ReconstructMe SDK: a C API for Real-time 3D Scanning

Christoph HEINDL[1], Harald BAUER, Martin ANKERL, Andreas PICHLER
PROFACTOR GmbH, Steyr, Austria

DOI: 10.15221/15.185 http://dx.doi.org/10.15221/15.185

Abstract

Summary: ReconstructMe SDK[2] is an ISO C library for performing real-time 3D reconstruction for hand-operated RGB-D camera devices. It provides interfaces for entire 3D scanning pipeline including pre and post processing steps, such as sensor data filtering and 3D surface post processing tasks. Developers can embed ReconstructMe SDK in their applications, saving themselves the work of implementing their own scanning pipeline. To target a wide range of applications ReconstructMe SDK designed to scale simple single sensor applications to distributed multi-sensor frameworks with cloud based 3D reconstruction back-ends.

Availability: ReconstructMe SDK was first released in June 2013. Binaries, x86 and x64, are available for all common Windows platforms including mobile versions.

Keywords: 3d body scanning, 3D surface generation, real-time scanning, 3D documentation

1. Introduction

3D reconstruction and 3D documentation are enabling technologies for customized product development. These personalized products require appropriate 3D modelling techniques. Among these techniques 3D scanning has recently experienced a strong demand, mostly due to increased computational power and availability of low cost RGB-D (color and depth) sensors. With the recent rise of low-cost RGB-D sensors that are able to generate dense 3D depth maps at frame rates of 30 Hz, real-time hand-held 3D scanning became achievable [1].

ReconstructMe SDK is designed as a middleware library that provides an easy to use interface for developers who need to integrate a high performance 3D reconstruction engine into their applications. As a middleware library it encapsulates communication with popular 3D sensors and abstracts away hardware details such as task parallelization. The library is designed to be

- **easy-to-use** A generic and consistent SDK based on ISO C allows you grasp the concepts quickly and develop your first reconstruction application within minutes.
- **easy-to-integrate** The SDK has pure C-based API without additional compile time dependencies. Interoperability with other languages is easily possible.
- **high-performance** The SDK is designed to provide a maximum performance for a smooth reconstruction experience.

[1] {christoph.heindl, harald.bauer, martin.ankerl, andreas.pichler}@profactor.at
[2] http://reconstructme.net

2. Library Architecture

2.1 Overview

The approach proposed in this paper is based on the work of [1] and optimized regarding portability, flexibility, usability and real-time data processing. The system environment consists of four main parts as shown in Figure 1.

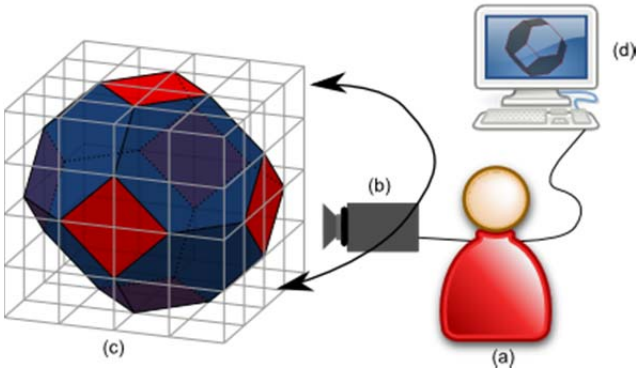

Figure 1 System components. (a) User, (b) Sensor, (c) Global model, (d) Computer / real-time visualization

In the simplest case a user hand operates a single sensor, moving it around the object to be digitized. ReconstructMe SDK keeps track of the sensor position by methods of visual odometry [2] and fusions the RGB-D sensor data into a single consistent 3D model. During the reconstruction the memory usage is constant, independent of scan duration and sensor data acquired (roughly 30 Mbyte per second). The ReconstructMe engine automatically fills in previously unseen regions and updates known areas with more recent scan data. This method, in general, leads to more accurate results as sensor data is averaged from different viewpoints as shown in section 2.4.

2.2 Hardware Considerations

ReconstructMe SDK relies on specific hardware to provide real-time reconstruction experience. It makes heavy use of the computational power of modern graphic cards through OpenCL [3] standard. OpenCL allows developers to write portable, high-performance code that can target all varieties of parallel processing platforms, including INTEL, NVIDIA, AMD CPUs and GPUs. Like with any parallel programming model for parallel processing, achieving good efficiency requires careful attention to how the computation is mapped to the hardware platform and executed.

ReconstructMe SDK uses real-time RGB-D sensors to generate a projection of the real world. Such camera devices ideally provides a high resolution depth map and color image at a frame rate of 30 Hz or faster. At the point of writing, ReconstructMe SDK supports sensors from Microsoft[3], PrimeSense, Asus[4] and Orbbec[5].

2.3 The API

ReconstructMe SDK provides a small set of well-defined C interfaces that are outlined in the next paragraphs.

The Context interface `reme_context_t` provides object life-time management and maintains the communication with a computation device (GPU, CPU). Every program needs at least one instance of it. The Sensor interface `reme_sensor_t` provides access to Image `reme_image_t` data of the real world. This data is fed into a Volume `reme_volume_t` where the actual reconstruction happens. The Surface interface `reme_surface_t` allows the extraction of 3D triangle meshes from the volume.

By default all interfaces created come with sensible default values. To adjust or access those settings, you the generic Options `reme_options_t` interface can be used, which can be bound to any parameter set through so called binder-methods.

[3] https://dev.windows.com/en-us/kinect
[4] https://www.asus.com/3D-Sensor/Xtion_PRO_LIVE/
[5] http://www.orbbec3d.com/index.php

ReconstructMe SDK comes with inline visualization utilities available via the Viewing interface `reme_viewer_t`. These utilities are meant for debugging purposes and not for a production ready applications, since their implementation is designed to be general purpose rather than computationally efficient. For testing purposes ReconstructMe API a Recorder `reme_recorder_t` interface to stream sensor data to disk. This data can later be replayed using a special file sensor.

To improve the accuracy of 3D sensors, ReconstructMe SDK provides the Calibration interface `reme_calibrator_t`. It allows estimation of camera intrinsic camera parameters (such as focal length, principal point and distortions). These parameters can later be used for reconstruction to compensate errors.

2.4 Example

The following example code shows a complete 3D scanning application utilizing a single hand-held 3D sensor. The initialization step below creates all necessary instances to perform scanning. At the beginning a new `reme_context_t` is created which manages all ReconstructMe SDK objects. The context is prepared for the default OpenCL device available. Next, a `reme_volume_t` with default size (1 meter cubed) is initialized in which all sensor data will fused. Finally, any connected 3D RGB-D `reme_sensor_t` is opened.

```
// Create a new context
reme_context_t c;
reme_context_create(&c);
// Compile for OpenCL device using defaults
reme_context_compile(c);

// Create a new volume
reme_volume_t v;
reme_volume_create(c, &v);

// Create a new sensor
reme_sensor_t s;
reme_sensor_create(c, "openni;mskinect;file", true, &s);
reme_sensor_open(c, s);
```

Once initialization is complete the main scanning loop is entered. This loop can be decomposed in three steps: initially a RGBD image from the real world is acquired through the sensor interface. Next, the 6DoF sensor movement is estimated with respect to data already present in the volume. If sensor movement tracking succeeds, the current RGB-D data is integrated into the global model using the estimated new sensor position.

```
// Reconstruction main loop
while (REME_SUCCESS(reme_sensor_grab(c, s))) {
       // Prepare image and depth data on GPU
       reme_sensor_prepare_images(c, s);
       // Estimate sensor movement via 3D data tracking
       if (REME_SUCCESS(reme_sensor_track_position(c, s))) {
              // Update volume with depth data from the
              // current sensor perspective
              reme_sensor_update_volume(c, s);
       }
}
```

After the scanning is complete the content of the volume is available for further processing. Here the volume information into a 3D surface representation by means of Marching Cubes [9]. The surface information is finally saved in one of the supported file formats such as .PLY, .STL, .OBJ[6].

```
// Create a new surface
reme_surface_t m;
reme_surface_create(c, &m);
reme_surface_generate(c, m, v);
reme_surface_save_to_file(c, m, "surface.ply");
```

2.5 Features

ReconstructMe SDK is applied in many real world applications. Examples include digitization of production halls, human body scanning for medical purposes and creation of virtual characters in movies. ReconstructMe SDK provides a variety of features for various areas of applications. Some of them are summarized in the following paragraphs.

2.5.1 Multi Sensor

ReconstructMe SDK supports a wide range of commodity RGBD sensors such as the ASUS Xtion Family, the PrimeSense Carmine Family or the Microsoft Kinect family. Besides single sensor scenarios, ReconstructMe SDK supports multi-sensor scenarios where many sensors are working together on creating a global model.

Figure 2] shows a snapshot from a video[7] of a cooperative reconstruction between two freely moved sensors. Both sensors are tracked simultaneously with respect to a single reconstruction volume.

Figure 2 Multi-sensor cooperative reconstruction. Left: Image of setup.
Top-right: first sensor. Bottom-right: second sensor.

2.5.2 Texture Support

ReconstructMe SDK is capable of capturing and processing the color information[8] of the object being scanned, as long as the sensor provides the necessary color stream. These color information can be applied either per surface vertex or as texture. In the second case the triangulated surfaces is decomposed into planar texture patches onto which the final texture is rendered. The process is illustrated below.

[6] https://www.youtube.com/watch?v=dcnPXDVtIR0
[7] http://www.youtube.com/watch?v=Xg2Yyk70_9E
[8] http://www.youtube.com/watch?v=1XDvA3yDA0E

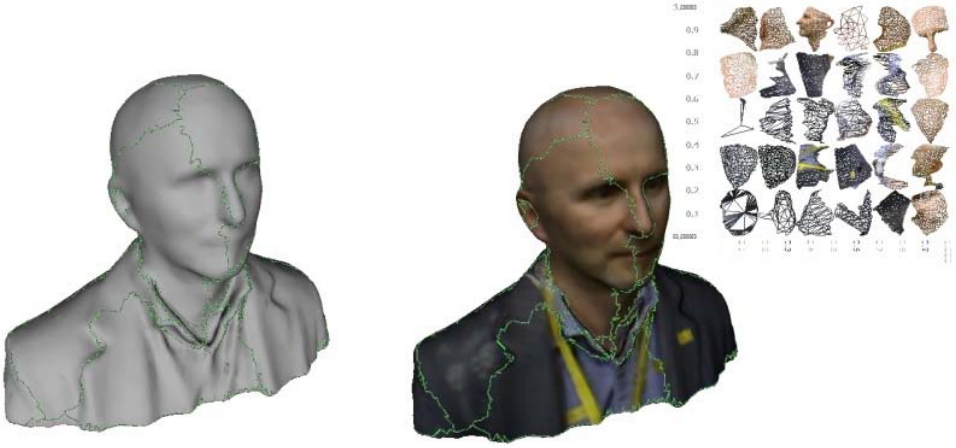

Figure 3 Texturing Pipeline of ReconstructMe. Left: Water-tight mesh with texture seams. Right: Colored surface with texture atlas.

2.5.3 Mesh Post-Processing

ReconstructMe SDK allows you to manipulate the scan result for further processing. Supported operations comprise removing noise, making meshes watertight, simplifying topology and performing CSG [4] operations on surfaces. These operations allow you to perform intersections, unions or differences between volume content and other objects. Currently, ReconstructMe SDK supports the following object types: spheres, axis aligned boxes, and planes, and free-form 3D surfaces.

In Figure 4 the CSG operations (union, difference, and intersection) are applied to a box and a sphere for illustrative purposes.

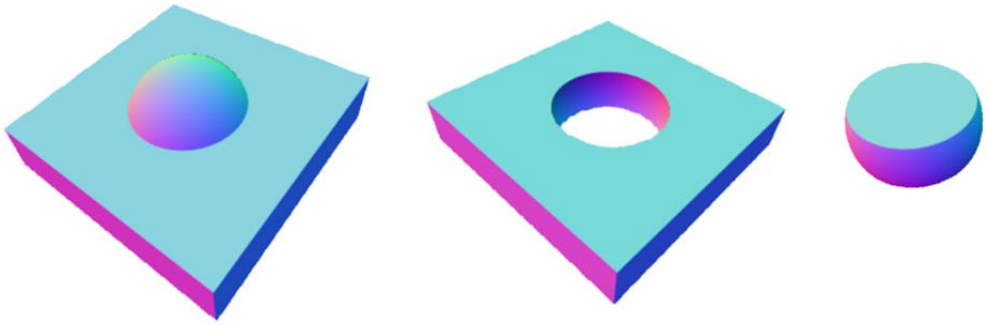

Figure 4 CSG operations between a sphere and a box. Left: union, middle: difference, right: intersection

All operations are performed on the volume and generate intersection free meshes when extracted. Additionally, we've added support for complex meshes. This allows you to easily add stands or cut your reconstruction.

2.5.4 Marker Support

ReconstructMe SDK supports marker detection and tracking. Markers can be used to pinpoint real-world 3D positions in the scan, or position the scan volume with respect to a marker placed in the room.

A marker is a special object placed in the view of the camera that has a couple of unique features: from a software point of view it is easily detectable, allows the estimation of camera pose with respect to the marker frame and does not need dense 3d data. Commonly used markers include RUNE-Tags [5] and Matrix markers [6] as shown in Figure 5.

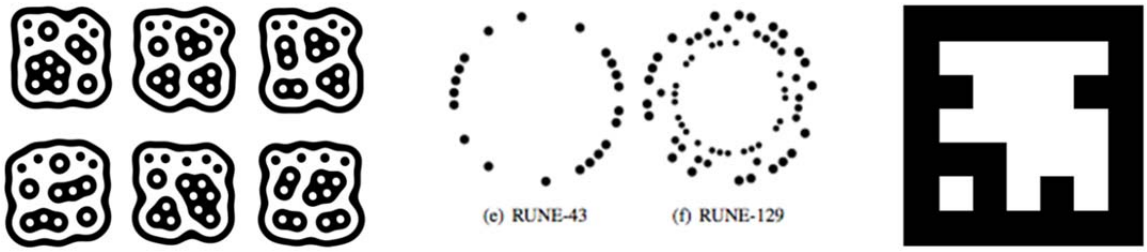

Figure 5 Different type of markers. a.) topological markers b.) RUNE Tags c.) Matrix markers as used by ReconstructMe

Markers allow you to

- **Define canonical world coordinate system** The marker defines the position and orientation of the world coordinate system[9]. It is designed in such a way that if laid on the floor, the z-axis of the world frame points towards the ceiling.
- **Automatic remove of stands and floor** data By letting the world volume start a bit above the marker coordinate system you can cut away floor and turntable data.
- **Improve tracking** Markers can be used to perform camera tracking in regions which are otherwise hard to recognize.

2.5.6 Metric Reconstruction

ReconstructMe SDK performs the entire reconstruction in metric space. Modelling with ReconstructMe SDK scales from smaller objects such as human faces up to entire rooms. The result can be exported to various CAD formats such as STL, OBJ, 3DS, and PLY.

In [7] the accuracy of ReconstructMe SDK was evaluated by comparing scanning results from a multi-sensor scan of a doll with a known ground truth scan. Figure 6Figure 1 shows the surface distance error between a ground truth reference scan and a ReconstructMe turntable scan.

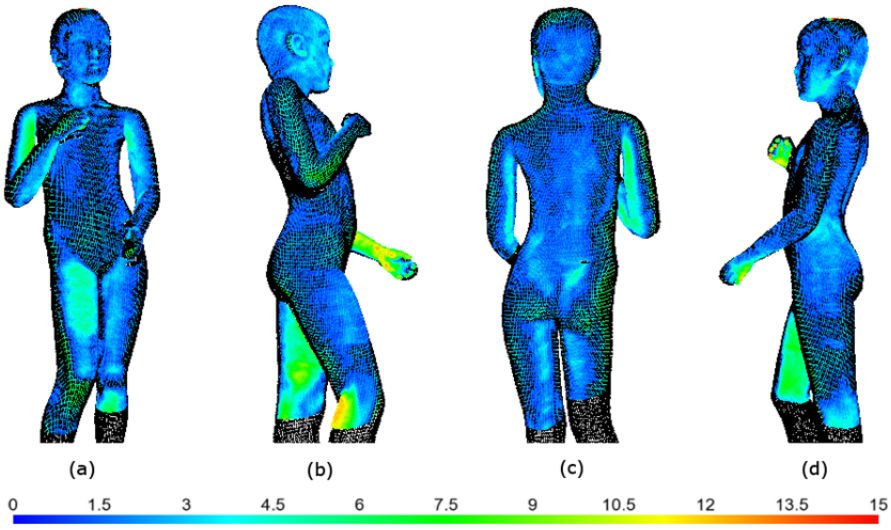

Figure 6 Quality measurements in millimeter of a scan, taken in a multi sensor setup.

The torso and head are very close to the reference scan. The highest errors occur on the limbs at (b) and (c). Statistics on error measurements are listed in Table 1.

[9] https://www.youtube.com/watch?v=eC9zuBkAKcE

	Mean [mm]	Median [mm]	Standard deviation [mm]
(a)	2.40313	2.08697	1.32874
(b)	2.89839	2.29675	2.02711
(c)	2.23083	1.96103	1.20602
(d)	2.47158	2.07378	1.55825

Table 1 Statistic values of quality measurements corresponding to Figure 6

The mean is about 2.5 millimeter and the standard deviation about 1.51 millimeters. Achievable accuracy highly depends on two factors

- **Sensor resolution and accuracy** ReconstructMe results are strongly linked to sensor accuracy. This accuracy tells how closely the output from a sensor will match the 'true' value. Unfortunately, the term accuracy is used differently among sensor manufacturers and therefore only figures on the resolution of the sensor are published.

- **Camera tracking** ReconstructMe tracks camera movements by evaluating differences between camera images. Therefore, if the target object does not offer any features to, tracking will be faulty and so will be the resulting scan. Camera tracking is also effected by the amount of movement between two subsequent camera frames. The larger the movement the higher the chances for bad alignment.

3. Applications

3.1 Customized Products

ReconstructMe SDK is used in variety of applications that aim at a reduction of production costs of customized products. ReconstructMe is employed to quickly and accurately scan body parts. Automatic measurements are then taken as parameters for manufacturing shoes, clothing and medical products. In all these applications the metric reconstruction is used as the fundament to manufacture individual products. Figure 7 shows an example of measurements taken on a 3D reconstruction for eyewear products.

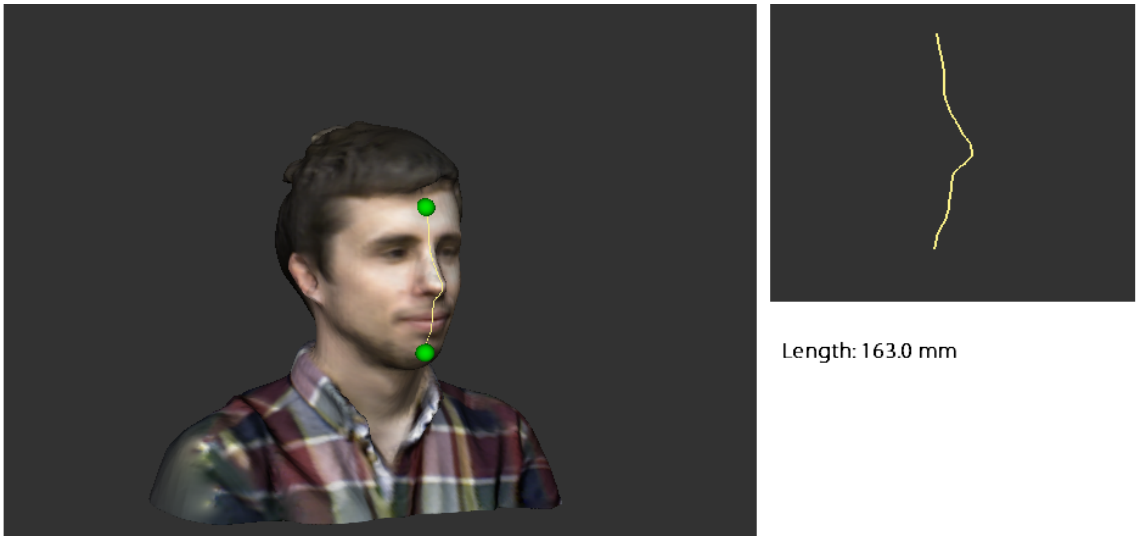

Figure 7 Automatic scan measurement. Nose profile measurement taken from 3D reconstruction.

3.2 3D Documentation of Production Plants

3D scanning with ReconstructMe SDK scales from small objects up to entire rooms or even production plants. The German company Ipo.Plan [8] created Ipo.Eye, an automatic robot system that digitizes the current state of a production hall. This plant reconstruction data is used as input for simulation and planning tasks. Figure 8 shows a visualization of the robot in action.

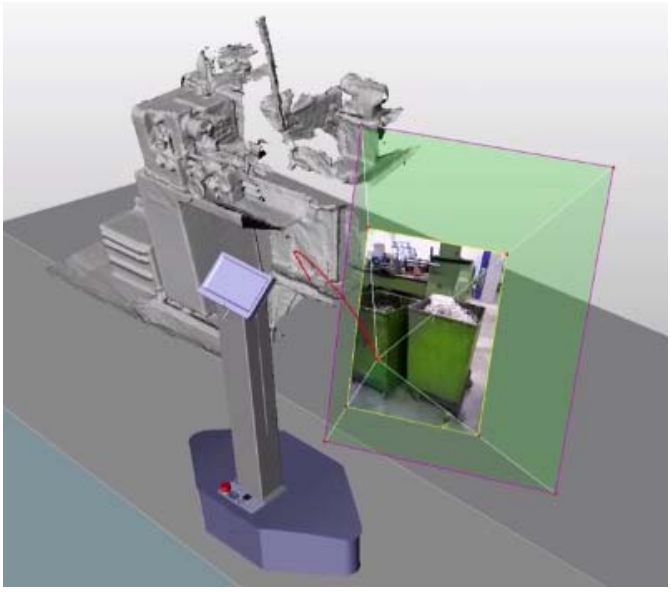

Figure 8 Ipo.Eye Robot digitizing a production hall. The green view frustum reflects the real world data, while the gray surface represents the reconstruction progress.

For areas where ground moved vehicles are impracticable, either because of obstacles or distances to be covered, ReconstructMe is used in conjunction with unmanned aerial vehicle (UAV) for parallel ground scanning and localization of the drone (SLAM algorithm) [10].

3.3 Rapid Creation of Realistic Avatars for Computer Games

ReconstructMe SDK has been used in the 3D gaming industry to digitize humans in order to create realistic avatars in computer games. One notably game is “This War of Mine” (TWoM) by 11 bit studios. Dominik Zieliński, lead artist, explains [11]

“For our scans we’ve used Kinect, a rotating platform (and some duct tape), and the results we’ve got were perfect for what we needed. I especially liked the cloth wrinkles that would require some time to simulate, or even more time to sculpt, and the effect probably wouldn’t be that good. That was a big plus because the characters in TWoM are mostly fully clothed. ReconstructMe contributed greatly into characters overall quality. Scanning heads is great. Percentagewise, while working on TWoM, most amount of time was saved thanks to scanning real people’s heads instead of sculpting them in 3d.”

Figure 9 shows the digitalization setup for the game.

Figure 9 Digitization of humans for the game This War of Mine

4. Conclusion and Future Work

The ReconstructMe SDK has been successfully integrated into a variety of scenarios. It provides components for real-time 3D scanning using low-cost RGB-D sensors. These components provide support for communicating with sensors, tracking 6 DoF camera movement via visual odometry, fusing of scan data into a global world frame and post-processing of scans. Applications range from 3D photo booths, custom-fitted products to 3D documentation of production halls and real estate.

ReconstructMe SDK is currently evaluated for extensions pushing 3D documentation and digitization to more 4D-like features: Object tracking capabilities of moving objects and activity observation capabilities for humans would widely open the field of applications. In a first use case, a production workplace is enhanced for the human worker. The goal here is to provide intelligent and smart assistance during human assembly activities. For establishing such a work place, it is necessary to have a model of the world for static elements (tables, constructions, etc.) but also for moving objects (work pieces, human) as digital pre-requisite. During the assembly procedure, the software needs to 'understand' (i.e. having a good estimate) about what is going on[10]. The form of assistance for the human worker to indicate the next task is done either by live instructions directly on known static elements (work pieces, tables) via projections or by live instructions integrated in smart glasses. Alternatively, it's even possible to trigger support actions by robot co-workers.

5. Acknowledgement

The research leading to these results has partly received funding from the FFG - Austrian Research Promotion Agency within the FTI Initiative "Intelligent Production" (InstructMe (PdZ-843693)).

References

[1] Newcombe, Richard A., et al. "KinectFusion: Real-time dense surface mapping and tracking." *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*. IEEE, 2011.
[2] Whelan, Thomas, et al. "Robust real-time visual odometry for dense RGB-D mapping." *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013.
[3] Stone, John E., David Gohara, and Guochun Shi. "OpenCL: A parallel programming standard for heterogeneous computing systems." *Computing in science & engineering* 12.1-3 (2010): 66-73.
[4] Requicha, Aristides AG, and Herbert B. Voelcker. "Constructive solid geometry." (1977).
[5] Bergamasco, Filippo, et al. "RUNE-Tag: A high accuracy fiducial marker with strong occlusion resilience." *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011.
[6] Munoz-Salinas, R. "ARUCO: a minimal library for Augmented Reality applications based on OpenCv." (2012).
[7] C. Kopf et al., "A Portable, Low-Cost 3D Body Scanning System", in Proc. of 4th Int. Conf. on 3D Body Scanning Technologies, Long Beach CA, USA, 2013, pp. 417-425, http://dx.doi.org/10.15221/13.417
[8] Ipo.Plan. 2015. Ipo.Plan Website. Available at: http://www.ipoplan.de/
[9] Lorensen, William E., and Harvey E. Cline. "Marching cubes: A high resolution 3D surface construction algorithm." *ACM siggraph computer graphics*. Vol. 21. No. 4. ACM, 1987.
[10] Huber G.: Full Autonomous Quadcopter for Indoor 3D Reconstruction Without External Sensors, 2014.
[11] ReconstructMe used in This War of Mine. ReconstructMe Website. Available at: http://reconstructme.net/2014/09/26/reconstructme-used-in-this-war-of-mine/

[10] https://www.youtube.com/watch?v=PbbdrtM2nxQ