling systems on ten sample data sets. For all data sets, the two images at top left are sample input images, the two at top right are our results, the two at middle left are results of Bundler, the two at middle right are results of Bundler + PMVS2, the two at bottom left are results of ARC 3D Web Service, and the two at bottom right are results of Autodesk's Project Photofly. For all data sets, the table below the graphic results shows the name and size of the data set, how many pictures are processed, how many 3D points are generated, and the runtime for Bundler, Bundler+PMVS2, ARC 3D Web Service, and our system. Note that as ARC 3D runs on their Web server, no time is recorded. No time and density information is available for Project Photofly as the modeling process is executed on Autodesk's cloud computing server and the results are in Autodesk's proprietary format.

Results of Bundler, Bundler+PMVS2, and ARC 3D Web Service are presented in discrete point-cloud format as these programs do not generate 3D texture-mapped models yet. We supplement texture-mapped results for Project Photofly and our system if one point-cloud picture is enough to illustrate the density and quality of such a discrete structure. Note that Bundler+PMVS2 display is Meshlab [27]. Photofly has its own client GUI for display. We have used our home-brewed display programs for the results from Bundler, ARC 3D Web Service, and our own system.

Of the ten data sets in Fig. 4, Lady, Flower and Soda Can were contributed from anonymous users, Building was used by PMVS2 [17], Fountain was from [36], Temple Ring was used in [38], and UNC was from [44]. The Lady data set is challenging because human faces are not strictly rigid. It takes about 30 seconds to take 10 to 20 pictures for constructing a head model. While the subject should be instructed not to move or talk, small movements due to breathing are unavoidable. Such movements violate the rigidity assumption used in the SfM and, as a result, all other systems failed for this data set except ours. (Photofly was able to obtain visually pleasing 3D point clouds, but the erroneous texture-mapped model reveals the deficiency in the cloud data.)

The Building data is also challenging because modern buildings often lack complicated, feature-rich facades common of old buildings (e.g,. churches and cathedrals). Again, all other systems failed except ours (Photofly constructed a "shrunk" 3D model which is structurally incorrect). It is unclear why Bundler

and Bundler+PMVS2 failed—when the data set is used as an example in the PMVS2 distribution. We surmise that different parameter settings might need be used in this case, and we were using the default settings of Bundler and PMVS2 from the binary distributions.

For structures with more complicated facades (Church and Fountain), all systems performed reasonably well and obtained qualitatively correct results. However, our systems produced much denser 3D clouds (2.7 times denser for Church and 3.4 times denser for Fountain than the best of the batch Bundler+PMVS2) and was able to analyze more images in these sequences with a small increase in runtime. The NTU Gate depicts a U-shaped structure of sharp curves. Furthermore, there were a few moving vehicles in some of the images. Bundler, Bundler+PMVS2, and ARC 3D failed to generate reasonable results on this data set.

Yet another data set that depicts buildings is the UNC data set. It is a very large data set with 128 images. These images represent a 360^o walk around of a building on the UNC campus. All systems did reasonably well because the building surface is highly textured. However, ARC 3D failed to return a result and we did not receive a response from ARC 3D explaining why this particular data set failed.

The Soda Can data set is difficult because of the specular and highly curved surface structures. All other programs were able to analyze and recover less than half of the surface structures while we were able to infer the 3D structures using all input images covering the whole 360^o.

The Temple Ring data were gathered using the Stanford Spherical Gantry and came with the ground-truth camera pose data [38]. However, none of these systems has a way to take such data in the computation. So to make the computation more realistic, we provided these systems only the images, but no ground-truth camera pose data. Again, while all these systems arrived at qualitatively correct 3D models, our cloud density is 4 times higher than the best.

The Flower data is interesting because of its highly irregular surfaces. When such surfaces are viewed close-by, the appearance of surface features can change drastically even with a small change in the viewpoint, which makes feature correspondences highly susceptible to errors. While Bundler did recover the general 3D profile, post-processing by PMVS2 deteriorates the 3D structures—probably because PMVS2, being a patch-based multi-view stereo system, prefers well defined surface patches, which are absent in this data set. This hypothesis is confirmed by noticing that the central flower region becomes empty while the planar 3D structure of the background wall is enhanced after PMVS2 processing. In fact, similar things seem to have

happened with the Lady data set too—PMVS2 enhanced the color towel in the background but filtered out 3D points in the face. In any case, our system obtained a 3D cloud that is 14 times denser than Bundler and Bundler+PMVS2 with a very small increase in runtime.

Finally, the Droopy data is very challenging. The surface of the stuffed animal is relatively homogeneous and devoid of features. We tried to compensate for the lack of features by increasing the number of images used (a total of 89 images). However, all systems did poorly for this particular data set, with reconstructed 3D structures suffering from a significant degree of missing data (on the back of the stuffed animal). ARC 3D failed to generate any result at all. This data set probably demarcates the boundary of what is and is not attainable for state-of-the-art, SfM-based, 3D modeling algorithms.

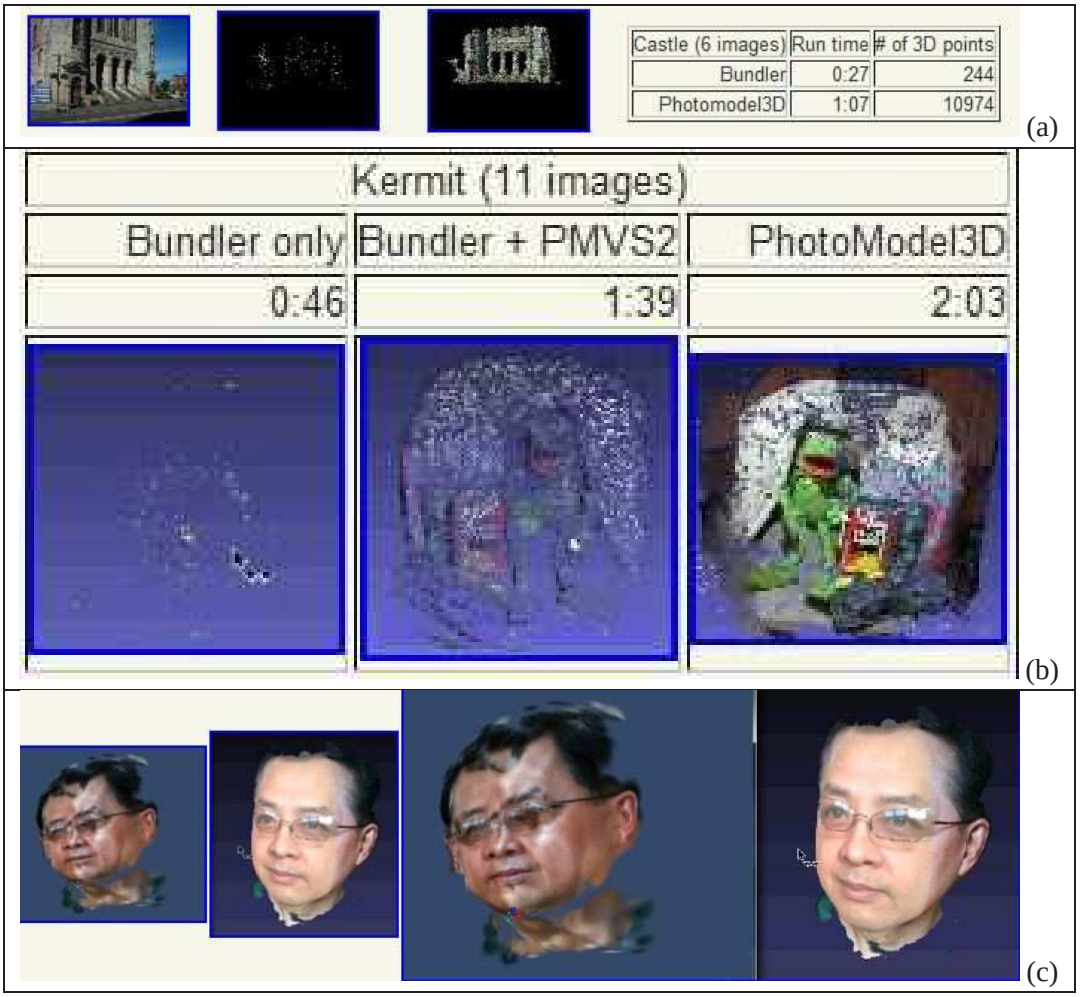


Figure 3: Pairwise comparison results. (a) Comparison results with Bundler: from left to right: a sample input image, a screen shot of Bundler's 3D model, a screen shot of our 3D model, and runtime and density statistics, (b) comparison results with Bundler+PMVS2, from left to right: Bundler only, Bundler+PMVS2, and our results, and (c) comparison results with Photofly: from left to right: Photofly's 3D model, our 3D model, and two models placed side-by-side.

Pairwise Comparison: We have also conducted many more pairwise comparisons that compared the performance of our system against Bundler (on 31 data sets), Bundler+PMVS2 (on 9 data sets), and Project Photofly (on 54 data sets) individually. One example is shown for each of the three cases (Bundler only, Bundler+PMVS2, and Photofly) in Fig. 3. These comparison results are summarized in our Web page
Bundler only: http://www.visualsize.com/3ddemo/true3d/comparison/index.html

Bundler + PMVS2: http://www.visualsize.com/3ddemo/true3d/comparison/index.html#_PMVS2

Photofly: http://www.visualsize.com/3Ddemo/comparison/index.html

As page limit does not allow us to show all these examples, and the quality and accuracy of a 3D model is best evaluated by viewing the model in 3D—instead of just a few screen shots, we strongly urge interested readers to browse our Web site for more information.

4 Analysis and Concluding Remarks

In terms of cloud density and quality, and the chance of success, the test data indicated that ours outperforms Project Photofly, which outperforms Bundler+PMVS2, which outperforms Bundler, and which outperforms ARC 3D. This observation also confirms our experience with the much larger, over 100 data set ensemble.

We attempt to answer two questions that are likely to be in a reader's mind: (1) How accurate is our 3D modeling system and (2) why our system seems to perform that much better that all others. Accuracy analysis requires comparison with the ground truth. However, as mentioned before, our test datasets were collected from a large number of consumer-market cameras and phones, and no ground-truth 3D profiles are available. The Dino and Temple data sets used in [38] were gathered using the Stanford Spherical Gantry, which provided the ground truth in the camera poses, but not in the 3D structures. Middlebury Stereo Datasets [9, 36] comprise only short sequences (up to 7 images) using a fixed linear camera translation, and hence, are not that interesting to us. To our best knowledge, [41] provides the only publicly available 3D data sets with ground-truthed 3D profiles that used a general camera motion, and were specifically generated to validate 3D modeling algorithms using the SfM principle. Ground-truth 3D profiles (gathered using a LIDAR system) for two data sets, Fountain-P11 and Herz-Jesu-P25, are used in our accuracy analysis. These data sets are available for download at: http://cvlab.epfl.ch/ strecha/multiview/denseMVS.html, and were used in the following academic paper: [41].

Evaluation Methodologies: We tried to emulate—as faithfully as possible—what a commercial 3D modeling system needs to accomplish for a client submitting 3D modeling tasks through a Web-service model. To this end:

• We down-sampled input images to the VGA size (640×480) for upload and processing (we do provide data processed at a higher 2150×1434 resolution for reference). While the original images are of a very high spatial resolution (3072×2048), it is unrealistic to expect that such high-resolution images will be available in real-world scenarios. Again, uploading such high-resolution images to a back-end 3D server is a very expensive proposition and is not feasible for most usage scenarios.

• We do not use any camera calibration data generated externally (i.e., we do not use the intrinsic or the extrinsic camera parameters supplied with the images). Again, in the real-world application scenarios, such intrinsic and extrinsic camera parameters are not available, and most, if not all, consumer-market digital cameras and phones are not calibrated. A 3D modeling pipeline must be able to automatically calibrate the intrinsic and extrinsic camera parameters using nothing but the input images, without any outside assistance.

• We have concentrated on the "end results" and ignored the "by products" of such 3D processing. That is, our comparison is on the faithfulness of the 3D models, not on the accuracy of the recovered intrinsic and extrinsic camera parameters. We believe that in the consumer market, the end users are mainly interested in the 3D models. Furthermore, we believe that the situation where a modeling system estimated erroneous camera parameters but somehow still obtains correct 3D structures fortuitously is extremely unlikely (we have never observed such a phenomenon).

To summarize, our methodologies are to exercise the full 3D modeling pipeline, leading from input images directly to output 3D models, without any user intervention or parameter tuning, without using any external calibration data (i.e., the camera's intrinsic and extrinsic parameters) that are not embedded in the input images themselves, and do all these using photos of a reasonable size (VGA). This is what we envision a robust, commercial-grade 3D photo modeling system must be able to do.

Evaluation Procedures: We down-sampled the ground-truth data 20 times for comparison. Down-sampling is necessary because the 3D ground-truth data uncompressed to 1GB for Fountain-P11 and 1.4GB for Herz-Jesu-P25. Meshlab dumped core when opening such huge files. The file-size limitation was present even on a state-of-the-art Windows 7 desktop with an Intel Core i-3 3.3GHz processor, 6GB of memory, and 1TB of

Table 1: Accuracy Evaluation results

Data set	Herz-Jesu-1	Fountain-1	Herz-Jesu-2	Fountain-2
# of images	25	11	25	11
spatial resolution	640 $\times$ 480	640 $\times$ 480	2150 $\times$1434	2150 $\times$1434
Runtime	11min 01sec	4min 30sec	59min 48sec	7min 12sec
# of 3D points	342,949	140,785	1,437,984	1,072,139
# of faces	735,419	315,195	2,781,106	2,107,543
max error	4.74%	5.07%	4.42%	1.92%
median error	0.62%	0.62%	0.44%	0.23%
average error	0.41%	0.44%	0.26%	0.17%

disk space. After our 3D pipeline finished generating 3D models from the input VGA images, these models were aligned with the ground-truth models in a two-stage procedure:

We first used the manual alignment process provided by Meshlab [27] to roughly align our 3D models with the ground-truth models. The alignment process consisted of loading both models into Meshlab, manually specifying a small number of corresponding points in the two models to establish a rough alignment, and then allowing Meshlab to refine the initial manual alignment using an iterative ICP algorithm.

After the models were roughly aligned as in the previous step, we loaded both models into our own display program that allowed small x, y, z rotations and translations applied to the ground-truth models. We applied such small translations and rotations interactively and eye-balled the display for the best qualitative alignment results.

After models had been aligned, we computed an absolute error measure for each and every 3D point in our 3D models. This error was the minimum distance from a 3D point in our models to the closest points in the corresponding ground-truth models. We then computed a percentage error by dividing the absolute error distance by the largest dimension of the ground-truth models in the x, y, or z direction.

Evaluation Results: Our modeling pipeline ran on a Windows 7 desktop with an Intel Core i-3 3.3GHz processor, 6GB of memory, and 1TB of disk space. For each data set, we used two different spatial resolutions: VGA and 2150$\times$1434 (the latter for reference purpose). The runtime, density and accuracy statistics are summarized in Table 1 and the 3D models are depicted graphically in Figs. 5 to 8[3]. One can also download

[3]The reported run time in Table 1 for the Fountain data set was faster than that in Fig. 4 because we used a faster machine and exploited the parallelism in our codes to speed up the processing.

or view these models in 3D at http://www.visualsize.com/3ddemo/comparison/accuracy.html.

As can be seen in Table 1, our modeling algorithm was able to construct 3D models which confirmed with the ground-truth models very well. The average error was less than 0.5%—and this error was computed over hundreds of thousands of recovered 3D points. Increasing spatial resolution improves the accuracy only marginally, but can lengthen the computation time significantly in the case of Herz-Jesu. Visual inspection of Figs. 5 to 8 seems to indicate that the models are less complete at higher image resolution even when more 3D points and surface patches were recovered and the model accuracy improves. This is probably due to the more stringent feature analysis requirement put in place to filter out outliers that tend to arise from matching unstable, evanescent fine-level features in high-resolution images.

Another obvious question is why our system seems to perform that much better than all others. We can surmise how some performance gain came about by examining the published papers of Bundler, PMVS2, and ARC 3D Web Service and contrasting the technical descriptions there with our own implementation. But one caveat is that such an analysis can be superficial because the complexity of a 3D modeling system cannot be fully appreciated without a detailed perusal of the source codes. However, we have never attempted to understand the codes of Bundler and PMVS2 or trace their program executions, simply because such an effort is too great and unnecessary for the purposes of comparison. The performance gain over Project Photofly will most likely remain a mystery because Project Photofly is a commercial product with IP shrouded in secrecy.[4] We include it in the comparison because, after an exhaustive Web search, Project Photofly is the only such commercial product based on the SfM currently available, the product team has a prestigious pedigree (from the RobotVis Group at INRIA, headed by Dr. Olivier Faugeras), and from our experience, it is a very mature and robust package.

In order to generate dense point clouds, it is necessary to have dense features to start with. Bundler uses a demo version of SIFT from David Lowe [25] for detecting features. However, the demo version has a fixed feature detection threshold that cannot be changed by the end user. While the pre-set threshold is reasonable for most data sets in our testbed, we can obtain even denser image features by using feature detectors such as FAST [35] and OpenCV [31]. These feature detectors come with adjustable parameters that allow significantly denser features to the SfM computation than those from the demo version of SIFT.

[4]The authors did contact Project Photofly to inform Autodesk of our comparison results. However, no response has been received so far.

However, simply making features denser is not enough. Quality is as important as density, if not more so. Using low-quality features blindly may increase the chance of erroneous correspondences and introduce significantly more outliers, which make the solution process much less robust and stable. Again, this represents the fundamental trade-off between precision and recall. We have instituted two mechanisms to guard against introducing outliers when using more features. One mechanism is a pre-filtering process: We have used the standard, normalized 8-point algorithm [19, 18] to compute the structure and motion parameters. The 8-point algorithm imposes the epipolar constraint between a pair of images in the form of $x'^T Fx = 0$ where F is the fundamental matrix and x and x' are the normalized image coordinates in the homogeneous form. Combined with RANSAC [18], the pre-filtering process provides a powerful mechanism to eliminate outliers to maintain the stability of the computation.

A second mechanism is in the way we solve the core nonlinear optimization problem, commonly known as sparse bundle adjustment, or SBA [18]. While Bundler has elected to use a public-domain SBA package by Lourakis and Argyros [29, 28], we have implemented our own version of SBA. Furthermore, we have used a different non-linear solver than the Levenberg-Marquardt (LM) used in [28], which is called the Double Dog-Leg (DLL) method.

Both LM and DLL are variances of a general class of numerical methods called the Trust-Region Methods [8, 12]. Trust Region Methods combine gradient descent and Newton's method [11] to find a local minimum of the objective function from any initial point. The gradient descent method guarantees progress towards a minimum but converges slowly, while Newton's method converges quickly when near a local minimum but diverges otherwise. Trust Region Methods combine gradient descent and Newton's method by varying the trust region radius depending on each iteration's success in reducing the value of the objective function. When the current point is not near a minimum, the trust region radius is reduced, and short steps are taken in the gradient direction to ensure progress. As a minimum is approached, the trust region radius is increased and longer steps are taken in the Newton direction to converge quickly. The objective function is modelled with a quadratic, and the trust region radius is adjusted to contain an area in which the model is believed to accurately represent the objective function. In more detail, the step s taken at each iteration is determined by

$$min_s f(x_c + s) = f(x_c) + (\nabla f(x_c))s + \frac{1}{2}s^T(\nabla^2 f(x_c))s, ||s|| \leq \delta_c \quad (1)$$

where f is the objective function, x_c is the current point, δ_c is the current trust region radius, ∇f is the

gradient and $\nabla^2 f$ is the Hessian. The optimal solution to the problem is shown to be [11]

$$s(\mu) = -((\nabla^2 f(x_c)) + \mu I)^{-1}(\nabla f(x_c)), ||s(\mu)|| = \delta_c \quad (2)$$

where I is the identity matrix and μ is chosen so that the length of the step is equal to the trust region radius.

LM varies μ as a parameter from iteration to iteration, rather than explicitly maintaining a trust region. Because of this, the expensive solution in Eq. 2 must be recomputed every time μ is changed. This can occur multiple times for the same values of ∇f and $\nabla^2 f$ if several unsuccessful steps are attempted from the same point. In fact, the optimal solution to Eq. 1 for varying the trust region size lies along a curved path starting along the gradient direction and connecting to the Newton step [8]. DDL uses a 3-piece approximation to this curve, which allows Eq. 2 to be solved many times for varying μ without needing to solve the entire linear system again. This approximation leads to large improvement in both efficiency and accuracy over LM, which may very well be an important factor of the improved accuracy and robustness in our pipeline.

Acknowledgements

We wish to thank Dr. Dan Koppel for providing his 8-point camera motion analysis code, professor Matthew Turk for his careful proofreading of this manuscript, and Ms. Gong Yi for helping with Meshlab. Part of the research was supported by funding from U.S. Navy and U.S. Army conducted in the author's employment as a Professor at the University of California, Santa Barbara.

References

[1] 2d3. http://http://www.2d3.com/, 2010.

[2] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building Rome in a Day. In *Proc. Int. Conf. Comput. Vision*, 2009.

[3] ARC 3D Webservice. http://homes.esat.kuleuven.be/ visit3d/webservice/v2/index.php, 2006.

[4] Autodesk. http://labs.autodesk.com/technologies/photofly/, July 2010.

[5] C.-I. Chen and D. Sargent and C.-M. Tsai and Yuan-Fang Wang and D. Koppel. Stabilizing Stereo Correspondence Computation Using Delaunay Triangulation and Planar Homography. In *Lecture Notes in Computer Science, 4th International Symposium on Visual Computing (ISCV)*, volume 5358, pages 846–855, 2008.

[6] C.-I. Chen and D. Sargent and C.-M. Tsai and Yuan-Fang Wang and D. Koppel. Feature Detector and Descriptor for Medical Images. In *Proceedings of the SPIE Medical Imaging Conference*, Feb. 2009.

[7] C.-I. Chen and D. Sargent and C.-M. Tsai and Yuan-Fang Wang and D. Koppel. Uniscale Multi-view Registration Using Double Dog-Leg Method. In *Proceedings of the SPIE Medical Imaging Conference*, Feb. 2009.

[8] A. Conn, N. Gould, and P. Toint. *Trust Region Methods, MPS-SIAM Series On Optimization*. SIAM, Philadelphia, PA, 2000.

[9] D. Scharstein and R. Szeliski. http://vision.middlebury.edu/stereo/, 2002.

[10] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-Time Single Camera Slam. *IEEE Trans. Pattern Analy. Machine Intell.*, 29, 2007.

[11] J. Demmel. *Applied Numerical Linear Algebra*. SIAM, 1997.

[12] J. Dennis and R. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. SIAM, Philadelphia, PA, 1996.

[13] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification, 2nd Ed.* Wiley, New York, 2001.

[14] O. Faugeras. *Three-Dimensional Computer Vision*. MIT Press, Cambridge, MA, 1993.

[15] Y. Furukawa, B. Curless, S. Seitz, and R. Szeliski. Reconstructing Building Interior from Images. In *Proc. Int. Conf. Comput. Vision*, 2009.

[16] Y. Furukawa and J. Ponce. Accurate, Dense, and Robust Multi-View Stereopsis. *IEEE Trans. Pattern Analy. Machine Intell.*, 1:1–14, 2008.

[17] Y. Furukawa and J. Ponce. http://grail.cs.washington.edu/software/pmvs/, 2010.

[18] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Camridge, MA, 2003.

[19] R. I. Hartley. In Defense of the Eight-Point Algorithm. *IEEE Trans. Pattern Analy. Machine Intell.*, 19, 1997.

[20] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer-Verlag, New York, NY, 2001.

[21] G. Klein and D. Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *International Symposium on Mixed and Augmented Reality*, 2007.

[22] L. Kobbelt and M. Botsch. A Survey of Point-Based Techninques in Computer Graphics. *Computers & Graphics*, 6:801–814, 2004.

[23] D. Koppel, C.-I. Chen, Y.-F. Wang, H. Lee, J. Gu, A. Poirson, and R. Wolters. Toward Automated Model Building from Video in Computer Assisted Diagnoses in Colonoscopy. In *Proceedings of the SPIE Medical Imaging Conference*, San Diego, CA, Feb. 2007.

[24] M. Lhuillier and L. Quan. A quasi-dense approach to surface reconstruction from uncalibrated images. *IEEE Trans. Pattern Analy. Machine Intell.*, 2005.

[25] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*, 60:91–100, 2004.

[26] Lukas Mach. http://insight3d.sourceforge.net/, 2010.

[27] Meshlab. http://meshlab.sourceforge.net/, 2010.

[28] M.I.A. Lourakis and A.A. Argyros. http://www.ics.forth.gr/ lourakis/sba/, 2008.

[29] M.I.A. Lourakis and A.A. Argyros. SBA: A Software Package for Generic Sparse Bundle Adjustment. *Transactions on Mathematical Software*, 36:1–30, 2009.

[30] N. Snavely. http://phototour.cs.washington.edu/bundler/, 2008.

[31] OpenCVWiki. http://opencv.willowgarage.com/wiki/, 2010.

[32] M. Pollefeys, L. V. Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch. Visual Modeling with a Hand-held Camera. *Int. J. Comput. Vision*, 59:207–232, 2004.

[33] M. Polleyfeys and *et al.* Detailed Real-Time Urban 3D Reconstruction from Video. *Int. J. Comput. Vision*, pages 143–167, Oct. 2007.

[34] L. Quan. *Image-Based Modeling*. Springer, 2010.

[35] E. Rosten and T. Drummond. Fusing Points and Lines for High Performance Tracking. In *Proc. Int. Conf. Comput. Vision*, volume 2, pages 1508–1511, October 2005.

[36] D. Scharstein and R. Seliski. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *Int. J. Comput. Vision*, 47:7–42, 2002.

[37] S. Schwartz. *Visual Perception: A Clinical Orientation, 4th Ed.* McGraw-Hill, New York, NY, 2009.

[38] S. Seitz, B. Curless, D. Scharstein, and R. Szeliski. A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recognit.*, 2006.

[39] N. Snavely, S. Seitz, and R.Szeliski. Photo Tourism: Exploring Photo Collections in 3D. In *ACM SIGGRAPH Conf. Proc.*, pages 332–338, 2006.

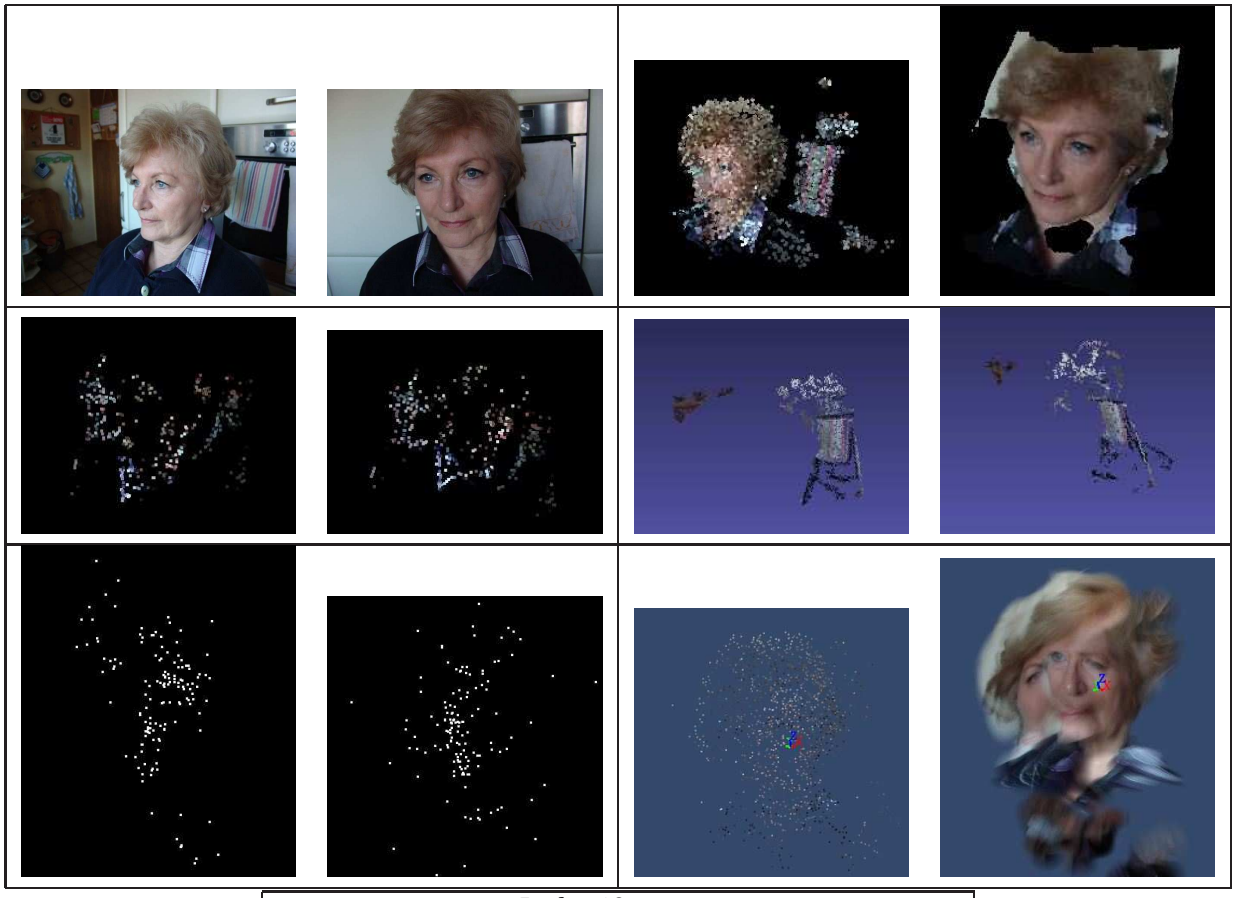
[40] N. Snavely, S. M. Seitz, and R. Szeliski. Modeling the World from Internet Photo Collections. *Int. J. Comput. Vision*, 80:189–210, 2008.

[41] C. Strecha, W. von Hansen, L. V. Gool, P. Fua, and U. Thoennessen. On Benchmarking Camera Calibration and Multi-View Stereo for High Resolution Imagery. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recognit.*, 2008.

[42] M. Vergauwen and L. V. Gool. Web-based 3D Reconstruction Services. *Machine Vision and Applications*, 17:411–426, 2006.

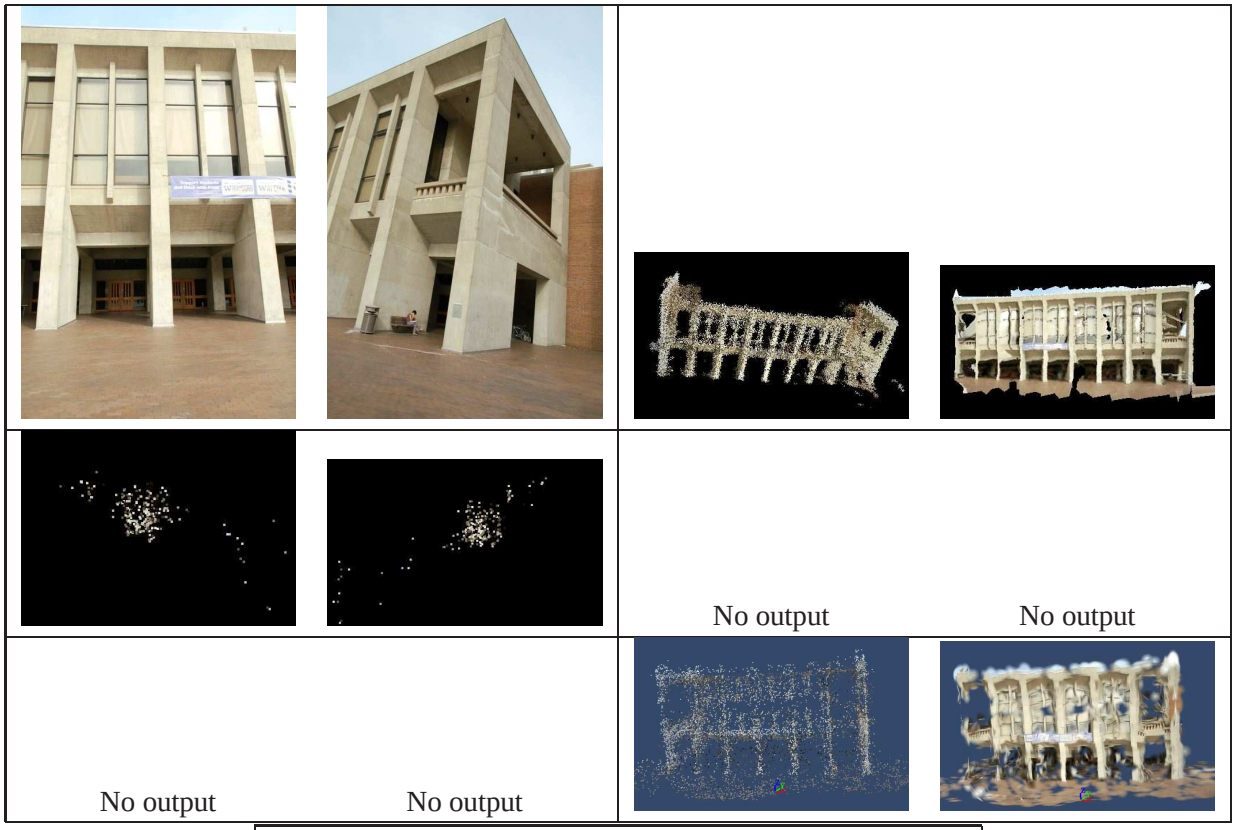
[43] G. Xu and Z. Zhang. *Epipolar Geometry in Stereo, Motion and Object Recognition*. Kluwer Academic Publishers, The Netherlands, 1996.

[44] C. Zach. http://www.inf.ethz.ch/personal/chzach/opensource.html, 2010.

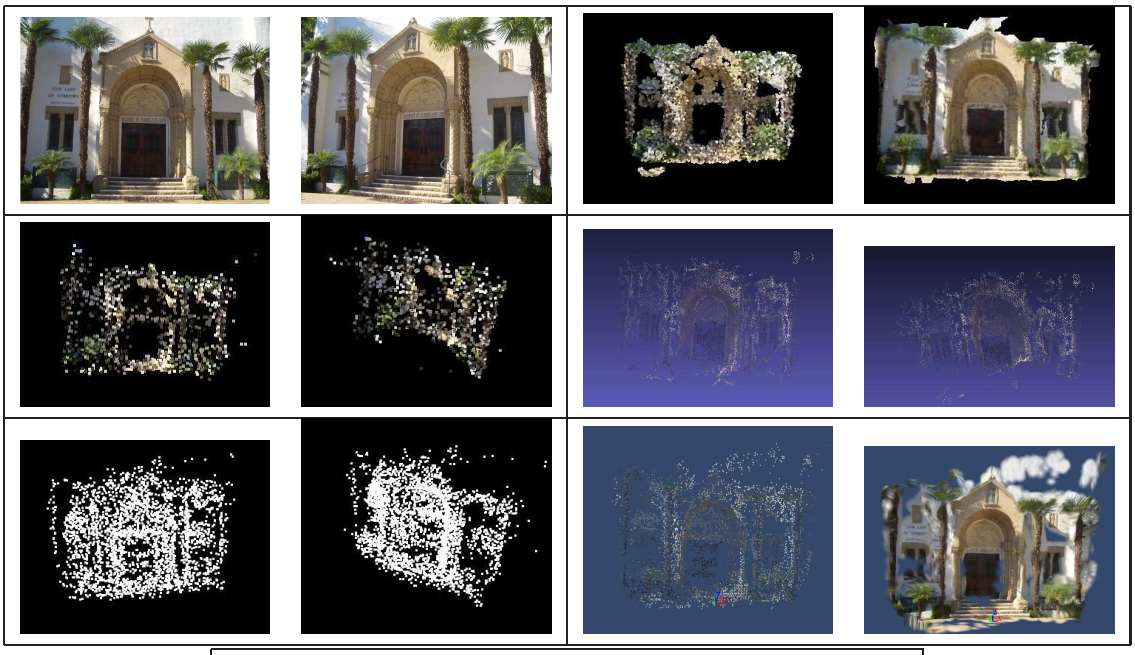


Lady: 10 images				
	Bundler	Bundler + PMVS2	ARC 3D	Ours
Pic processed	4	4	6	10
# 3D points	467	2,428	148	10,810
Run time	0:39	0:58	-	2:00

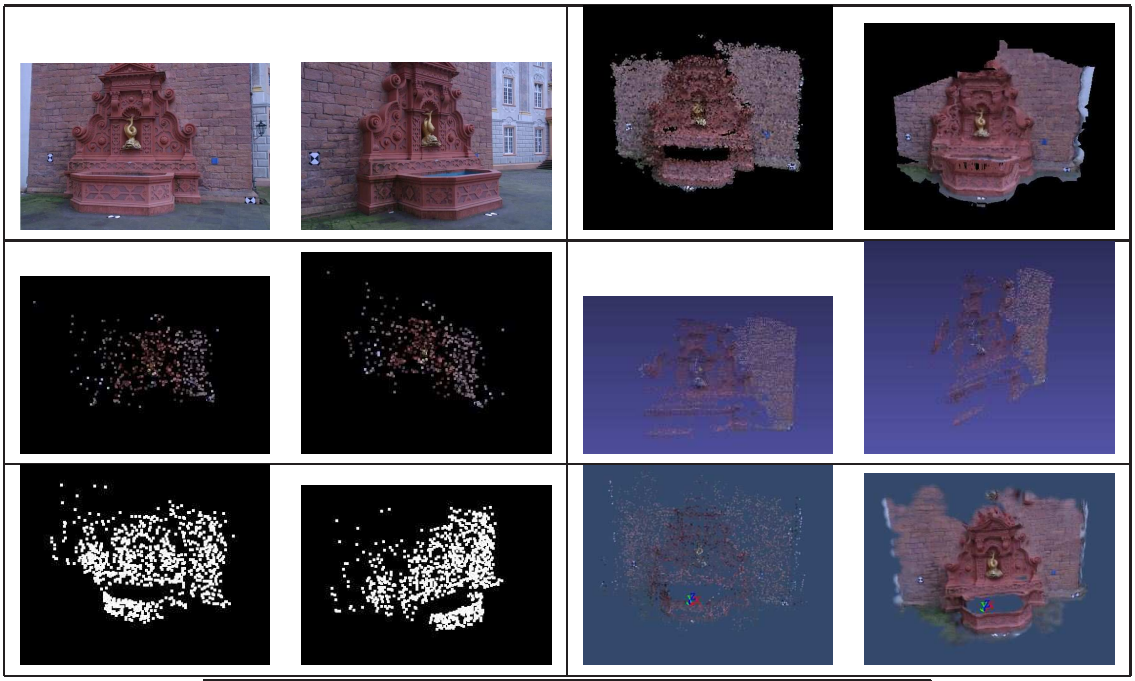
Figure 4: For all data sets in this figure, the two images at top left are sample input images, the two at top right are our results, the two at middle left are results of Bundler, the two at middle right are results of Bundler + PMVS2, the two at bottom left are results of ARC 3D Web Service, and the two at bottom right are results of Autodesk's Project Photofly. For all data sets, the table below the graphic results shows the name and size of the data set, how many pictures are processed, how many 3D points are generated, and the runtime for Bundler, Bundler+PMVS2, ARC 3D Web Service, and our system. This figure continues onto the following 5 pages.



Building: 61 images				
	Bundler	Bundler + PMVS2	ARC 3D	Ours
Pic processed	11	11	0	61
# 3D points	384	0	0	112,762
Run time	4:00	4:16	-	20:25

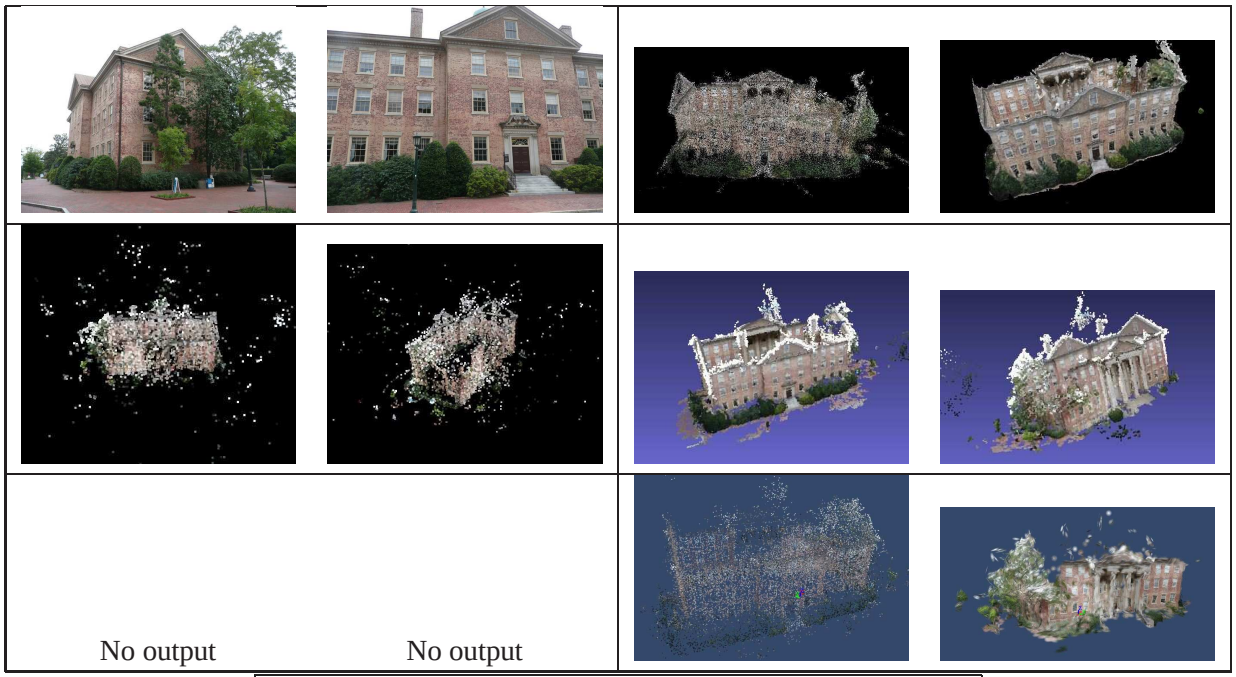


Church: 24 images				
	Bundler	Bundler + PMVS2	ARC 3D	Ours
Pic processed	5	5	24	24
# 3D points	1,643	14,315	3,812	38,796
Run time	3:00	7:24	-	10:03

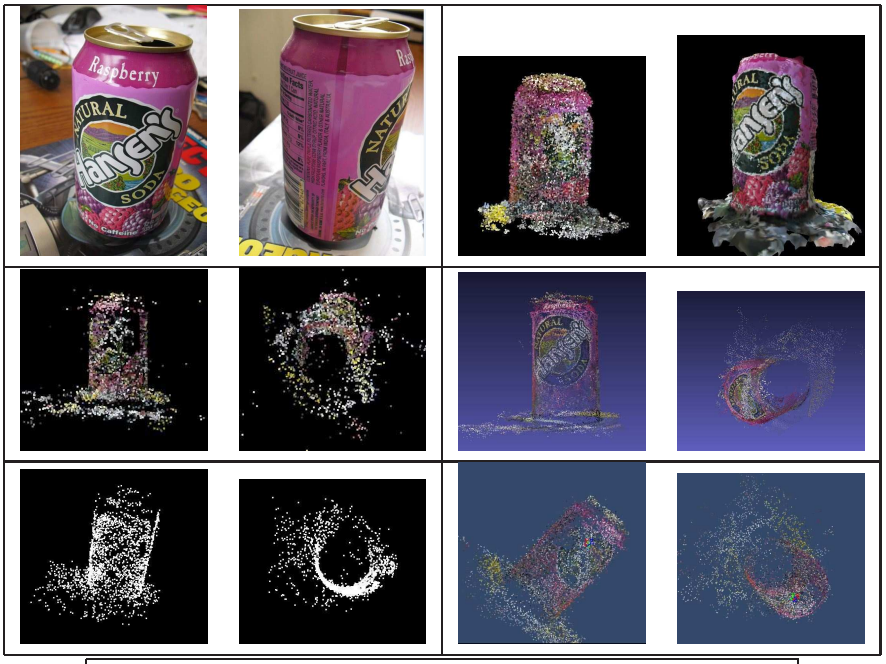


Fountain: 11 images				
	Bundler	Bundler + PMVS2	ARC 3D	Ours
Pic processed	5	5	11	11
# 3D points	604	7,221	1,025	132,670
Run time	0:55	1:30	-	6:05

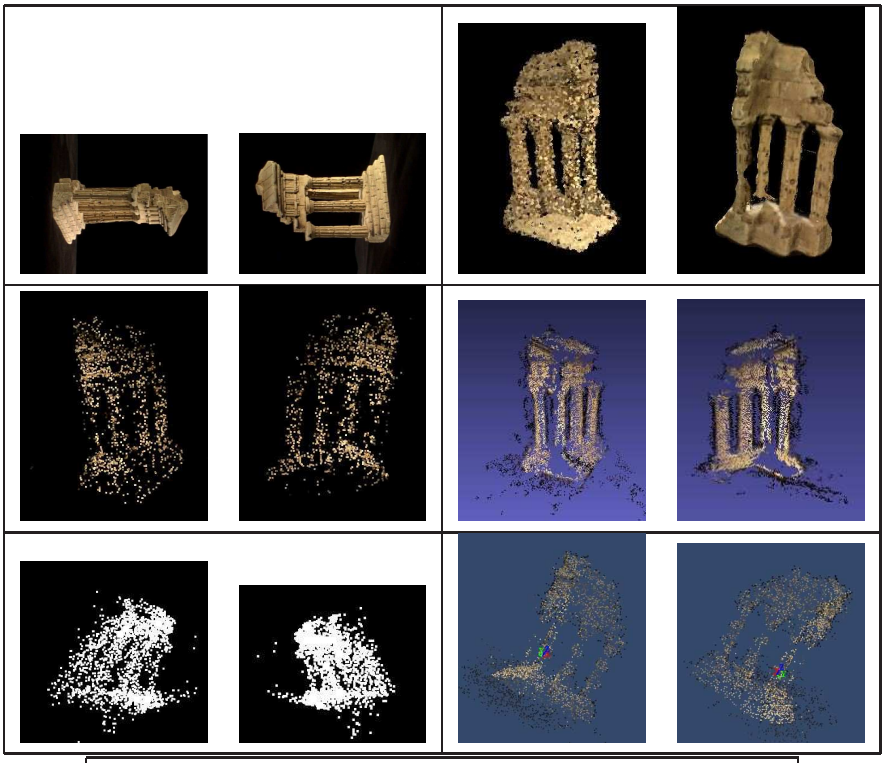
NTU Gate: 34 images				
	Bundler	Bundler + PMVS2	ARC 3D	Ours
Pic processed	11	11	6	34
# 3D points	618	793	117	64,046
Run time	3:18	3:31	-	8:58



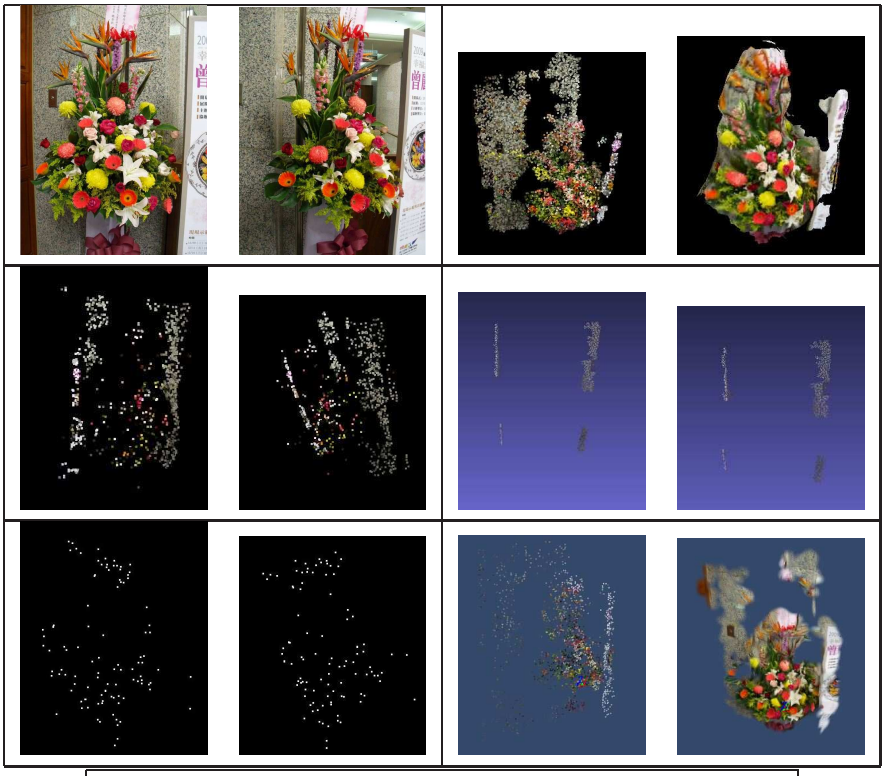
UNC: 128 images				
	Bundler	Bundler + PMVS2	ARC 3D	Ours
Pic processed	53	53	0	85
# 3D points	18,516	95,440	0	191,054
Run time	42:16	59:34	-	77:23



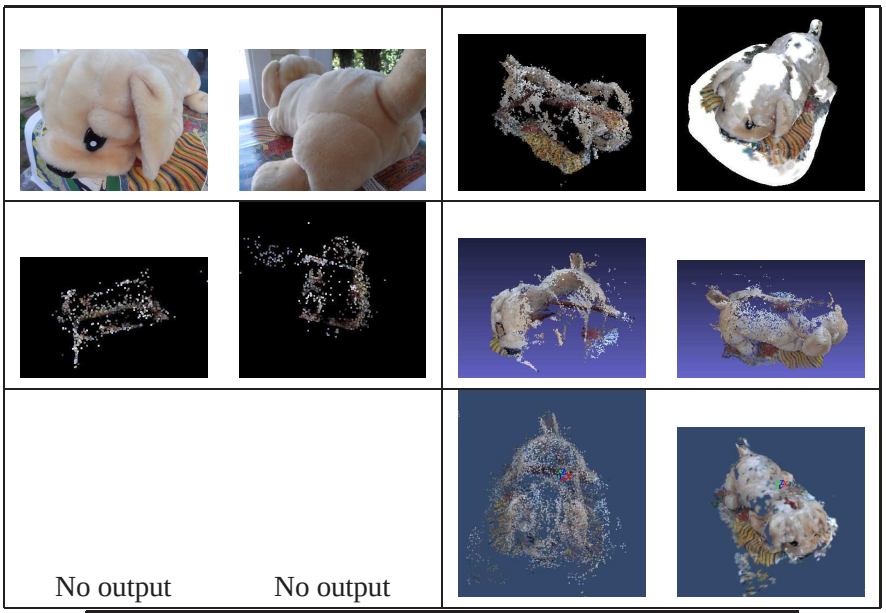
Soda Bottle: 46 images				
	Bundler	Bundler + PMVS2	ARC 3D	Ours
Pic processed	18	18	35	46
# 3D points	6,214	27,221	2,473	75,275
Run time	5:48	11:07	-	11:18



Temple Ring: 47 images				
	Bundler	Bundler + PMVS2	ARC 3D	Ours
Pic processed	22	22	37	47
# 3D points	1,766	8,705	2,407	33,988
Run time	3:16	5:35	-	18:22



Flower: 7 images				
	Bundler	Bundler + PMVS2	ARC 3D	Ours
Pic processed	5	5	5	7
# 3D points	1,054	1,162	107	16,180
Run time	0:51	0:59	-	1:48



Droopy: 89 images				
	Bundler	Bundler + PMVS2	ARC 3D	Ours
Pic processed	38	38	0	61
# 3D points	3,398	53,428	0	61,752
Run time	6:00	19:33	-	30:04

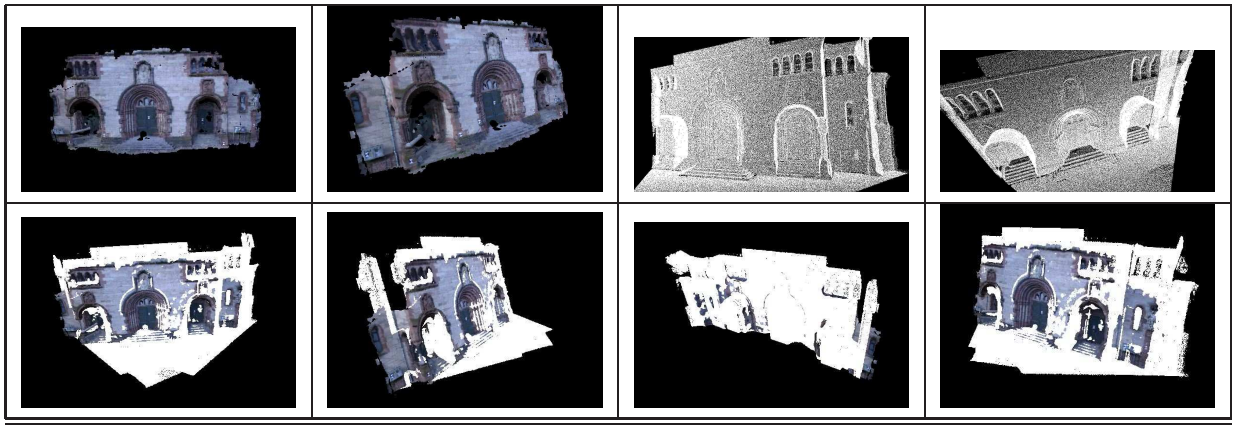


Figure 5: Herz-Jesu-P25 data sets processed at the VGA resolution. Left two on the top row: our results. Right two on the top row: ground truth. The bottom row displays both our model and the ground truth registered in a common reference frame. Short movies of these models can be viewed at http://www.visualsize.com/3ddemo/comparison/accuracy.html.

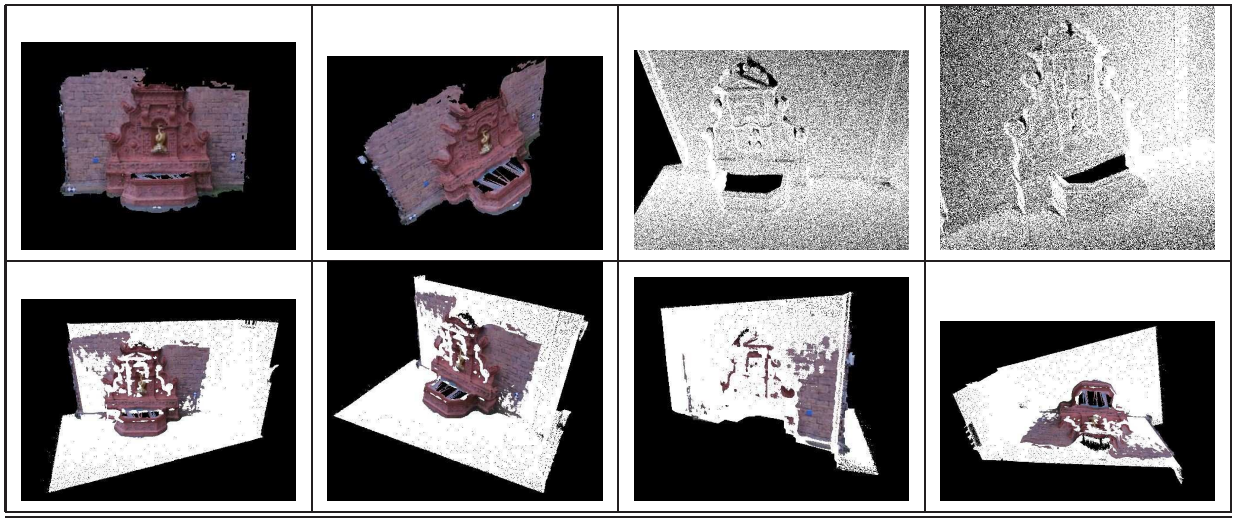


Figure 6: Fountain-P11 data sets processed at the VGA resolution. Left two on the top row: our results. Right two on the top row: ground truth. The bottom row displays both our model and the ground truth registered in a common reference frame. Short movies of these models can be viewed at http://www.visualsize.com/3ddemo/comparison/accuracy.html.

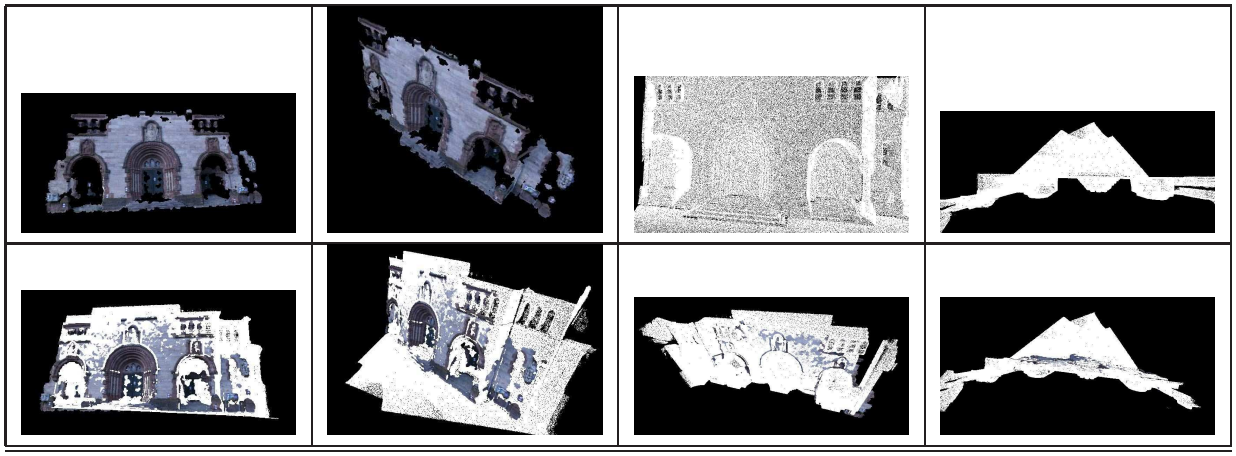


Figure 7: Herz-Jesu-P25 data sets processed at the 2150 $\times$1434 resolution. Left two on the top row: our results. Right two on the top row: ground truth. The bottom row displays both our model and the ground truth registered in a common reference frame. Short movies of these models can be viewed at http://www.visualsize.com/3ddemo/comparison/accuracy.html.

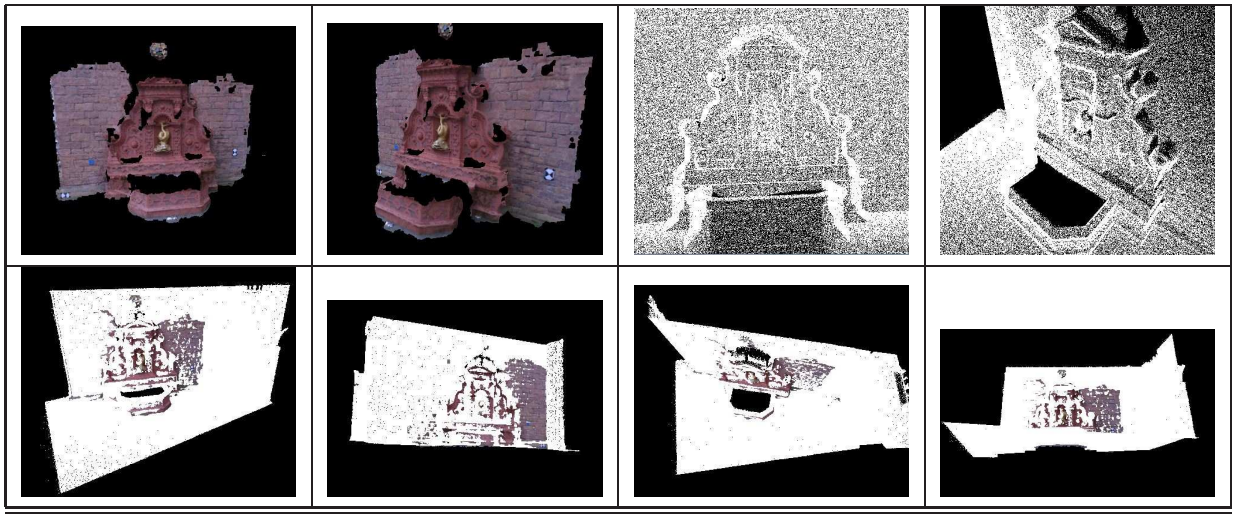


Figure 8: Fountain-P11 data sets processed at the 2150 $\times$1434 resolution. Left two on the top row: our results. Right two on the top row: ground truth. The bottom row displays both our model and the ground truth registered in a common reference frame. Short movies of these models can be viewed at http://www.visualsize.com/3ddemo/comparison/accuracy.html.